

Discovering Emerging Entities with Ambiguous Names

Johannes Hoffart
Max Planck Institute for Informatics
jhoffart@mpi-inf.mpg.de

Yasemin Altun
Google Inc.
altun@google.com

Gerhard Weikum
Max Planck Institute for Informatics
weikum@mpi-inf.mpg.de

ABSTRACT

Knowledge bases (KB's) contain data about a large number of people, organizations, and other entities. However, this knowledge can never be complete due to the dynamics of the ever-changing world: new companies are formed every day, new songs are composed every minute and become of interest for addition to a KB. To keep up with the real world's entities, the KB maintenance process needs to continuously discover newly emerging entities in news and other Web streams. In this paper we focus on the most difficult case where the names of new entities are ambiguous. This raises the technical problem to decide whether an observed name refers to a known entity or represents a new entity. This paper presents a method to solve this problem with high accuracy. It is based on a new model of measuring the confidence of mapping an ambiguous mention to an existing entity, and a new model of representing a new entity with the same ambiguous name as a set of weighted keyphrases. The method can handle both Wikipedia-derived entities that typically constitute the bulk of large KB's as well as entities that exist only in other Web sources such as online communities about music or movies. Experiments show that our entity discovery method outperforms previous methods for coping with out-of-KB entities (called *unlinkable* in entity linking).

Categories and Subject Descriptors

H.3.1 [Information Storage and Retrieval]: Content Analysis and Indexing

Keywords

Entity Disambiguation, Emerging Entity Discovery, Knowledge Base Population

1. INTRODUCTION

Knowledge bases (KB's) like DBpedia, YAGO, Freebase, the Google Knowledge Graph, and Microsoft Satori contain

crisp facts about entities such as people, places, products, organizations, and events. These entities occur in Web contents; however they are usually mentioned in ambiguous form. Linking unstructured contents like text, tables or lists to knowledge bases (or other structured data) by mapping ambiguous names onto canonicalized (i.e., uniquely identifiable) entities, is the problem of named entity recognition and disambiguation, or NERD for short. Its first stage, NER, serves to detect phrases in text that are likely to denote named entities; the second stage, NED, performs the actual disambiguation of the recognized mentions into entities. This paper focuses on NED.

Our world is highly dynamic; so no knowledge base will ever be complete. New songs are composed every day, new movies are released, new companies are founded, there are new weddings, sports matches, and disasters and formerly unimportant people become noteworthy and should be added to a knowledge base. We refer to such entities as *emerging entities* (EE's) or out-of-knowledge-base (OOKB) entities.

1.1 Problem and Prior Work

This paper focuses on how NED methods can cope with this incompleteness, both during the disambiguation process itself as well as to extend the knowledge base with new entities. The problem is particularly challenging because new entities may appear under the same names as existing ones. For example, when hurricane "Sandy" occurred, several singers, cities, and other entities with the name "Sandy" already existed in Wikipedia (and thus also in the big knowledge bases). When faced with the first news about the hurricane, a smart NED method should distinguish the emerging entity from the existing ones, although the new entity may have very sparse context.

Prior methods for NED addressed this problem by thresholding on the scores they computed for mapping a given mention to different candidate entities. If none of the existing entities can be chosen with a score above a specified threshold, then the mention is marked as an emerging entity (see, e.g., [14, 23]). The approach has several shortcomings, though. First, its empirical quality is not that good, at least not in difficult situations. Hachey et al. [8] discussed this aspect for the TAC 2010 entity-linking benchmark. Second, score thresholds of this kind are hard to tune in a robust manner. The same threshold hardly works for different kinds of contents; so these methods may need frequent re-tuning and appropriate training data – not easily feasible in real applications. Third, deciding for each mention separately whether it is an in-KB entity or an EE based on a global

threshold comes with the risk that local decisions adversely affect the outcome for all mentions.

Example: Consider the first news about the disclosure of the Prism program by Edward Snowden. Both “Prism” and “Snowden” are ambiguous names, now being primarily used for the two emerging entities `PRISM_(program)` and `Edward_Snowden`. Suppose an NED method is fed with the input sentence “Washington’s program Prism was revealed by the whistleblower Snowden.”, from one of the early articles on this topic. The NED method needs to decide that “Prism” does not refer to the band or the TV network going by the same name, and should instead be mapped to an EE. However, “Snowden” is another mention for which the NED method has low mapping scores. With a thresholding decision per mention (like in state-of-the-art work), it may happen that “Prism” is below the threshold while “Snowden” is still above. Then, the NED method may choose to map “Snowden” to the city of Snowden in the state Washington, and map “Washington” to the this state, for coherence between these two entities. Thus, the poor thresholding decision for “Snowden” adversely affects the NED output for the seemingly easy mention “Washington”. Interestingly, it is exactly the best NED methods, with joint inference over all mentions, that are most penalized by such effects.

Another difficulty is that there could be multiple EE’s, all out-of-KB, with the same name. For example, the singer Katy Perry has announced a new album, “Prism”, which should not be confused with the surveillance program Prism. To deal with such situations, we need to consider the entire life-cycle of a knowledge base. Once we have identified a new EE, it should be added to the knowledge base in a representation that is strong enough to distinguish it from further EE’s with the same name that will possibly be encountered. At some point, after having observed an EE sufficiently often, it should be promoted, with assistance by a human editor, to a canonicalized entity in standard form. Both the separation from other EE’s and the canonicalization step need contextual information about the EE. In our approach, we harness salient phrases from the texts where the EE is observed.

1.2 Our Approach

Our goal in this paper is to develop a principled approach for the problem of NED with emerging entities, NED-EE for short. This comprises both distinguishing an EE from existing entities with the same name, and distinguishing an EE from an equally named other EE that is observed later.

Our approach builds on NED methods that use *characteristic (weighted) keyphrases* (or just keywords) of entities as part of their feature space. Wikipedia category names of an entity or href anchor texts that link to the entity are simple but highly effective forms of such keyphrases. NED methods along these lines include AIDA [11], Illinois Wikifier [23], Kulkarni et al. [14] among others. The NED-EE method developed in this paper works with any of these NED tools. Our NED-EE method is based on two techniques:

- assessing the confidence of the NED method’s mapping of mentions to in-KB entities, and
- enriching a possible EE with a keyphrase representation.

Equipped with these two assets, we can then extend the NED method by adding the potential EE to the method’s space of candidate entities, using keyphrase features.

For assessing the confidence, we devise several techniques,

based on perturbing the mention-entity space of the NED method. We also consider transforming the NED mapping scores into normalized confidence values. The confidence is then used to decide whether a new EE should be considered or not. Note that this is quite different from prior work, which would drop a mention in a low-confidence situation (i.e., declaring it unlinkable or mapping it to a global “nil” node) although candidate entities might exist for it. Our method never drops any mentions, but instead considers an EE as an *additional candidate*.

For constructing a keyphrase-based representation of EE’s, we need to identify phrases that are characteristic for the EE and uncharacteristic for the existing entities with the same name. This cannot be done directly using only the text where we observe the mention. Our approach first builds a global set of keyphrases that frequently co-occur with *any* entity of the ambiguous name. We do this by searching the Web for the name and extracting contexts and phrases from the retrieved matches. With appropriate weighting, we obtain a global representation of *all* entities with the given name, including the EE. Since we already have keyphrase features (or at least keyword features) for the in-KB entities, we can now compute a *model difference* between the global model and the union of all in-KB models, yielding the (weighted) keyphrases that are salient for the EE.

1.3 Contribution and Outline

Our key contributions in this work are the following:

- New building blocks for robustly coping with emerging entities (EE’s) in NED: confidence assessment for mention-entity mappings, and a keyphrase model for the emerging entities themselves.
- A principled method, referred to as NED-EE, that extends a family of NED tools by EE’s as additional candidate entities and makes judicious decisions about mapping mentions to existing entities or EE’s.
- Experiments with a news corpus that demonstrate the viability of our approach and the superior NED quality compared to state-of-the-art NED methods.

The rest of the paper is organized as follows. Section 2 presents the computational model for NED-EE and an overview of our system architecture. Section 3 discusses different techniques to assess the confidence of a mention-entity mapping. Section 4 introduces the extended keyphrase model to represent in-KB entities as well as EE’s. Section 5 presents the algorithm for performing NED-EE with integrated discovery of emerging entities. Section 6 shows our experiments. Section 7 discusses related work.

2. MODEL AND ARCHITECTURE

In the following we recap the computational model for named entity disambiguation (NED). NED receives a tokenized text document as input, for example, a news article of an incoming news stream. We assume that the text is first processed by a method for named entity recognition (NER), for example, the Stanford NER Tagger [7] which is based on a trained conditional random field. NER computes boundaries of phrases such that the phrases refer to named entities (without knowing the actual entities). We refer to the phrases as *mentions*. Different mentions may refer to

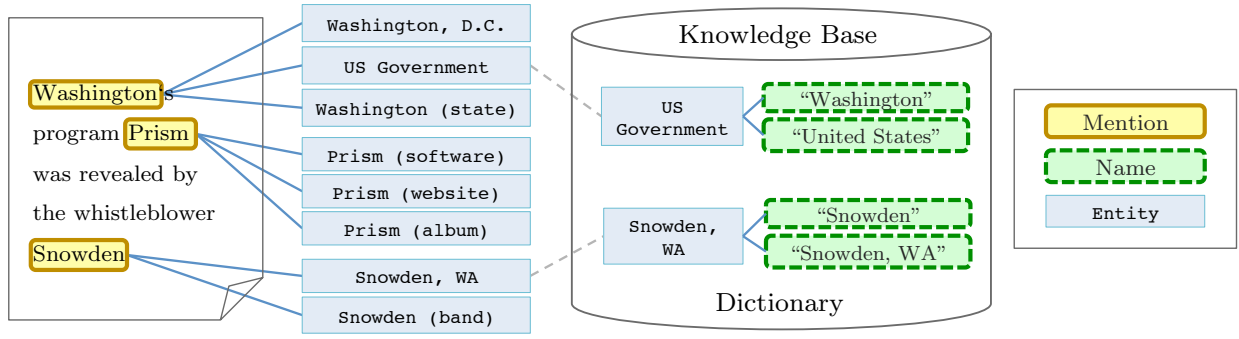


Figure 1: Disambiguation problem instance

the same entity, but NER is unaware of such situations.

NED: The goal of NED then is to link the set M of mentions in the input text to entities in a knowledge base. The knowledge base KB consists of the entity repository E containing all entities, as well as a dictionary $D \subset (N \times E)$ of pairs of mention names $\in N$ and entities $\in E$. For each mention $m \in M$, the dictionary D provides a set of candidate entities E_m for m . If D does not give any candidates for mention m , E_m is empty and the mention is trivially assigned to refer to an emerging entity (EE). Additionally, the KB also contains features F_e describing each entity e in the repository as a basis for the disambiguation. An example of a disambiguation problem is depicted in Figure 1.

Many disambiguation approaches use keywords or keyphrases as features, favoring entities whose keywords overlap strongly with the surrounding context of the name. Examples for such approaches are the Illinois Wikifier [23] and AIDA [11]. Both use Wikipedia as a source of entities. AIDA also uses a notion of coherence among entities, computed by keyphrase overlap of the entity keyphrases (KORE [9]) or using the Wikipedia link-graph ([19]). AIDA then disambiguates a set of mentions by determining a set of entities that match the context and have high coherence among each other using a graph algorithm on a representation as depicted in Figure 3. The m nodes are mentions in the input, connected to their candidates e by weighted edges, with weights derived from keyphrase-context similarities.

NED-EE: In established NED methods, mentions for which all candidate entities have only low scores according to the method’s similarity and coherence measures are often considered unlinkable and mapped to an artificial node “nil” (the same node for all unlinkable mentions). In the current paper, we extend this NED model into an NED-EE setting which integrates emerging entities (EE’s) in a more flexible and principled manner. We assume that each mention m could also be an EE and introduce an additional entity node as a candidate for the disambiguation. In Figure 3, there are 3 such EE nodes, marked by the suffix `_EE`. Section 4 will later describe how we obtain keyphrase features for this new out-of-KB entity. Once we have expanded the candidate space E_m and the feature space this way, we can feed the NED-EE problem back to the NED method, which can now treat the EE the same way as it treats in-KB entities.

Although we may apply this expansion of E_m for every mention, it could be wise to omit it for such mentions that have a high-confidence mapping to an existing in-KB entity. Adding an EE candidate would potentially introduce noise

even in such clear cases, and would also come with the extra cost of computing keyphrase features. Section 3 will later describe how we compute confidence measures and select mentions for which EE candidates need to be created. The outlined building blocks are components of our system architecture for the NED-EE method presented in this paper. Figure 2 gives a visual overview. Our implementation uses the Stanford NER Tagger for NER, and YAGO [10] for the knowledge base. Our primary choice for the NED component is AIDA [11]. Note, though, that any NED method with suitable API can be plugged in.

3. DISAMBIGUATION CONFIDENCE

We have devised and studied various ways of modeling and computing the confidence for the individual outputs (i.e., mention-entity pairs) of an NED method. In the following, we first discuss a simple technique, based on normalizing the scores that most NED methods provide anyway. Subsequently, we introduce two more elaborate approaches, based on perturbing the input of NED.

3.1 Normalizing Scores

The output scores that most NED systems provide for mention-entity pairs can often be interpreted as a notion of confidence. For probabilistic NED models, the output probability of a mention-entity pair being the right choice would be a direct confidence measure. However, many good NED methods are not probabilistic and yield scores that are not normalized in any way and cannot be directly interpreted. Even for probabilistic methods, all output probabilities could be very small because of (conditional) independence assumptions in the model and not easy to interpret.

To handle a-priori unbounded scores, we propose to normalize scores for m - e pairs on a per mention basis as follows. For each candidate e of a given mention m , compute:

$$\text{norm}_{\text{score}}(m, e) = \frac{\text{score}(m, e)}{\sum_{e_i \in E_m} \text{score}(m, e_i)}$$

The intuition is that if the best entity for m accumulates high score mass with respect to the total score mass of all other candidates, the confidence that the disambiguation is correct is high. So we define the confidence for mapping mention m correctly as follows:

$$\text{conf}_{\text{norm}}(m) = \text{norm}_{\text{score}}(m, \arg \max_{e \in E_m} \text{score}(m, e))$$

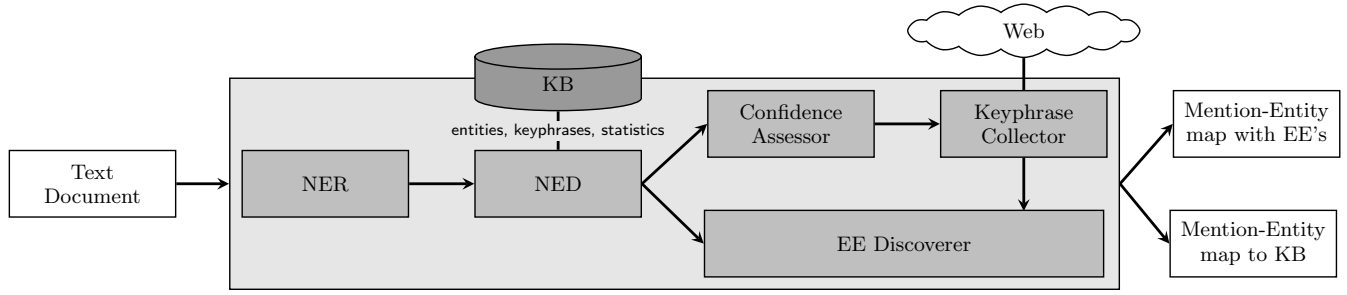


Figure 2: System architecture for NED-EE

3.2 Perturbing Mentions

Intuitively, we should have high confidence in a NED method choosing entity e for mention m if that choice remains invariant under many variations of the input context. For example, if leaving out or changing some of the sentences of a news article does not affect the m - e mapping, this is a sign of a robust and high-confidence choice.

One specific technique along the lines of such a perturbed input approach is to drop mentions from the input text. There are many ways of dropping one, two, or a small number of mentions; so we choose them randomly and iterate such perturbations (in the spirit of i.i.d. – independent and identically distributed – sampling).

We feed each of the perturbed inputs into a NED method, treating NED like a black box, and collect the outputs of each run: the best m - e mappings for the given input context. We keep track how often different entities are chosen for the same mention, and aggregate over all perturbations. This procedure works as follows:

1. For a given set of mentions M , run the NED(M) algorithm to get the initial results (m_i, e_i)
2. Keep two counters for each m_i : c_i, k_i
3. For a fixed number of times (e.g. $k = 500$):
 - 1 Choose R as random subset of M
 - 2 For each $m \in R$ increase k_i , keeping track of how often m was actually present in the input
 - 3 Run NED on the random subset, producing new results $(m_i, r_i) = \text{NED}(R)$
 - 4 Increase the mention counter c_i if the initial entity e_i equals r_i , keeping track of how often the initial entity remained stable

Based on these counters over all perturbations, the confidence for a mention m_i is then computed as:

$$\text{conf}_{\text{perturb}}(m_i) = \frac{c_i}{k_i}$$

3.3 Perturbing Entities

Another way of perturbing the input to the NED method is to fix certain choices of entities for a small set of mentions, and study how the other – freely assignable – mentions are then best mapped. To this end, we first run the NED method at hand to compute its best output, ideally using joint inference over all mentions. Then, we change some of

the chosen entities, forcing the corresponding mentions to be mapped to “incorrect” entities – incorrect as far as the NED method over all mentions would see it. Again, we choose the artificially “switched” entities at random, and we repeat this kind of perturbation many times.

The intuition here is that we should have high confidence in an m - e pair if switching entities for a small number of other mentions does not affect the final choice of e by the NED method. So we reward disambiguations that remain stable in the presence of perturbations.

Treating NED as a black box again, we run the following procedure:

1. For a given set of mentions M , run NED(M) for initial results (m_i, e_i)
2. Keep two counters for each m_i : c_i, k_i
3. For a fixed number of times (e.g. $k = 500$):
 - 1 Choose random subset R of M
 - 2 For each e_i the (m_i, e_i) pairs in R , force-map m_i to an alternate e_j , choosing e_j in proportion to the scores of NED(M) (or just uniformly)
 - 3 For each $m \in M \setminus R$ increase k_i , keeping track of how often m was actually present in the input
 - 4 Run NED on $M \setminus R$, but leaving the removed mention strings in the input text and the forced-mapped alternate entities in the coherence model of the NED method; producing new results $(m_i, r_i) = \text{NED}(M \setminus R)$
 - 5 Increase the mention counter c_i if the initial entity e_i equals r_i , keeping track of how often the initial entity was chosen again

The fraction of times e_r is equal to e_i (the result of the original disambiguation) is the estimated confidence for a mention, the same as with the method perturbing the input.

Note that both perturbation methods can be emulated on a suitable representation of the NED problem without actually re-running the full NED method multiple times.

4. EXTENDED KEYPHRASE MODEL

Many NED methods use keyphrases as entity features. The NED-EE method developed in this paper primarily builds on the AIDA tool for standard NED (without EE’s) [11]; so the techniques presented in the following are explained with respect to the AIDA method. Note, however, that it is straightforward to carry our NED-EE method over to other systems with similar features, e.g., the Illinois Wikifier [23].

4.1 Keyphrases for Existing Entities

All NED methods that use entity-specific keyphrases as features pre-compute these keyphrases and their weights, often based on Wikipedia href anchor texts or similar data. We utilize such pre-computed keyphrases for in-KB entities as well, but extend this approach by additionally harvesting keyphrases from document collections.

We perform this only for the entities that high-confidence mentions are mapped to by the given NED method; so we harness the confidence assessment presented in the previous section. We do this in order to enrich the keyphrase representation of in-KB entities, which in turn gives us a better handle to contrast these existing entities against the EE candidates for this and other mentions.

The dynamic keyphrase acquisition works as follows:

1. Retrieve the context for the high-confidence mention: all sentences in the window (5 preceding, 5 following) surrounding the mention
2. Identify keyphrases in these sentences:
 - 1 Part-of-speech tag each sentence
 - 2 Extract all sequences conforming to a set of pre-defined part-of-speech tag patterns. These patterns extract all proper nouns as well as technical terms such as “surveillance program” (as defined in [13]), both serving as keyphrase candidates
3. Output each distinct entity-keyphrase and entity-keyword (every word occurring in a phrase), for subsequent co-occurrence counting

The NED method that we specifically extend by our NED-EE method, AIDA, uses these counts to assign weights to keyphrases and keywords based on their frequency (IDF) and mutual information (MI). The IDF score is a global score computed once per keyphrase and keyword. The MI score is entity-specific and can be computed by contrasting the occurrence count of the entity c_e , the keyphrase c_k , and the intersection c_{ek} . All of these counts are the sum of both the dynamically harvested and the in-KB counts. From each sum we derive (estimated) probabilities by dividing by the collection size n , e.g. $p(e) = c_e/n$. The normalized pointwise mutual information score for an entity e and a keyphrase k is then computed as follows:

$$\text{npmi}(e, k) = \frac{\text{pmi}(e, k)}{-\log p(e, k)},$$

where

$$\text{pmi}(e, k) = \log \frac{p(e, k)}{p(e)p(k)}.$$

Keyphrases with $\text{npmi}(e, k) \leq 0$ are discarded for the actual NED.

4.2 Modeling Emerging Entities

The key to NED-EE is to make EE’s first-class citizens. To this end, we introduce, for each mention, an additional out-of-KB candidate entity that is initially represented just by the mention string. Recall that the interesting and difficult case is when that string is ambiguous and may also refer to one or more already existing in-KB entities. The point of NED-EE is that we nevertheless introduce a new

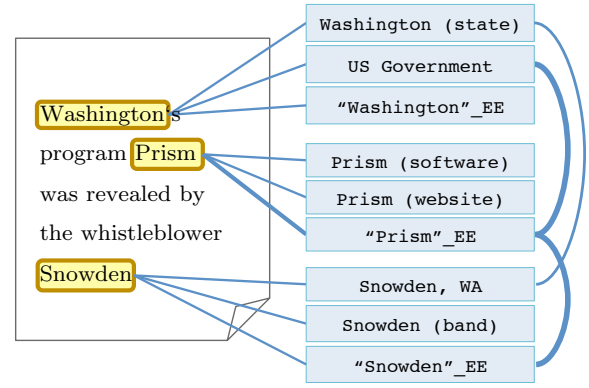


Figure 3: Disambiguation graph with EE nodes

EE candidate node for each mention. Figure 3 illustrates this expanded situation.

For an informative contrast between equally named existing entities and such newly introduced EE’s, we need a richer representation of the EE nodes; merely using the mention string is, by itself, meaningless. Therefore, we also compute characteristic keyphrases for each EE. We do this analogously to the dynamic keyphrases computed for in-KB entities, as described in the previous subsection: constructing text windows around mentions, extracting noun phrases, and so on.

Exploiting news streams: Since we are mainly aiming to perform NED-EE over news streams, we address this problem by mining EE-specific keyphrases from chunks of news articles within a news corpus or incoming stream. For the given input document that NED-EE needs to handle, we identify all news articles whose timestamps are in the vicinity of the publication date and time of the input document, for example, the same day and the preceding week, or just the same day. For social-media postings, the temporal granularity would be smaller, say a few hours instead of a few days. The assumption here is that there is likely a fair amount of redundancy, and many EE’s would appear in more than one article - but everywhere in ambiguous form.

Model difference: Since we cannot a priori distinguish an in-KB and an EE with the same name from each other, we cannot directly compute keyphrases that are truly characteristic for the EE and uncharacteristic for the in-KB entities with the same name. This is a fundamental problem that we solve as follows. We actually compute keyphrases for the EE name, that is, the mention string, and the obtained phrases are actually characteristic for *all* entities with that name, covering both in-KB and out-of-KB entities. Now, since we already have a keyphrase model for all the in-KB entities with this name, potentially even dynamically enriched by the technique of the previous subsection, we can compare the keyphrase model for the name with the keyphrase model for the in-KB entities. In a nutshell, the set difference between these models is the characteristic model for the EE. As all phrases are weighted, this *model difference* is more elaborate, as shown in Algorithm 1.

The final result of this keyphrase acquisition and weighting procedure for EE nodes is that we have a rich and crisp characterization of each introduced EE that is discriminative

Algorithm 1: Harvesting keyphrases for EE

Input: document d , ambiguous name n

Output: EE-keyphrases and weights

1. Define a suitable chunk of news articles based on the input document’s publishing date and time.
 2. Harvest all keyphrases K_n for all mentions m of n from all articles in the news chunk as explained in Subsection 4.1, resulting in a list of tuples of $(k_n, b) \in (K_n \times \mathbb{N})$ of name-keyphrase co-occurrence counts.
 3. Get all in-KB entity candidates E_n for n , and their keyphrases K_E with their co-occurrence counts $(k_e, c) \in (K_E \times \mathbb{N})$. The EE-count d for each keyphrase in K_n is adjusted by the collection size balance parameter $\alpha := \frac{\text{KB collection size}}{\text{EE collection size}}$ in the following way: $d = \alpha(b - c)$, where c is 0 if the keyphrase is not in K_E . The fraction accounts for the differences in collection sizes. In our use case, the KB size is the number of entities in YAGO, the EE size is the number of news articles the keyphrases are extracted from.
 4. Adjust the occurrence counts of n : analogously to name-keyphrase co-occurrence counts, subtract the occurrence counts for all candidate entities $e \in E_n$ of n . The count is balanced with the same scalar α as the keyphrase co-occurrence.
 5. Compute all the EE-keyphrase MI weights based on the adjusted counts. Compute all keyphrase IDF weights based on the combined document frequencies, where both the document frequencies and the collection sizes are balanced by α .
-

against the in-KB entities that share the EE’s name. This principle model is the basis for actually deciding whether a mention refers to an existing or an emerging entity. The extensions to the NED method for this purpose are presented in the following section.

5. DISCOVERING EMERGING ENTITIES

A knowledge base comes with an entity repository and a dictionary listing names of the entities. Both are incomplete due to the dynamics of the world. There are different tasks in maintaining and growing the knowledge base:

- To increase the coverage of the dictionary, new names or aliases for in-KB need to be discovered.
- Emerging entities can have a new name or make an existing name more ambiguous. Once an emerging entity is discovered, it needs to be added to the knowledge base with a representation suitable for further disambiguation tasks.
- The keyphrase models of both in-KB entities and emerging entities need to be continuously updated as well, by enriching them with additional phrases and by adjusting weights.

In the following, we focus on the second task, the core of the NED-EE problem. The first task, discovering new names for existing entities, is orthogonal to the topic of this paper

and thus out of scope. The third task has essentially been solved by the model and techniques of the previous section.

Algorithm 2: Discovering Emerging Entities

Input: mentions M , threshold t

Output: emerging entities EE

1. Run the disambiguation for each $m \in M$ to get the initial entity assignment: $(m_i, e_i) = \text{NED}(M)$;
 2. Compute the confidence c_i for each (m_i, e_i)
 3. Set e_i to EE for each m_i where $c_i < t$
 4. For each m_i where e_i is not set to EE:
add EE-candidate e_o with EE-keyphrase features
add m_i to M' , the to-be-disambiguated mentions
 5. $(m_i, e'_i) = \text{NED}(M')$
 6. Set e_i to EE for each m_i where e'_i is e_o
-

A general method for discovering emerging entities with any kind of underlying NED system is described by Algorithm 2. It first runs the regular NED method, then (optionally) applies a threshold on the confidences to drop low-confidence mentions in favor of EE. A variation for step 3 would be to add EE-placeholders e_o as candidates instead of assigning EE directly, allowing the subsequent NED run to determine the result. After this step, the disambiguation problem instance is extended with EE-placeholders with the appropriate keyphrase representation as input to the second run of the NED. Choosing the threshold as 0.0 creates the special case of only using the EE representation to discover emerging entities, running the NED only once.

Emerging entities are especially interesting in the setting of a continuous stream of news, where they should be identified as quickly as possible to cover the latest events and associated people, organizations, etc. The discovery of emerging entities in a news corpus works by acquiring keyphrase from recent news chunks and constructing EE models, in terms of weighted keyphrases, as described in the previous section.

For this news setting, we introduce a new hyper-parameter γ to balance the influence of Wikipedia-derived weights and news-based weights in the final NED scores. Because of the very different styles and resulting differences in frequencies, merely reconciling these two assets based on counts alone is not sufficient. Thus, all edge weights in the disambiguation graph connected to at least one EE-placeholder are multiplied with a factor of γ , tuned on withheld data.

The output of running NED on the NED-EE-based extended disambiguation graph is a mapping of all mentions to either in-KB or emerging entities. The mentions that are mapped to the same EE can be grouped together, and this group is added – together with its keyphrase representation – to the KB for the further processing in the KB maintenance life-cycle, as shown in Figure 4.

6. EXPERIMENTS

We have experimentally evaluated two main contributions of this paper: assessing the confidence in the disambiguation of mentions, discussed in Section 6.1, and discovering emerging entities for ambiguous names, discussed in Section 6.2.

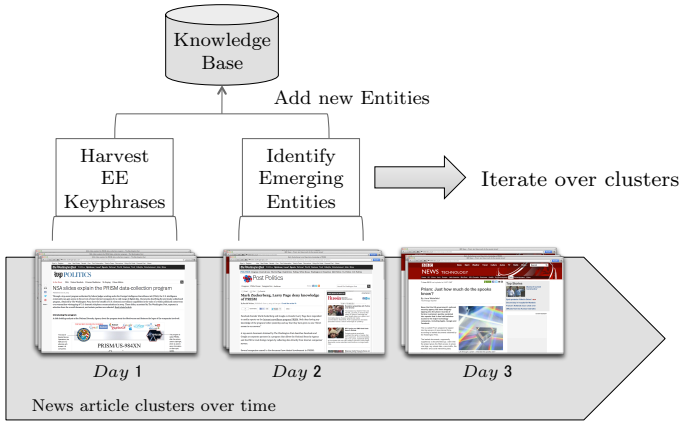


Figure 4: Identifying EE on news data

6.1 Disambiguation Confidence

The confidence in the disambiguation of a mention to an entity is important both for down-stream applications using the results of NED, as well as for the method itself to improve its own quality. In Section 3, we introduced three methods for confidence assessment, one based on normalizing NED scores and two based on input perturbations. Preliminary studies, not presented here for brevity, suggested that of the three assessors, a linear combination of normalized scores and entity perturbation (with coefficients 0.5) gives the best results. Therefore, we use only this CONF method in the following experiment, showing its influence on the overall NED quality.

Experimental setup: One measure for the goodness of a confidence assessor is the quality of the ranking of mentions that the assessor enables. For each document, we rank all its mentions in decreasing order of confidence. We then compute the precision@ $k\%$ by disposing the mentions below $k\%$ and computing the fraction of correctly disambiguated mentions with respect to the ground truth for the remaining mentions, evaluating how well the confidence reflects the actual correctness. We do this for all k between 1 and 100% to get the average precision for a single document, and take the mean of all averages of all documents to compute the mean average precision (MAP) for one ranking method on the whole corpus. As an additional measure for all the methods whose disambiguation scores are interpretable as confidence we give the precision at the confidence values 95% and 80%. We also give the number of mentions in the document collection that have at least this confidence.

For this study, we used the CoNLL-YAGO testb collection [11] with ground-truth annotation for 4,485 mentions in 231 newswire articles. As underlying NED systems we primarily used AIDA, but also studied the Illinois Wikifier.

The confidence-estimation via normalization for AIDA takes different kinds of scores as input:

keyphrase: This is computed based on how well the entity keyphrases match the context of the mention.

weighted-degree: This score combines the keyphrase score and all the relatedness scores of a given entity in the disambiguation graph (see Figure 3) by summing the individual scores of all graph edges adjacent to the entity. The benefit of this score is that it captures how

well the given entity fits to the other candidate entities instead of being restricted to the mention-entity score only. This score is used for the normalization component of the CONF method.

We compare the following competitors to our CONF method:

prior Disambiguation using only the mention-entity prior, estimated by counting how often the mention is used as a link anchor in Wikipedia to link to a given entity.

AIDA_{coh} AIDA graph-coherence disambiguation ranked by keyphrase score (as in the original AIDA work).

IW Ranked by Illinois Wikifier [23] linker score.

Measure	Competitors			Ours
	prior	AIDA_{coh}	IW	CONF
Prec@95%conf	91.98%	-	-	97.77%
#Men@95%conf	2020	-	-	1661
Prec@80%conf	82.55%	-	-	93.78%
#Men@80%conf	3038	-	-	2509
MAP	87.87%	86.75%	67.11%	93.72%

Table 1: Influence of confidence assessors on NED quality

Results: The results for this comparison are given in Table 1. We see that CONF, the linear combination of the normalized weighted degree and entity perturbation, leads to substantial improvements in precision and MAP, at different cut-off levels for confidence. The precision-recall curves in Figure 5 show a similar picture. Note that the prior is most probably correct for mentions with a very high prior for their most popular entity (by definition), so the initial ranking of the prior and also the MAP are very good. However, it drops more sharply, as seen in Figure 5.

An interesting observation is the result for Precision@95% confidence. Being nearly 98% correct at the 95% confidence-level is an important factor in extending the keyphrases for existing entities without introducing a lot of noise. The number of entities that can be disambiguated with this confidence is not marginal, either, it's 1/3 of the mentions in the collection.

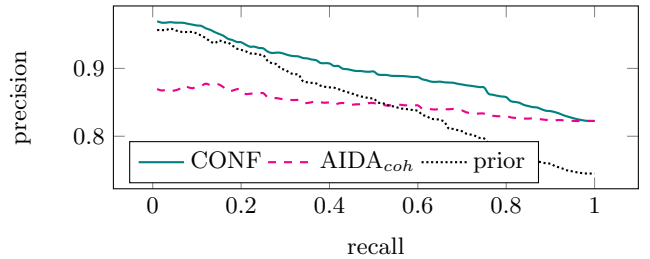


Figure 5: Precision-recall curves with different confidence assessors

6.2 Emerging Entity Discovery

6.2.1 Dataset

A natural setup to evaluate the discovery of emerging entities would be to mine the keyphrases for the EE representation from the Web, subtracting the existing keyphrases

from a truly global representation. However, these experiments are beyond the scope of an academic setting. Thus, we evaluate discovery in the setting of identifying emerging entities in news. There are no recent corpora available that are both chronological in nature and labeled with entities. The news content in the annotated dataset published in [11] and used above predates Wikipedia and is not useful here.

To evaluate our emerging entity discovery, we annotated 150 Associated Press news articles published on October 1st and 150 published on November 1st, 2010, taken from the GigaWord 5 corpus [22]. We automatically recognized mentions using the Stanford NER system, then cleaned them manually to remove spurious results. Each mention was annotated with **EE** if the correct entity was not present in Wikipedia as of 2010-08-17. Otherwise, we annotated the correct entity. Every mention was annotated by two students, conflicts were reconciled by the authors.

documents	300
mentions	9,976
mentions with emerging entities EE	561
words per article (avg.)	538
mentions per article (avg.)	33
entities per mention (avg.)	104

Table 2: AIDA-EE GigaWord dataset properties

The dataset is available online¹ in the form of annotations for the original data, statistics are given in Table 2.

6.2.2 Evaluation Measures

We are evaluating the quality of the NED and **EE** methods with measures defined over the following sets: D , the collection of documents; G_d , all unique mentions in document $d \in D$ annotated by a human annotator with a gold standard entity; $G_d|_{EE}$, the subset of G annotated with an emerging entity **EE**; $G_d|_{KB}$, the subset of G annotated with an existing, in-KB entity; and A_d , all unique mentions in document $d \in D$ automatically annotated by a method. The measures used are:

- **Micro Average Accuracy**, the fraction of correctly disambiguated gold mentions in the whole collection:

$$\text{MicA} = \frac{|\bigcup_{d \in D} G_d \cap \bigcup_{d \in D} A_d|}{|\bigcup_{d \in D} G_d|}$$

- **Document Accuracy**, the fraction of correctly disambiguated mentions in a single document d :

$$\text{DA}_d = \frac{|G_d \cap A_d|}{|G_d|}$$

- **Macro Average Accuracy**, the document-averaged fraction of correctly disambiguated mentions:

$$\text{MacA} = \sum_{d \in D} \frac{\text{DA}_d}{|D|}$$

- **EE Precision**, the correct fraction of all mentions disambiguated as **EE**:

$$\text{EE Prec}_d = \frac{|G_d|_{EE} \cap A_d|_{EE}|}{|A_d|_{EE}|}$$

The **EE Precision** is averaged over all documents.

- **InKB Precision**, the correct fraction of all mentions disambiguated as an existing, in-KB entity:

$$\text{InKB Prec}_d = \frac{|G_d|_{KB} \cap A_d|_{KB}|}{|A_d|_{KB}|}$$

The **InKB Precision** is averaged over all documents.

- **EE Recall**, the fraction of all **EE**-labeled gold standard mentions correctly recognized by a method:

$$\text{EE Rec}_d = \frac{|G_d|_{EE} \cap A_d|_{EE}|}{|G_d|_{EE}|}$$

The **EE Recall** is averaged over all documents.

- **EE F1**, the harmonic mean between the **EE Precision** and **Recall**, calculated per document then averaged.

6.2.3 Experiments

All **EE** experiments are run on the 150 annotated documents from 2010-11-01 in the AIDA-EE GigaWord corpus described in Section 6.2.1, adjusted in the following way:

- The mentions that are not in the entity dictionary are removed, as they can be resolved trivially.
- The full name of (existing) person entities is replaced with the last token of the name only (assumed to be the family name), e.g. we transformed “Edward Snowden” to “Snowden”. This is to increase ambiguity and counter data sparsity issues in this small corpus.
- Mentions that occur in less than 10 distinct articles in the last 3 days before the day of the actual article (in the full GigaWord corpus, which has 7 different news sources) are removed. This is done to restrict the mention space to such mentions where the **EE** method has data.

This leaves a total of 3,436 mentions, out of which 162 are both ambiguous and refer to an emerging entity.

We estimated hyper-parameters on the training set of 150 documents from 2010-10-01 containing 3,935 mentions out of which 187 are **EE**, adjusted in the same way. In the case of the competitors, the hyper-parameter is the threshold. For the emerging entity method we estimated the γ parameter balancing the **EE** entity weight against regular entities, the number of days to look back when harvesting keyphrases, and the maximum number of keyphrases to use for each entity (to better balance very popular entities with a large number of keyphrases against long-tail entities). The best values are given below in the description of the methods. When harvesting the keyphrases for existing entities, the number of days is set to 30.

The methods should label each (Stanford NER recognized) mention with either the correct entity or **EE**. Note that we restrict the input to the labeled mentions to compare the method’s ability to distinguish between existing and new entity, not its ability to recognize names in the input text.

The competitors we compare against are:

IW Ranked by Illinois Wikifier linker score. The threshold was estimated as -0.66 . Note that we are not using the Wikifier’s internal mechanism but perform the equivalent operation of thresholding after the actual run to simplify the integration in our framework.

¹Dataset: <http://www.mpi-inf.mpg.de/yago-naga/aida/>

Measure	Competitors			EE	
Measure	AIDA _{sim}	AIDA _{coh}	IW	EE _{sim}	EE _{coh}
Micro Avg. Acc.	75.03%	75.81%	62.34%	53.75%	62.28%
Macro Avg. Acc.	73.22%	72.58%	62.79%	48.90%	60.90%
EE Prec.	72.84%	53.49%	66.59%	97.97%	93.92%
EE Rec.	89.09%	90.92%	90.88%	70.69%	71.72%
EE F1	66.61%	49.80%	63.89%	68.92%	67.92%

Table 3: Emerging entity identification quality on 150 labeled news documents

Measure	Competitors			NED-EE	
Measure	AIDA _{sim}	AIDA _{coh}	IW	AIDA-EE _{sim}	AIDA-EE _{coh}
Micro Avg. Acc.	76.02%	75.81%	70.31%	76.11%	71.33%
Macro Avg. Acc.	73.40%	72.58%	70.52%	72.90%	70.40%
EE Prec.	72.84%	53.49%	66.59%	97.97%	93.92%
InKB Prec.	73.96%	72.93%	70.51%	76.12%	73.34%

Table 4: Quality of NED-EE (EE as preprocessing followed by regular NED)

AIDA_{sim} AIDA keyphrase based disambiguation ranked by keyphrase confidence, thresholded at 0.15.

AIDA_{coh} AIDA graph link-coherence disambiguation ranked by the CONF confidence, thresholded at 0.11.

For both our emerging entity methods the EE keyphrases are harvested from the previous 2 days, the maximum number of keyphrases per entity is set to 3000, and γ is set to 0.04 for EE_{sim} and 0.06 for EE_{coh}. The knowledge base and dictionary is YAGO2, built from the Wikipedia snapshot from 2010-08-17.

EE_{sim} AIDA keyphrase based disambiguation including EE placeholders and harvested keyphrases for existing entities.

EE_{coh} AIDA graph coherence based disambiguation including EE placeholders and harvested keyphrases for existing entities, using KORE relatedness scores for the coherence. Link-based coherence is not applicable, as it relies on the Wikipedia link graph that does not contain EE placeholders.

The results in Table 3 show that the EE methods clearly outperform the competitors in terms of EE Precision and F1. The EE_{sim} method achieves a very high precision of more than 98% with a recall of 71%. This result includes harvested keyphrases for existing entities, which improves the EE precision by 7 points and the EE F1 by 4.5 points over the same method using only EE-keyphrases (see Figure 6). Using graph coherence, EE_{coh}, does not change the results much in terms of F1, however it trades off precision in favor of recall. The reason for the average F1 being lower than both the average precision and recall in all cases is that F1 becomes 0 if either precision or recall are 0, which happens for a number of documents.

The results also show that the EE-methods, which include the regular entity disambiguation step, do not perform so well in the general NED setting, reflected by lower accuracy numbers. We also experimented using the EE-method as a preprocessing step, removing all the recognized EE mentions for a subsequent NED run. This NED run uses the best non-EE AIDA out of the box, which is the AIDA_{coh} variant

without thresholding. The results are shown in Table 4, where each method from Table 3 is used as pre-processing step to identify emerging entities. The EE Precision stays the same, as EE’s are used as input and stay untouched. Both accuracy and InKB precision numbers show that using pre-identified emerging entities improves AIDA-EE, which achieves the best NED quality in this setting.

However, the results of the AIDA-EE_{coh} method reveal that judging the quality of the EE methods by the EE measures alone is not sufficient and that they need to be evaluated in the full NED setting. Even though the same NED method is run for the outputs produced by EE_{sim} and EE_{coh}, which are similar in terms of the EE measures, AIDA-EE_{coh} performs worse. This is because EE_{coh} tends to make mistakes that create a more difficult input for the actual NED method. To given an example, EE_{coh} falsely labels the mention “Kennedy” as EE instead of John F. Kennedy in one of the test documents, removing a mention from the input that would be solved correctly by the NED and is crucial to disambiguate the remaining mentions in the same document.

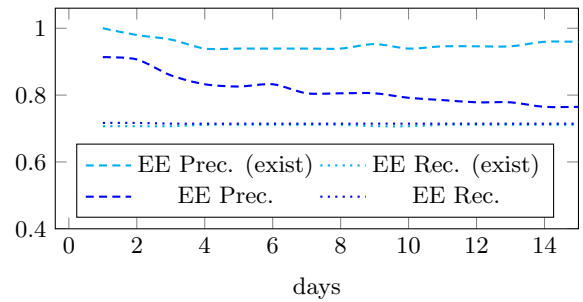


Figure 6: EE discovery over number of days used for the EE representation

Harvesting keyphrases for in-KB entities makes the NED-EE method more robust. Figure 6 shows how the precision of the EE-discovery degrades once keyphrases for representing the EE-placeholder are harvested from more days. Adding keyphrases for existing entities not only improves the precision, but keeps EE-placeholders from dominating the exist-

ing entities, stabilizing precision over time. The keyphrases for existing entities were harvested from the full month preceding the test data (100,000 GigaWord documents) – here noise is not a big issue as we only include keyphrases for very high confidence disambiguations ($\geq 95\%$), which we found to be accurate for 98% of mentions (cf. Table 1).

One more interesting observation is that the achieved EE F1 of the thresholding approaches is better on the training data where the threshold is optimized on, but it does not generalize well. This was remarked numerous times [8, 23] before. Explicitly representing the EE entity instead of deciding based solely on the features for existing entities is more robust. We also evaluated TagMe [6], however the numbers are far worse due to the fact that TagMe does not accept pre-defined mentions in the input. This makes results nearly incomparable, so we do not give numbers.

7. RELATED WORK

Entity Disambiguation. Work in disambiguating named entities to knowledge bases (usually derived from Wikipedia) was first done by Bunescu and Pasca [2]. It was followed by Mihalcea and Csomai [17], who took a less restrictive approach and considered all Wikipedia articles as disambiguation targets, including conceptual ones like “music”.

Entity Disambiguation with Joint Inference. The first method to acknowledge the use of doing a joint disambiguation of all mentions in the input document at the same time was by Cucerzan [5]. However, his method still disambiguated each mention separately, trying to capture the coherence by taking into account the suitability of an entity with all other candidate entities in the document. A similar approach was taken by Milne and Witten [20], where the relatedness to unambiguous mentions in the context was used to improve the disambiguation for ambiguous names. Ratnikov et al. [23], as well as Ferragina and Scaiella [6], while extending the features and thus improving the disambiguation quality, still disambiguate each mention individually. The first work to do a joint disambiguation by means of an ILP-representation was by Kulkarni et al. [14], followed by Hoffart et al. [11] who solve the disambiguation using a greedy graph algorithm.

Discovering Emerging Entities. Nearly all of the introduced methods already address the problem of out-of-KB entities. Most of them use a threshold on the best scoring entity. The disambiguation method by Kulkarni [14] uses a trained threshold to identify emerging entities (EE). Ratnikov et al. [23] refine the EE discovery by specifically training a separate component that decides whether or not to annotate a mention as EE, using an SVM on results of the initial disambiguation step. The final decision whether to annotate is again taken by thresholding on the SVM score. Other works like TagMe [6, 4] either used a simple threshold or, like Hoffart et al. [11], ignored the EE problem completely, focusing only on the disambiguation part.

The key distinction between these works and our work is the way the discovery of non-existent entities is handled. Previous work did not do any modeling of the unknown entities, but rather based the decision on the absence of good context for the existing entities.

Understanding New Names. Other work addresses the discovery of emerging entities from a different angle. Both the work by Nakashole et al. [21] and by Lin et al. [16] assume that existing names always refer to existing entities,

and that only new names are of interest. Their focus is then on trying to assign a type to the emerging entity as a first step to make it useful. For example, the name “Edward Snowden” was previously unknown (to Wikipedia), and would be assigned the types `person`, `computer specialist`, and `whistleblower`, but the methods would ignore the name “Prism” as there are existing entities in Wikipedia going by that name. Our work makes the same assumption that new names refer to new entities, but also tries to discover new entities going by the name of existing entities.

Keyphrase Mining. There are a number of works on extracting keyphrases from single documents, both in an unsupervised [18] and supervised manner [25]. Our requirements for keyphrases are more specific, as the extraction needs to fit into the overall architecture of our NED system. Related work that complements our keyphrase harvesting is by Taneva and Weikum [24], which focuses on harvesting context for long-tail entities. Another recent work by Li et al. [15] creates a background model for unknown entities to get additional context for entities for the disambiguation. We do not compare our methods, as our main focus is on the discovery of new entities, whereas they are trying to improve the disambiguation quality for existing ones.

Alias Discovery. A related problem to discovering when an existing name refers to a new entity is the discovery of new names for existing entities, also known as alias discovery [3, 1, 12]. The key difference is that alias discovery usually does not assume a canonical entity for which to harvest names, but takes existing name-alias pairs as input.

8. CONCLUSIONS

We presented a new approach to discovering emerging entities with ambiguous names by discriminating them against existing entities in a knowledge base. Our method is based on the principle to make emerging entities first-class citizens in the NED method itself, by introducing new EE’s to the candidate space of the method. To make this idea work, we devised and studied confidence assessors for entity mentions, and we developed an extended form of keyphrase representation that captures EE candidates in terms of a model difference. Our experiments show the viability of our NED-EE approach, the high precision in discovering emerging entities, and improvements in the precision of the NED output (regarding also the disambiguation for in-KB entities).

These contributions are part of our ongoing work in better understanding the life-cycle of knowledge base maintenance. Once we have discovered EE’s, we can add them to the KB. But we will observe them again in incoming news or social-media postings, so we should gradually enrich our keyphrase model and knowledge about them. At some time, the EE’s detected so far should become regular in-KB entities, with a canonicalized representation, perhaps after approval and with assistance by a human knowledge editor. Our ongoing and future work addresses such life-cycle aspects by developing methods and tools that support knowledge communities.

Acknowledgements. Thanks to Stephan Seufert for his help and for contributing a beautiful figure, to Vasanth Venkatraman for his help with the annotation interface, and to Dobromir Vasilev and Artem Galushko for the extra effort when annotating the dataset. A personal thank you from Johannes to the team at Google Zurich hosting him during his internship for their great support, especially to Yasemin.

9. REFERENCES

- [1] D. Bollegala, Y. Matsuo, and M. Ishizuka. Automatic Discovery of Personal Name Aliases from the Web. *Knowledge and Data Engineering, IEEE Transactions on*, 23(6):831–844, 2011.
- [2] R. Bunescu and M. Pasca. Using Encyclopedic Knowledge for Named Entity Disambiguation. In *11th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2006, Trento, Italy*, pages 9–16, Apr. 2006.
- [3] S. Chaudhuri, V. Ganti, and D. Xin. Mining document collections to facilitate accurate approximate entity matching. *Proceedings of the VLDB Endowment*, 2(1):395–406, 2009.
- [4] M. Cornolti, P. Ferragina, and M. Ciaramita. A Framework for Benchmarking Entity-Annotation Systems. In *22nd international conference on World Wide Web, WWW 2013, Rio de Janeiro, Brazil*, pages 249–260, 2013.
- [5] S. Cucerzan. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2007, Prague, Czech Republic*, pages 708–716, 2007.
- [6] P. Ferragina and U. Scaiella. Fast and Accurate Annotation of Short Texts with Wikipedia Pages. *IEEE Software*, 29(1):70–75, 2012.
- [7] J. R. Finkel, T. Grenager, and C. D. Manning. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, ACL 2005, University of Michigan, USA*, pages 363–370, 2005.
- [8] B. Hachey, W. Radford, J. Nothman, M. Honnibal, and J. R. Curran. Artificial Intelligence. *Artificial Intelligence*, 194(C):130–150, 2013.
- [9] J. Hoffart, S. Seufert, D. B. Nguyen, M. Theobald, and G. Weikum. KORE: Keyphrase Overlap Relatedness for Entity Disambiguation. In *21st ACM International Conference on Information and Knowledge Management, CIKM 2012, Hawaii, USA*, pages 545–554, 2012.
- [10] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum. YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence*, 194:28–61, 2013.
- [11] J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust Disambiguation of Named Entities in Text. In *Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, Edinburgh, Scotland*, pages 782–792, 2011.
- [12] L. Jiang, J. Wang, P. Luo, N. An, and W. Min. Towards Alias Detection Without String Similarity: An Active Learning based Approach. In *35th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR 2012, Portland, USA*, pages 1155–1156, 2012.
- [13] J. Justeson and S. Katz. Technical Terminology: Some Linguistic Properties and an Algorithm for Identification in Text. *Natural Language Engineering* 1:9–27, 1995.
- [14] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective Annotation of Wikipedia Entities in Web Text. In *15th ACM SIGKDD international conference on knowledge discovery and data mining, KDD 2009, Paris, France*, pages 457–466, 2009.
- [15] Y. Li, C. Wang, F. Han, J. Han, D. Roth, and X. Yan. Mining evidences for named entity disambiguation. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD 2013, Chicago, IL, USA*, pages 1070–1078, 2013.
- [16] T. Lin, Mausam, and O. Etzioni. No noun phrase left behind: detecting and typing unlinkable entities. In *2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, Jeju Island, Korea*, pages 893–903, 2012.
- [17] R. Mihalcea and A. Csomai. Wikify! Linking Documents to Encyclopedic Knowledge. In *16th ACM Conference on Information and Knowledge Management, CIKM 2007, Lisbon, Portugal*, pages 233–242, 2007.
- [18] R. Mihalcea and P. Tarau. TextRank: Bringing Order into Texts. In *Conference on Empirical Methods in Natural Language Processing, EMNLP 2004, Barcelona, Spain*, pages 404–411, 2004.
- [19] D. Milne and I. H. Witten. An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links. In *AAAI 2008 Workshop on Wikipedia and Artificial Intelligence (WIKIAI 2008)*, Chicago, IL, 2008.
- [20] D. Milne and I. H. Witten. Learning to Link with Wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Mining, CIKM 2008, Napa Valley, United States*, pages 509–518, 2008.
- [21] N. Nakashole, T. Tylenda, and G. Weikum. Fine-grained Semantic Typing of Emerging Entities. In *51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, Sofia, Bulgaria*, pages 1488–1497, 2013.
- [22] R. Parker, D. Graff, J. Kong, K. Chen, and K. Maeda. English Gigaword Fifth Edition. Technical report.
- [23] L.-A. Ratinov, D. Roth, D. Downey, and M. Anderson. Local and Global Algorithms for Disambiguation to Wikipedia. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, ACL-HLT 2011, Portland, Oregon, USA*, pages 1375–1384, 2011.
- [24] B. Taneva and G. Weikum. Gem-based Entity-Knowledge Maintenance. In *ACM International Conference on Information and Knowledge Management, CIKM 2013, San Francisco, CA, USA*, 2013.
- [25] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning. KEA: practical automatic keyphrase extraction. In *4th ACM conference on Digital libraries, DL 99, Berkeley, CA, USA*, pages 254–255, 1999.