

# The Wisdom of Minority: Discovering and Targeting the Right Group of Workers for Crowdsourcing

Hongwei Li<sup>\*</sup>  
Department of Statistics,  
UC Berkeley  
hwli@stat.berkeley.edu

Bo Zhao  
Microsoft Research,  
Mountain View, CA  
bozha@microsoft.com

Ariel Fuxman  
Microsoft Research,  
Mountain View, CA  
arielf@microsoft.com

## ABSTRACT

Worker reliability is a longstanding issue in crowdsourcing, and the automatic discovery of high quality workers is an important practical problem. Most previous work on this problem mainly focuses on estimating the quality of each individual worker jointly with the true answer of each task. However, in practice, for some tasks, worker quality could be associated with some explicit characteristics of the worker, such as education level, major and age. So the following question arises: how do we automatically discover related worker attributes for a given task, and further utilize the findings to improve data quality? In this paper, we propose a general crowd targeting framework that can automatically discover, for a given task, if any group of workers based on their attributes have higher quality on average; and target such groups, if they exist, for future work on the same task. Our crowd targeting framework is complementary to traditional worker quality estimation approaches. Furthermore, an advantage of our framework is that it is more budget efficient because we are able to target potentially good workers before they actually do the task. Experiments on real datasets show that the accuracy of final prediction can be improved significantly for the same budget (or even less budget in some cases). Our framework can be applied to many real word tasks and can be easily integrated in current crowdsourcing platforms.

## Categories and Subject Descriptors

H.1.1 [Information Systems]: Models and principles

## General Terms

Frameworks, Algorithms, Experimentation, Human Factors

## Keywords

Crowdsourcing; Crowd Targeting; Worker Quality

<sup>\*</sup>This work is done during an internship of the first author in Microsoft Research, Silicon Valley.

## 1. INTRODUCTION

Crowdsourcing systems are widely used today in both industry and academia for collecting human-labeled data in a fast and inexpensive way. By having multiple workers perform the same task and aggregating their responses, one could get results whose quality is on par with responses from experts, as confirmed in some early studies [21, 22, 25]. The notion of “wisdom of crowds” has proved to be powerful in many scenarios over the years, making crowdsourcing ever more popular.

For aggregating the responses from multiple workers, the simplest approach would be treating the response from each worker equally and taking the most frequent one. This majority voting approach can be reasonably effective but it is often sub-optimal. Intuitively, some workers are better than others, and those higher quality workers are not necessarily the majority: as philosopher Soren Kierkegaard once said, “Truth always rests with the minority, and the minority is always stronger than the majority, because the minority is generally formed by those who really have an opinion”. Soren’s claim might be too strong, but it is true in many cases.

Therefore, how to find the “minority” in the crowd that truly has wisdom is a very important problem. There are quite a few previous work related to this problem [2, 10, 15, 18, 27]. The general approach is that the quality of each worker and the true answer of each task are modeled as latent variables that depend on each other, and the optimal values of these variables can be jointly estimated by optimizing certain objective functions. It is shown these methods can almost always outperform simple majority voting for deciding the final output.

In practice, previous worker quality estimation approaches still have some limitations. First, they mainly utilize signals from the distribution of workers’ responses, but in practice, there could be many other factors that are related to worker quality. Among those factors, various characteristics of the worker, such as demographics information and educational background, can be very significant in some tasks, as shown in some recent empirical studies [6, 11, 12]. Intuitively, for a given task, if some characteristics of workers are indeed related to their quality, they should be considered in worker quality estimation. On the other hand, it is also possible that none of the available worker attributes are related to quality in some tasks, and in such cases, they should not play a role in quality estimation.

The second limitation of previous approaches is that they can only estimate the quality of each worker after she per-

forms the task, which means it is difficult to improve budget efficiency for future tasks of the same kind. There are some ways to handle this, such as identifying spammers and refusing to pay them, and even banning them from doing the task in the future. However, for identified high quality workers, it could be difficult to reach them and have them do the task when needed, since they may not be available or willing to do the task with the same price.

In contrast, if we are able to discover that certain groups of workers, based on their characteristics, have on average better quality on a task, we can easily target getting new workers from the crowd that are in the same groups and the responses from the targeted worker groups will have higher quality in expectation than responses from the entire crowd.

In fact, most crowdsourcing platforms already provide some quality control mechanisms such as qualification test that allows task owners to select workers that meet certain criteria. For example, task owners may have some prior knowledge on which groups of workers might perform better, or use some sample questions with known ground truth to test workers. The only problem is that it is not guaranteed the qualification tests designed based on task owners' prior knowledge can effectively select high quality workers due to many potential issues, such as that the better groups could be different for different tasks, the designated criteria might be too strict or too loose, etc. Therefore, it will be more effective if the selection criteria can be automatically learned from the actual data.

To solve this problem, in this paper, we proposed a general crowd targeting framework, that can automatically discover if any specific groups of workers based on their characteristics have higher quality in average on a given task, and target on these groups, if they exist, for future work of the same task to improve data quality and budget efficiency. The general idea is to send a small part of the task to the entire crowd at the beginning, and learn the selection criteria from the data and use that criteria to target workers for the remaining and future work. Since the targeting criteria is automatically learned from the data, its effectiveness is more stable. Notice that the definition of worker characteristics can be very general, e.g., workers' available profile information, or their responses to any questions we put in the task.

To the best of our knowledge, we are the first to propose such crowd targeting framework. Our experiments on real world tasks and crowdsourcing systems show our method can significantly improve the accuracy of final output with the same or even less budget. The automatic worker group discovery method requires very few data with ground truth, and can also perform well when there is no ground truth at all. In addition, this framework can be easily implemented on top of most of the current crowdsourcing platforms by utilizing their existing quality control mechanisms.

In the rest of this paper, Section 2 introduces the crowd targeting framework. Section 3 presented the notation and setup for the crowdsourcing problem, and derives a new worker effect measure which can reflect how good a worker is. Then, Section 4 focuses on the details of two algorithms for implementing the crowd-targeting framework. Section 5 presents two experiments we have designed to verify the framework, and Section 6 discusses the most related work to this paper.

## 2. CROWD TARGETING FRAMEWORK

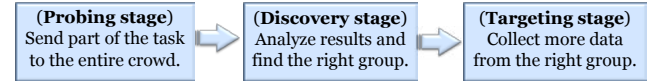


Figure 1: The crowd targeting framework.

In this section, we describe our crowd targeting framework in details. Figure 1 illustrates the three main stages of the framework: probing stage, discovery stage and targeting stage.

### 2.1 Probing stage

At this stage, we distribute a part of the task to the entire crowd population and allow everyone to perform it. This way, we get a unbiased sample of workers for the next stage.

We also gather characteristics of workers at this stage. In some crowdsourcing platforms, some basic information is available in workers' profiles, such as gender and location, which we can directly get. Otherwise, we can do the same as in [11, 12] and simply add survey questions as part of the task. We can require workers to answer these survey questions by utilizing the screening mechanism provided by most crowdsourcing platforms or some other techniques like Javascripts.

The notion of worker characteristics is very general. The default questions include demographics (major, education level, age, language, etc.) or some questions that can capture personality traits [11, 12]. Task owners can customize the questionnaire, and choose any questions that they consider relevant. (It is their responsibility to ensure that the questions do not affect the worker's privacy and that they comply with antidiscrimination laws.)

Figure 2 illustrates the design for collecting worker information that we used in our experiments. Notice that we assure the workers we protect their privacy and make clear that we will not identify any individual worker based on the information.

[== Step 1. Simple demographic information ==] [== Step 2. Knowledge questions ==]

**Instructions:** this survey is intended to be used in scientific research on crowdsourcing only. There will be two steps: (1) In step 1, you will input some demographic information. (2) In step 2, you will answer some comprehensive knowledge questions.

We will not identify anyone from the answers they provided. Therefore, please help us by trying your best to answer the questions. You can also test your comprehensive knowledge by finishing this test, and we will pay you after you finish all the questions to your best knowledge.

**You input is very important for us on scientific purpose. Thank you.**

1. What is your age?
2. What is your gender with which you identify?  
☐ Male  
☐ Female
3. Where are you from? ( choose the best option below to describe where is the place that you live in most of the time in your past)

Figure 2: A screen shot of part of the design for collecting worker information.

### 2.2 Discovery stage

After collecting labeled data and worker characteristics at the probing stage, we then focus on discovering if there exist any groups of workers from the unbiased sample of the entire population who are performing significantly better than other workers.

Quality is the most important criteria for worker group discovery: the target groups have to be better than other workers based on some measures. There are some standard measures that can be used to judge how good or useful a worker is, and in our framework we propose a new measure that can perform better in practice. Further discussions on measures are in Section 3.

Availability or the population ratio of the target groups is another important criteria for discovery. We cannot target at groups which are so rare in the crowd such that we have low chance to immediately get new workers from these groups for future tasks. This is even more important for tasks that are preferred to be finished within a short period of time. We are utilizing a population threshold in the group discovery algorithms, which can be tuned by task owners based on the urgency of their tasks.

### 2.3 Targeting stage

At this stage, we distribute the remainder of the task and future tasks of the same type only to the discovered groups output by the previous stage. We can utilize the quality control mechanisms provided by most crowdsourcing platforms to enforce the group selection.

The targeting can improve data quality and budget efficiency. With a given budget, we can get the same number of workers from groups that are more likely to provide higher quality responses. This is usually equivalent to that given a certain quality goal, we are able to hire less workers by targeting at better worker groups. In cases when the method works really well, we can even get higher quality results with less budget.

## 3. WORKER EFFECTS

### 3.1 Setup of the crowdsourcing scenario

There are many possible formats of crowdsourcing tasks, without loss of generality, in this paper we will focus on tasks that are in the format of multiple choice problems, and our framework can be easily generalized to other types of tasks where the worker reliability can be quantitatively measured.

In the multiple choice scenario, given a question and several possible answers, the task for each worker is to choose the best answer. This is a very common scenario and a lot of labeling tasks are in such format. Assuming there are  $M$  workers,  $N$  questions and each question has  $L$  options. Throughout this paper, we index workers by  $i$  and questions by  $j$ . For simplicity, we define  $[n] \doteq \{1, 2, \dots, n\}$ ,  $\forall n \in \mathbb{N}$ , so the pool of workers is denoted by  $[M]$ , the pool of questions  $[N]$ , and the pool of options  $[L]$ . Let  $Z$  be the label matrix where  $Z_{ij}$  denotes the answer worker  $i$  provides for question  $j$ , then  $Z_{ij} \in [L] \cup \{0\}$ , where 0 represents the answer is missing (worker  $i$  does not perform task  $j$ ). Let  $y_j \in [L]$  denote the true answer for task  $j$ .

We use a succinct model [10, 18] to model the relation between worker reliability and true answers by assuming that worker  $i$  is characterized by accuracy  $w_i \in [0, 1]$ , i.e., how often her answer is correct. And when she makes mistakes, her answer is uniformly distributed among all the wrong answers. Formally,

$$\mathbb{P}(Z_{ij} = k | y_j = k) = w_i, \quad \forall k \in [L] \quad (1)$$

$$\mathbb{P}(Z_{ij} = l | y_j = k) = \frac{1 - w_i}{L - 1}, \quad \forall l, k \in [L], l \neq k. \quad (2)$$

The posterior probability for the  $k$ -th answer to be true in task  $j$  given the collected answers, i.e.,  $\mathbb{P}(y_j = k | Z, \{w_i\}_{i=1}^M)$ , and the accuracy of each worker  $i$ , i.e.,  $w_i$ , are unknown at the beginning, but they can be estimated iteratively by the EM algorithm [3, 18]. In literature, this method is referred to as the maximum likelihood method on the Succinct Dawid-Skene model or one-coin model [10, 18, 13, 14], which is a special case of the full Dawid-Skene model [2]. We use this method due to its simplicity and effectiveness in many scenarios, and our crowd targeting framework can be easily generalized when other methods are applied for estimating worker accuracy and predicting true answers.

### 3.2 Worker quality measures as effects

Our approach is based on characterizing the relationship between the characteristics of the workers and the quality of the results that they produce (which we call *effects*). These measures should effectively reflect how good or how useful a worker's answers are for the final prediction of true answers. So intuitively, the measure should be a function of the worker's accuracy. Formally the effect of the  $i$ -th worker is  $\tau_i = f(w_i)$ .

The simplest measure would be directly using the accuracy, or its logit form as follows,

**Accuracy:**  $f(w_i) = w_i \in [0, 1]$ .

**Logit Accuracy:**  $f(w_i) = \ln \frac{w_i}{1 - w_i} \in (-\infty, +\infty)$ .

The motivation behind the above two measures is that workers with higher accuracy will provide more accurate answers, which should improve the final prediction in principle.

If we use simple methods such as majority voting for the final prediction, the only hope we can have is to get more workers with high accuracy. However, in practice EM methods described can almost always outperform majority voting in final prediction, which motivates us to explore if there are other measures for a worker's contribution that suits better with the EM methods.

Based on the analysis of the EM algorithm on Dawid-Skene models [13, 14], we can know that if we can correctly estimate each worker's accuracy  $w_i$ , then we will give higher positive weights to answers provided by more accurate workers; on the other hand, for workers who have accuracy worse than random guessing, we can effectively give their answers negative weights, which can help filter out potential wrong answers. In another word, a less accurate worker, if her accuracy can be effectively estimated, could still make positive contribution for the final prediction.

Then the question is, what should be the measure that can capture a worker's actual contribution or usefulness for predicting true answers using the EM method.

The EM algorithm tries to maximize the expected complete data log-likelihood given unknown parameters  $\Theta = \{\{w_i\}_{i=1}^M\}$  and current estimated posterior  $\rho_{jk} = \mathbb{P}(y_j = k | Z, \Theta^{\text{old}})$ :

$$\begin{aligned} & Q(\Theta, \Theta^{\text{old}}) \\ &= \mathbb{E}_{Y|Z, \Theta^{\text{old}}} [\ln \mathbb{P}(Y, Z | \Theta)] \\ &= \sum_{i=1}^M \sum_{j=1}^N \sum_{k=1}^L \rho_{jk} \left[ I(Z_{ij} = k) \ln w_i + I(Z_{ij} \notin \{0, k\}) \ln \frac{1 - w_i}{L - 1} \right] \\ & \quad + \text{constant}, \end{aligned}$$

where *constant* is independent of  $w_i$ . So the data likelihood contributed by worker  $i$  is essentially:

$$C_i = S_i \ln w_i + (n_i - S_i) \ln \frac{1 - w_i}{L - 1}, \quad (3)$$

where  $S_i = \sum_{j=1}^N \sum_{k=1}^L \mathbb{P}(y_j = k | Z, \Theta^{\text{old}}) I(Z_{ij} = k)$ , i.e., the expected number of questions correctly answered by worker  $i$ ; and  $n_i = \sum_{j=1}^N I(Z_{ij} \neq 0)$ , i.e., the total number of questions worker  $i$  has answered.

The optimal  $w_i$  that maximizes  $C_i$  is:  $\hat{w}_i = S_i/n_i$ , so the maximum value of  $C_i$  can be written as:

$$\begin{aligned} \hat{C}_i &= n_i \left( \hat{w}_i \ln \hat{w}_i + (1 - \hat{w}_i) \ln \frac{1 - \hat{w}_i}{L - 1} \right) \\ &= -n_i \cdot \text{Entropy} \left( \left\{ \hat{w}_i, \frac{1 - \hat{w}_i}{L - 1}, \dots, \frac{1 - \hat{w}_i}{L - 1} \right\} \right). \end{aligned} \quad (4)$$

Notice that if  $\hat{w}_i$  can freely take values from  $[0, 1]$ , then  $\hat{C}_i$  is maximized when  $\hat{w}_i = 1$ , i.e., worker  $i$  is a perfect worker; and  $\hat{C}_i$  is minimized when  $\hat{w}_i = 1/L$ , which is equivalent to the accuracy of a worker who randomly guesses the answer. This reveals the fact that if EM is used for finding the true answer, a random guesser is the most helpless worker comparing with workers with higher or even lower accuracy. Note that  $\ln L$  is the entropy of the answer distribution of a random guesser. Therefore, (4) is equivalent to the *information gain* from the answers of worker  $i$  compared to a random guesser, upto a constant  $n_i \ln L$ .

By normalizing based on  $n_i$  to get a worker's average contribution on each task she takes, and adding a constant  $\ln L$  to make the value in a proper range, we formally define the third measure **Information Gain**:

$$f(w_i) = \ln L + w_i \ln w_i + (1 - w_i) \ln \frac{1 - w_i}{L - 1}, \quad (5)$$

where  $f(w_i) \in [0, \ln L]$ . As we have explained, this measure is well justified by information theory and has strong connection with the EM algorithm.

## 4. WORKER GROUP DISCOVERY ALGORITHMS

In this section, we discuss the core algorithms in the discovery stage of the crowd targeting framework.

As we have mentioned in Section 2, the problem is to automatically find groups of workers that are more helpful on the task, which is measured by the *effects* proposed in the previous section. To compute effects, we need to estimate worker accuracy  $w_i$  from the data we gathered in the probing stage. We could either leverage some ground truth data to directly compute it, or estimate it using the EM algorithm if there is no ground truth.

In general, the goal is to partition the entire crowd into two sets of characteristic-based worker groups and one set of groups should be better at the task than the other set. There are two natural ways to do it, (1) Bottom-up approach: the entire crowd can be seen as having already been partitioned into small subgroups by different levels of worker characteristics. Then what needs to be done is to merge the small subgroups that contain good workers into a larger target group, and therefore we call it the bottom-up approach. (2) Top-down approach: the entire crowd can be looked as a whole at the beginning, and we need to gradually pick different characteristics to split the crowd into subgroups, choose

the best subgroup and try to split further. In the end, we can stop and get a target group that contains good workers.

Based on the ideas above, we designed two algorithms: the Bottom-up Discovery Algorithm and the Top-down Discovery Algorithm. The Bottom-up Discovery Algorithm learns a model that, given the characteristics of the worker, decides whether the worker is eligible for the given task. The advantage of this algorithm is that potentially all characteristics of the worker can be used to make the eligibility determination. On the other hand, it has two drawbacks: the criteria to determine worker eligibility is hard to interpret, and it cannot capture the availability of judges with each characteristic. These problems are addressed with the Top-down Discovery Algorithm, which determines a priori an optimal subset of characteristics that must be satisfied by the eligible workers. The resulting tradeoff, however, is that some possibly valuable judge characteristics will not be used to make eligibility decisions if they are not chosen by the algorithm. As such, the decision on which algorithm to choose ultimately depends on the requirements of the crowdsourcing task.

### 4.1 Bottom-up Discovery Algorithm

Formally, let us assume that there are  $M$  workers and their worker effects are  $\{\tau_1, \dots, \tau_M\}$ , and there is a feature pool which contains  $t$  features (i.e., worker characteristics), denoted by  $\mathbb{F} = \{F_1, \dots, F_t\}$ . For example,  $\mathbb{F}$  could be {Education, Major, Gender}. The feature vector of worker  $i$  is denoted as  $X_i = (X_i^{(1)}, \dots, X_i^{(t)})$ , for example it could be {College, Science, Female}.

For analyzing the association between the worker features and the worker effect, intuitively we can apply the fixed effect model [16]:

$$\tau_i \sim \beta_0 + \beta_1 X_i^{(1)} + \dots + \beta_t X_i^{(t)} + \epsilon, \quad \forall i \in [M], \quad (6)$$

where  $\beta = (\beta_0, \beta_1, \dots, \beta_t)$  are coefficients and  $\epsilon \sim \mathcal{N}(0, \sigma^2)$ , i.e., Gaussian noise with mean 0. Note that  $X_i$  could be categorical variables, and coded as a vector via dummy coding [16]. In that case  $\beta_i$  will be a vector and each element of it will be an fixed effect coefficient of a level of  $F_i$ .

---

#### Algorithm 1 Bottom-up Discovery Algorithm

---

**Input:** Feature pool:  $\mathbb{F} = \{F_1, \dots, F_t\}$ ;  $M$  workers with worker effect  $\{\tau_1, \dots, \tau_M\}$  and their feature vectors  $\{X_1, \dots, X_M\}$  where  $X_i = (X_i^{(1)}, \dots, X_i^{(t)})$ ; Accessibility parameter:  $\lambda \in (0, 1)$ ;

- 1: Fit the fixed effect model (6) to learn the model parameters  $(\hat{\beta}_0, \hat{\beta}_1, \dots, \hat{\beta}_t)$ .
- 2: Obtain the fitted value for each workers by  $\hat{\tau}_i \leftarrow \hat{\beta}_0 + \hat{\beta}_1 X_i^{(1)} + \dots + \hat{\beta}_t X_i^{(t)}, \quad \forall i \in [M]$
- 3: Rank fitted effect  $\{\hat{\tau}_i\}$  in descending order:  $\hat{\tau}_{\pi(1)} \geq \hat{\tau}_{\pi(2)} \geq \dots \geq \hat{\tau}_{\pi(M)}$ , where  $\pi$  is a permutation of worker index  $\{1, 2, \dots, M\}$ .
- 4: The threshold  $\tau_0 \leftarrow \min \{\hat{\tau}_{\pi(i)} | i \in [M] \text{ and } \frac{i}{M} \geq \lambda\}$ .

**Output:** The learned model parameter  $(\hat{\beta}_0, \hat{\beta}_1, \dots, \hat{\beta}_t)$  and the effect threshold  $\tau_0$ .

---

The Bottom-up Discovery Algorithm is described in Algorithm 1. The fixed effect model as in (6) is easy to fit by multiple regression with  $L_2$  loss after dummy coding, and many packages are available in public software such as R<sup>1</sup> and Mat-

<sup>1</sup>[http://www.ats.ucla.edu/stat/r/modules/dummy\\_vars.htm](http://www.ats.ucla.edu/stat/r/modules/dummy_vars.htm)

lab<sup>2</sup>. After fitting this fixed effect model to data gathered during the probing stage, we will learn the coefficients  $\hat{\beta}$  and a threshold  $\tau_0$  on predict effect for a worker to be in the target group. For a worker with features  $(X_1, X_2, \dots, X_t)$  in the targeting stage, we evaluate the effect of the subgroup he/she belongs to with  $\hat{\tau} \leftarrow \hat{\beta}_0 + \sum_{k=1}^t \hat{\beta}_k X_k$ . The worker will be qualified for the task if  $\hat{\tau} > \tau_0$ .

Note that the accessibility parameter  $\lambda$  reflects roughly how many workers in the crowd satisfy the criterion — with predicted effect greater than  $\tau_0$ . It thus controls how accessible the target group will be. For example, if  $\lambda$  is close to 0, then  $\tau_0$  will be relatively large, thus there might be only very few qualifying workers. In contrast, if  $\lambda$  is close to 1,  $\tau_0$  will be small, and most workers will qualify.

## 4.2 Top-down Discovery Algorithm

The Bottom-up Discovery Algorithm is directly solving the problem of predicting effect values for each worker group. However, it does not consider whether each feature is significant enough to affect worker reliability, so the fitted model may not reveal the true association between worker features and effects. For example, if in the probing stage, there is one attribute that only very few workers have, e.g., {Education=PhD}, the bottom up approach will still try to connect such feature to the effect, which may not be stable. Therefore, we want to find a method that can generate more stable and interpretable results. The Top-down Discovery Algorithm described in this section is one of such approaches.

The general idea is we should choose subgroups based on features that are significantly associated with worker effects. ANOVA (ANalysis Of VAriance) [23] based on the fixed effect model (6) is an appropriate tool for testing feature significance.

One remaining issue is that when there are multiple features in  $\mathbb{F}$ , and each feature has multiple levels (multiple possible values), the number of workers that have the same features might be too small, especially when  $M$  is already very small (typically 100 or 200 in real crowd-sourcing settings). The multiple-way ANOVA will be unstable in such case. More importantly, for achieving interpretability and reducing the risks of over-fitting, we also hope that output worker subgroups are not too many.

Based on the intuitions above, we propose to do one-way ANOVA sequentially on each feature and obtain the p-value  $p_k$  for  $F_k$  based on the fixed effect model:

$$\tau_i \sim \beta_0 + \beta_1 X_i^{(k)} + \epsilon, \quad \forall i \in [M], F_k \in \mathbb{F}. \quad (7)$$

Since not every feature will be strongly associated with the worker effect, we have to use a significance threshold  $p_{sig}$  to control the significance of each test, i.e., p-value. It is common to choose 0.10 or 0.05 as the significance threshold, and we use  $p_{sig} = 0.10$  as default in our method.

Similar to the Bottom-up Discovery Algorithm, we need to have a accessibility parameter  $\lambda$  to ensure that the size of the target group is more than  $100\lambda\%$  of the crowd.

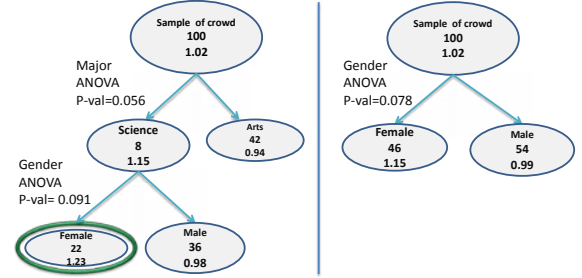
Algorithm 2 is the detailed description of the Top-down Discovery Algorithm. The general steps include: (1) sequentially testing if features are significantly associated with worker effects, (2) splitting the crowd using the most significant features, and (3) picking the subgroup with highest

### Algorithm 2 Top-down Discovery Algorithm

**Input:** Feature pool:  $\mathbb{F} = \{F_1, \dots, F_t\}$ ;  $M$  workers with effect  $\{\tau_1, \dots, \tau_M\}$  and feature vectors  $\{X_1, \dots, X_M\}$  where  $X_i = (X_i^{(1)}, \dots, X_i^{(t)})$ ; Accessibility parameter:  $\lambda \in (0, 1)$ ; Significant level:  $p_{sig}$  with default value 0.1.

- 1: **Initialization:** Current feature pool  $\mathbb{F}_{current} \leftarrow \mathbb{F}$  and current crowd  $\mathbb{S}_{current} \leftarrow [M]$ ;  $\mathbb{F}_{out} \leftarrow \emptyset$  and  $\mathbb{L}_{out} \leftarrow \emptyset$ .
- 2: **repeat**
- 3:   **for** feature  $F_k$  in  $\mathbb{F}_{current}$  **do**
- 4:     Computing one-way ANOVA on feature  $F_k$  with  $\tau_i \sim \beta_0 + \beta_1 X_i^{(k)} + \epsilon$ , and obtain the p-value  $p_k$ .
- 5:   **end for**
- 6:    $k^* \leftarrow \arg \min \{p_k | F_k \in \mathbb{F}_{current}\}$ .
- 7:   Suppose  $F_{k^*}$  has  $n$  levels  $\{\mathcal{L}_1^{(k^*)}, \dots, \mathcal{L}_n^{(k^*)}\}$ , which partitions the  $\mathbb{S}_{current}$  to  $\{S_1, \dots, S_n\}$ . Then  $\forall l \in [n]$ , compute the average effect:  $E_l \leftarrow \frac{1}{|S_l|} \sum_{i \in S_l} \tau_i$ .
- 8:    $l^* \leftarrow \arg \max_{l \in [n]} \{E_l\}$ ,  $\mathbb{S}_{current} \leftarrow \{i | X_i^{(k^*)} = \mathcal{L}_{l^*}^{(k^*)}\}$ .
- 9:   **if**  $\frac{|\mathbb{S}_{current}|}{M} > \lambda$  or  $p_{k^*} \leq p_{sig}$  **then**
- 10:      $\mathbb{F}_{out} \leftarrow \mathbb{F}_{out} \cup \{F_{k^*}\}$  and  $\mathbb{L}_{out} \leftarrow \mathbb{L}_{out} \cup \{\{\mathcal{L}_{l^*}^{(k^*)}\}\}$ .
- 11:      $\mathbb{F}_{current} \leftarrow \mathbb{F}_{current} \setminus \{F_{k^*}\}$ .
- 12:   **else**
- 13:     Stop and return  $\mathbb{F}_{out}$  and  $\mathbb{L}_{out}$ .
- 14:   **end if**
- 15: **until**  $\mathbb{F}_{current} = \emptyset$

**Output:** Target feature pool  $\mathbb{F}_{out}$  and feature levels  $\mathbb{L}_{out}$ .



**Figure 3: Illustration of group discovery algorithm by a running example:** suppose we have 100 workers and then only have features “Gender” and “Major”. Each circle represents a group of workers, the integer in each circle is the number of workers in the group and the real value is the average worker effect. Since the feature “Major” has smaller p-value than “Gender” (more significant), the algorithm splits the crowd on “Major” and chooses the worker group with the highest average effect: “Science”. Then it continues with the rest of the features. In the end, the algorithm outputs “Major = Science” and “Gender = Female” as the target group.

average effect, and go to (1) to check if the group should be further partitioned.

As an illustration, Figure 3 shows a running example of this algorithm. Suppose we have hired 100 workers in the probing stage, and the feature pool we choose to run the Top-down Discovery Algorithm is  $\mathbb{F} = \{\text{Major}, \text{Gender}\}$ , and the feature “Major” has two levels {“Science”, “Arts”}. We choose the accessibility parameter  $\lambda = 0.20$  and the default

<sup>2</sup><http://www.mathworks.com/help/stats/linearmodel.fit.html>

significance threshold  $p_{sig} = 0.10$ . We first conduct one-way ANOVA test on both of the features, and find that “Major” has the lower p-value 0.056, which is significant. Then we choose the level “Science” to be the current target crowd. By testing feature “Gender” again, we find it is still significant, then we further split the target crowd and reach  $\{\text{Major} = \text{Science}, \text{Gender} = \text{Female}\}$ . If the size of this target group is larger than  $100 \cdot \lambda = 20$ , then we directly output the group. Otherwise, we just output the parent group  $\{\text{Major} = \text{Science}\}$ .

#### 4.2.1 Feature merging step

One practical issue in the Top-down Discovery Algorithm is that for some characteristics of workers, such as major or nationality, there might be too many levels. Hence, the number of workers in each level could be too small, which could be a serious issue for the Top-down Discovery Algorithm since the algorithm may potentially partition the crowd into too small sets, which will easily violate the accessibility criterion (controlled by parameter  $\lambda$  in Algorithm 2).

To solve this problem, in this section we propose an algorithm called the feature merging algorithm, which tries to merge multilevel features into bi-level for maintaining accessibility and high reliability of workers.

---

#### Algorithm 3 Feature Merging

---

**Input:** Feature  $F$  which has  $K$  levels  $\{\mathcal{L}_1, \mathcal{L}_2, \dots, \mathcal{L}_K\}$ ;  $M$  hired workers with worker effect  $\{\tau_1, \dots, \tau_M\}$  and their feature values  $\{X_1, \dots, X_M\} \subset F$ ;

```

1: if  $K \leq 2$  then
2:    $F^* \leftarrow F$  and no need to merge its levels.
3: else
4:   for  $k$  from 1 to  $K$  do
5:      $S_k \leftarrow \{i \in [M] \mid X_i = \mathcal{L}_k\}$ 
6:      $E_k \leftarrow \frac{1}{|S_k|} \sum_{i \in S_k} \tau_i$ 
7:   end for
8:   Ranking  $\{E_k\}$  to obtain the order  $\pi$  such that  $E_{\pi(1)} \geq E_{\pi(2)} \geq \dots \geq E_{\pi(K)}$ .
9:    $k^* \leftarrow \max \left\{ k \mid \sum_{1 \leq i \leq k-1} S_{\pi(i)} < \frac{M}{2}, \sum_{1 \leq i \leq k} S_{\pi(i)} \geq \frac{M}{2} \right\}$ .
10:   $k^* \leftarrow \min \{k^*, K - 1\}$  for avoiding combine all levels.
11:   $\mathcal{L}'_1 \leftarrow \mathcal{L}_{\pi(1)} \cup \dots \cup \mathcal{L}_{\pi(k^*)}$ ,  $\mathcal{L}'_2 \leftarrow F \setminus \mathcal{L}'_1$ , and  $F^* \leftarrow \{\mathcal{L}'_1, \mathcal{L}'_2\}$ .
12: end if

```

**Output:** Feature  $F^*$  which has at most two levels  $\{\mathcal{L}'_1, \mathcal{L}'_2\}$

---

Algorithm 3 presents the details for merging multi-level features. The major idea of this algorithm is that: (1) Partitioning the  $M$  workers from the probing stage into subgroups by feature  $F$ . (2) Ranking the mean effect of subgroups in descending order. (3) Combining the levels in the sorted order into one target level until the size of the combined group is no less than half, i.e.,  $\frac{M}{2}$ .

Figure 4 illustrates the idea of Algorithm 3 with a running example. Suppose we have 100 workers in the probing stage and the feature “Major” has 4 levels {Art, Business, Science, Engineering}. We first compute the average worker effect of the workers in each major, then rank them from high to low. After combining major levels from the higher effect to the lower one until the number of workers in the group is above 50 (i.e., half of the crowd), “Science/Engineering” is a new level, and “Business/Arts” is another new level.

	Art.	Bus.	Sci.	Eng.
(1) Effect	0.9	1.0	1.2	1.1
Popul.	25	17	30	28
	Sci.	Eng.	Bus.	Art.
(2) Effect	1.2	1.1	1.0	0.9
Popul.	30	28	17	25
	Science/Engineering		Business/Arts	
(3) Effect	1.15		0.94	
Popul.	58		42	

**Figure 4: A running example of the feature merging algorithm:** (1) suppose we have 100 workers, and the feature “Major” contains 4 levels. “Popul.” represents the number of workers in each major. (2) Combining majors with higher average effect until the number of workers in the combined group is above 50 (half of the crowd). (3) “Science/Engineering” is a new level, and “Business/Arts” is another new level.

Note that in practice, we will apply feature merging first before applying the Top-down Discovery Algorithm.

## 5. EXPERIMENTS

### 5.1 Setup

We performed our experiments on Microsoft Universal Human Relevance (UHRS) system, which is the default crowdsourcing platform in Microsoft that handles numerous labeling tasks from various teams. Workers on UHRS are mainly from external vendor crowdsourcing companies, and task owners can choose vendors for their tasks. In our experiments, we used ClickWorker.com.

#### 5.1.1 Knowledge test data

In the first experiment, we would like to test on a domain where we think worker reliability is very likely to be related to worker demographics, and verify if the crowd-targeting framework is effective on such domain. This dataset consists of 75 knowledge based questions from allthetests.com with ground truth. All these questions have 4 options, and workers on the crowdsourcing platform are asked to answer each question by choosing one of the options based on the best of their knowledge. Topics among the questions cover science, math, common knowledge, sports, geography, U.S. history and politics and India, etc.

A typical example of knowledge test question could be as follows: (*Question*) *In what year was the Internet created?* (*Options*) *A. 1951; B. 1969; C. 1985; D. 1993.* Apparently, this question will be easier if the worker knows the answer, or has certain background knowledge or logical inference skills to exclude some obviously wrong options.

Normally, aggregating the answers from 10 to 20 workers for a question is good enough empirically [22]. However, we want to investigate how the performance of aggregation algorithms such as majority voting or EM increases w.r.t. the number of answers provided for each question. Therefore, we required each worker to finish all the questions. We distributed the tasks on UHRS and set the number of workers needed to 100, and the price was \$1.5 for finishing all the questions.

It is worth mentioning that on this data, where every question is labeled by all the workers, majority voting achieves accuracy 80%, and EM algorithm achieves 81.33%.

### 5.1.2 Disambiguation data

In the second experiment, we deployed a real world labeling task, which is needed for gathering training data for actual product development. The task is that given a highlighted entity in a sentence, the workers should identify which Wikipedia page the entity actually refers to.

Figure 5 shows a typical example of this task. The word “runtime” has different meanings in different contexts. In this question, the correct answer is the second option.

1. The Microsoft .NET Framework 4 redistributable package installs the .NET Framework **runtime** and associated files that are required to run and develop applications to target the .NET Framework 4.
  - [http://en.wikipedia.org/wiki/Run-time\\_system](http://en.wikipedia.org/wiki/Run-time_system)
  - [http://en.wikipedia.org/wiki/Runtime\\_library](http://en.wikipedia.org/wiki/Runtime_library)
  - [http://en.wikipedia.org/wiki/Run\\_time\\_\(program\\_lifecycle\\_phase\)](http://en.wikipedia.org/wiki/Run_time_(program_lifecycle_phase))
  - [http://en.wikipedia.org/wiki/Run-Time\\_Infrastructure\\_\(simulation\)](http://en.wikipedia.org/wiki/Run-Time_Infrastructure_(simulation))

**Figure 5: A typical example of disambiguation questions. The word “runtime” has different meanings, and the task is to identify which Wikipedia page it actually refers to in this sentence. The correct answer is the second option.**

We collected 50 of such questions in the technology domain with ground truth available, and each question has 4 options. Then we randomly partitioned them into two even sets, which we call disambiguation data part 1 and part 2.

The two parts of the questions were distributed to the crowdsourcing platform in different time to simulate the probing stage and the targeting stage, since the participating workers will not be the same. In the experiments, we will try to discover good worker groups on one part and apply targeting on the other to test the effectiveness of the crowd targeting framework.

Crowd Statistics	Dataset 1	Dataset 2
Crowd size	144	133
Average accuracy	0.559	0.572
Majority voting acc.	84%	80%
EM accuracy on all	88%	84%

**Figure 6: A brief summary statistics of the disambiguation dataset. The performance of both MV and EM are based on all the labels in each dataset.**

Brief statistics on the disambiguation data is presented in Figure 6. Part one has 144 workers and part two has 133. We can compute the “true” accuracy of each worker using the ground truth: the average worker accuracy is 55.9% on part 1 and 57.2% on part 2, which is fairly low meaning the task is quite difficult. Majority voting and EM have reasonably good accuracy, but note that it is achieved by running against all the labels in each dataset.

### 5.1.3 Recognizing textual entailment data

This data is sampled from the well known public crowdsourcing dataset by Snow *et al.* [22]. For each question, a worker is presented with two sentences and a binary choice

of whether the second sentence can be inferred from the first. As a typical example given in [22], the sentence “Oil prices drop” would constitute a true entailment from the text “Crude Oil Prices Slump”, but a false entailment from “The government announced last week that it plans to raise oil prices”.

From the original RTE dataset, we randomly select 60 questions when the majority answered wrong or there is a tie. Meanwhile, we randomly choose 20 questions from the ones that the majority answer correctly. By randomly dividing these 80 questions into two even parts, we have RTE dataset part 1 and part 2. The two parts are posted on UHRS in different time and we asked workers to finish all the questions and payed them \$1.5.

## 5.2 Experimental results

In the experiments, the estimated worker accuracy is transformed into the worker effect based on Information Gain defined in (5). Worker characteristic information is collected before the worker answers any regular questions. We investigated which group will be more suitable for the task by Top-down Discovery Algorithm after merging the multi-level features to bi-level by Algorithm 3. We also apply the Bottom-up Discovery Algorithm to learn the fixed effect model and effect threshold  $\tau_0$  without merging the feature.

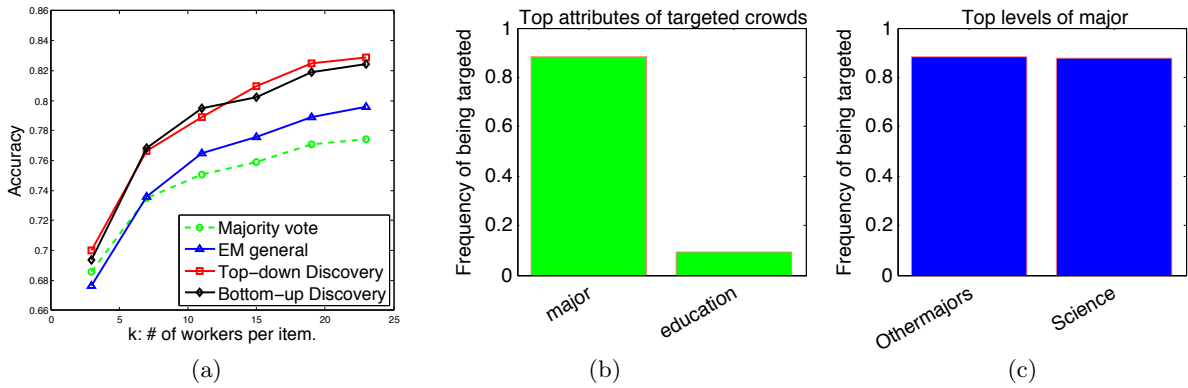
### 5.2.1 Results on knowledge test data

In this experiment, we randomly sample 5 out of 75 questions as the questions used in the probing stage. We estimate the worker accuracy by comparing the answers of these 5 questions from each worker against the ground truth, and then run worker group discovery to get the targeted worker groups. For the rest of the questions (excluding the 5 exam questions), we basically simulate the targeting stage by randomly sampling  $k$  workers from the targeted groups for each question, and then we run the EM algorithm on the answers provided by the selected workers. We call this *EM with targeting*.

As a comparison, we also run EM and MV on the entire crowd (without targeting at any specific group of workers) by random sampling  $k$  answers for each question from the entire data.

Figure 7(a) shows the comparison. In the results, we refer the “Top-down Discovery” as running EM with targeting crowd learned from Top-down Discovery Algorithm, and “Bottom-up Discovery” as from Bottom-up Discovery Algorithm. “EM general” is running EM without targeting any specific groups, and majority voting is also running on general crowd. We can see generally the performance of EM with targeting is dominating the results of EM and MV without targeting. This indicates that the data quality is significantly improved by our crowd-targeting framework since we compare exactly the same EM algorithm on the data collected from two different ways — one is only from the targeted group discovered in the probing stage, another is from the general crowd.

Another question we are also interested in is that what features (i.e., worker characteristics) are the most important ones that distinguish the “good” group of workers from the general crowd. To address this question, we plot the frequency of each feature selected by the Top-down Discovery Algorithm. We can get the frequency since we repeat all our



**Figure 7: Knowledge test data.** (a) Comparing majority voting, EM without targeting, and EM with Bottom-up Discovery Algorithm and Top-down Discovery Algorithm respectively. (b) Plotting the frequency of significant features output by Top-down Discovery Algorithm. (c) The frequency plot of top features in “Major”.

experiments for multiple times and the reported results are the ones averaged across multiple runs.

Figure 7 (b) and (c) visualize the frequencies of the features selected for identifying targeted groups. Apparently, “Major” is the most important feature, and “Other majors” and “Science” are the two top majors. Note that “Other major” represent all majors except “Science”, “Engineering”, “Literature” and “Arts”.

### 5.2.2 Results on Disambiguation data

In this experiment, we randomly divided the question set into two parts, which we called Disambiguation dataset part 1 and part 2 respectively as we have discribed in Section 5.1.2.

First, we conducted experiments on the same part of the disambiguation data. For each  $k$ , which is the number of labels (or workers) sampled for each question, we randomly sample 5 questions and used them as “exam” questions in the probing stage. We estimated the worker reliability  $w_i$  by comparing the sampled labels from worker  $i$  to the answers of the 5 questions which we know the ground truth. After that, we applied the crowd targeting framework by running Bottom-up Discovery Algorithm or Top-down Discovery Algorithm.

Figure 8(a) shows the comparison of different methods on the Disambiguation dataset 1. We get similar results on part 2, so we omit them due to limited space.

Note that “Subset labels” refers the subset of labels used in “EM general”. It is selected in this way: after sampling  $k$  workers from the general crowd for each question, we have a fixed label matrix  $Z$  which is used in “EM general”, then we subset the labels of  $Z$  corresponding to the workers in the targeted crowd from Top-down Discovery Algorithm. This will lead to a label matrix  $Z_{sub}$ . The curve corresponding to the results we run EM on  $Z_{sub}$ .

The “Crowd qualified” refers to the scenario where we use the 5 questions with ground truth as qualification tests and only allow workers who achieve a certain accuracy to perform the task. No worker characteristics are being used for targeting.

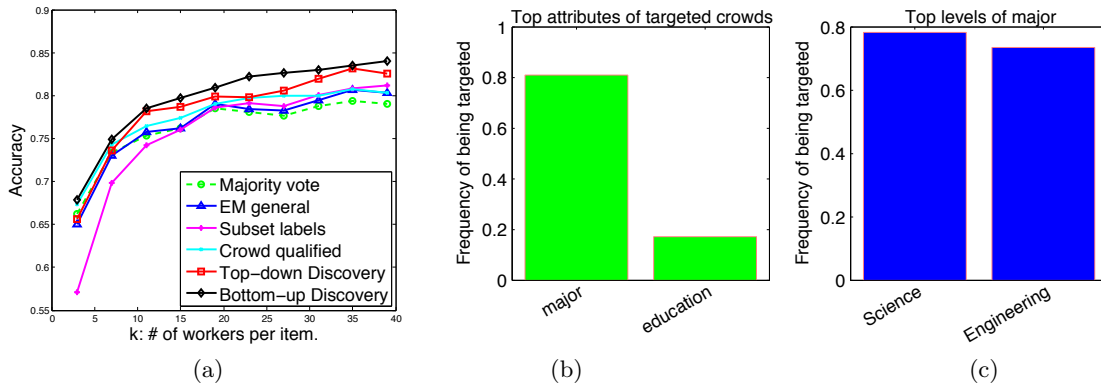
What we can observe from Figure 8(a) is that the performance of “EM general” is better than that of majority voting, but worse than that of “Crowd qualified”. And EM

on the targeted crowd from both Top-down Discovery and Bottom-up Discovery performs better than all the others. One interesting phenomenon we can see from the figure is about comparing “EM general” with “Subset labels”. Since both of them are running the same algorithm – EM, but the latter use only a subset data of the previous one. When  $k$  is small, the subset data will contains much less “useful” information than the data used by “EM general”, so the performance of EM on “Subset labels” is much less than “EM general”. However, when  $k$  increases to a certain level, using the subset data will even be better than using more data. This could be due to the Signal-Noise-Ratio is much higher in the data provided by targeted crowd than that by general crowd. This is the essential philosophy of “the wisdom of the minority”.

Figure 8(b)&(c) shows the frequency plot as explained in Figure 7. For the disambiguation dataset, the most important feature is major, and the target group concentrated on the workers who majored in “Science” and “Engineering”.

Since having seen promising results of utilizing crowd-targeting with the “Exam” questions and testing on the rest of the questions from the same dataset, we want to know if the similar results hold when we do crowd-targeting on one crowd and question set, then test on a different crowd and question set. Figure 9(a) shows the results when we randomly choose 5 questions which we know the ground truth from dataset part 1, conduct the crowd-targeting, and then test on the dataset part 2 which is collected form a different crowd and question pool. The EM algorithm with both Bottom-up Discovery and Top-down Discovery perform better than that with data collected from the general crowd. The frequency plot based on the output from Top-down Discovery Algorithm are very similar to Figure 8 (b)&(c). We omit them here for saving space.

Next, what should we do if there is no ground truth answers available? Can the crowd-targeting framework still be applied? In the next experiment, we randomly sample  $k$  workers for each question in Disambiguation dataset 1, then infer the true labels and the worker accuracy by EM algorithm without any ground truth labels. After obtaining the estimated worker accuracy, we apply the both Bottom-up Discovery Algorithm and Top-down Discovery Algorithm to the workers in dataset part 1. On the test stage, we also



**Figure 8: Disambiguation dataset.** (a) Comparing majority voting, EM without targeting, and EM with targeting. We randomly sample 5 questions as “Exam” questions to workers, and tested on the rest questions in this part of data. (b) The top two targeted features and their frequency of been targeted. (c) The top two levels of the feature “Major” and their frequencies.

randomly sampled  $k$  workers for each question in dataset part 2 from the target group, run EM algorithm, and then compare with running EM on the data sampled from general crowd. The results are shown in figure 9(b). What we can see from this two figures is basically the same with the previous results (targeting crowd with several ground truth labels). The only difference is that the performance of the EM algorithm drops slightly. The features selected by the Top-down Discovery Algorithm is almost the same as what is shown in Figure 8(b)&(c).

In the last experiments on this dataset, we compare the different effects — Information Gain, accuracy and logistic as discussed in Section 3. The results is shown in Figure 9(c). The Information Gain performs the best but not significantly better than the other two. This is because when a workers is better than random guessing, this three effect measures will be highly correlated.

### 5.2.3 Results on RTE data

We repeated the similar experiments on RTE dataset and obtained similar results — the performance of the EM algorithm on the data collected from the target group, discovered by both Top-down Discovery Algorithm and Bottom-up Discovery Algorithm, are better than the EM algorithm on the data collected without crowd targeting. Note that in RTE data  $L = 2$  and crowd size are smaller than the previous two datasets. We run worker group discovery algorithms on RTE dataset 1 without ground truth labels as the same as 9(b), and test on RTE dataset 2. The Top-down Discovery Algorithm performs better than Bottom-up Discovery Algorithm from the experimental results on the RTE data.

## 6. RELATED WORK

In recent years, many work have been done on improving data quality for crowdsourcing. A lot of them focused on the joint inference of true labels of items and worker reliability after data are collected [2, 25, 8, 18, 24, 10, 1, 15, 27]. These methods are generally post-processing methods, so they cannot be easily used during task assignment to improve data quality. Our proposed crowd targeting framework is complementary with these methods, and they can be used in our

framework to estimate worker quality when ground truth is not available, as shown in the experiments.

Some recent work started to investigate quality and budget optimization techniques that can be more tightly integrated with task assignment [26, 4, 9, 17, 5]. However, these methods often implicitly assume that task owners have full control of the crowdsourcing system so that when they need a specific worker to label an item, this worker will be available and willing to complete the task, which may not be a very practical assumption in most of the current crowdsourcing platforms. In our crowd targeting framework, we only target on high quality worker groups instead of individual workers, and therefore the availability of workers is more likely to be guaranteed.

Quite a few work focused on worker characteristics in crowdsourcing. [7, 19] provided detailed analysis of demographic distribution of workers on Amazon Mechanical Turk. [20] proposed to collected demographic information for human subject research so that researchers can know better who are participating in the studies. Recently, Kazai et al. [11, 12] performed some empirical studies on how worker reliability fluctuates cross different worker demographic groups for some relevance assessment tasks.

To the best of our knowledge, there are no previous work that try to utilize worker characteristics to improve data quality and budget efficiency for crowdsourcing, which is the main contribution of this paper.

## 7. DISCUSSION

The Crowd Targeting framework can be applied to many tasks as long as we can measure the worker effect. It is also useful in practice since it can be implemented in the current crowdsourcing platform without too much effort. However, there might be some concerns that task owners might have, such as the question — *what should we do if some characteristics are too sensitive to us?* For example, the task owners might do not want to identify gender for this task. What he/she could do is simply not include the feature in the candidate feature pool, thus the algorithms will not take this feature into account. Note that even the Top-down Discovery Algorithm outputs “gender as female” as an important feature of workers to be in the target group, this does not imply that for all type of tasks, female workers in the crowd will

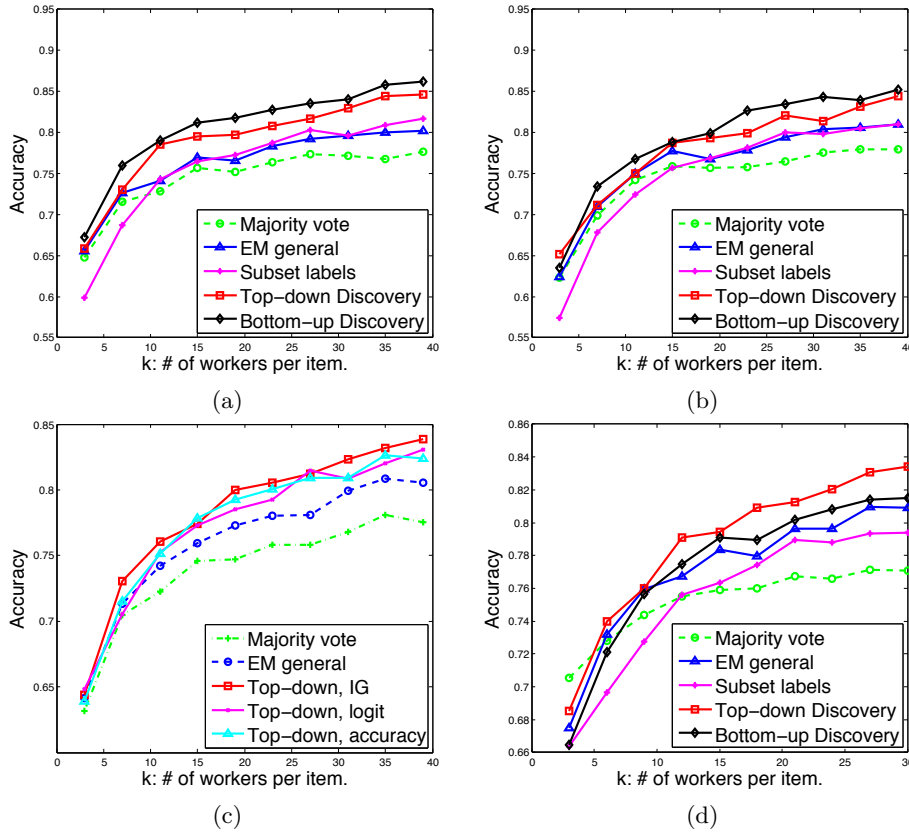


Figure 9: (a),(b) and (c) is on Disambiguation data, and (d) is on RTE data: comparing EM on general crowd and target crowd with majority voting. Use Disambiguation data part 1 for learning in probing stage, and tested on data part 2. (a) With ground truth labels: evaluate workers by comparing with the ground truth labels of 5 randomly drawn “exam” questions (b) Without ground truth labels: draw  $k$  items in dataset part 1 and inference the worker effect without ground truth labels. Discover groups based estimated worker effects. (c) Compare three effect types by running Top-down Discovery Algorithm without ground truth labels on Disambiguation dataset. (d) RTE dataset, learning without ground truth and using same setting as (b).

be suitable. The target crowd is very much task-dependent. Therefore, this framework has no bias to any worker groups in general.

## 8. CONCLUSIONS

As a summary for this paper, we have proposed a Crowd Targeting framework for automatically discovering and targeting at the specific sub-crowd to improve data quality in the data collection process. The targeted crowd is defined by the worker characteristics such as nationality, education level, gender and major etc., or even personality test score and any other screening measures. Meanwhile, we proposed a new measure, named Information Gain, to be the worker effect, which reflects how “strong” a worker is. With fixed effect model, we can study what kind of worker characteristics are associated with the good quality of the data collected from the workers with these characteristics.

For demonstrating the effectiveness of the framework, we designed two major algorithms – Bottom-up Discovery Algorithm and Top-down Discovery Algorithm– to learn the a target crowd with the worker characteristics in the probing stage. Experimental results on the real data have confirmed that by deploying this framework, the performance of the prediction by the same algorithm such as the EM algorithm,

which is prominent and widely used in crowdsourcing community, is significantly improved.

In the future, we plan to explore the optimal strategy of implementing Crowd Targeting framework such as figuring it out the optimal number of ground truth answers in the probing stage, and consider this problem under the budget constraint setting. Another problem is that without any ground truth but with finite budget and finite number of tasks, how we can divide the task set into two parts, put one of them in probing stage and another one in the test stage, for achieving the best possible performance.

## 9. ACKNOWLEDGMENTS

We thank Bin Yu for her comments and generous support on this work, and thank Terry Speed for the advices and discussions on the fixed effect model and categorical data analysis. We thank Ashok Chandra for his valuable comments on this work, and also thank our anonymous reviewers for their thoughtful comments and suggestions. The original idea and primary work of this paper is developed when Hongwei Li is a summer intern in Microsoft Research. During the writing, Hongwei Li is supported by NSF Grant SES-0835531 (Cyber-Enabled Discovery and Innovation).

## 10. REFERENCES

- [1] Y. Bachrach, T. Graepel, T. Minka, and J. Guiver. How to grade a test without knowing the answers—a bayesian graphical model for adaptive crowdsourcing and aptitude testing. *arXiv preprint arXiv:1206.6386*, 2012.
- [2] A. P. Dawid and A. M. Skene. Maximum Likelihood Estimation of Observer Error-Rates Using the EM Algorithm. *Journal of the Royal Statistical Society, Series B (Methodological)*, 28(1):20–28, 1979.
- [3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.
- [4] S. Ertekin, H. Hirsh, and C. Rudin. Learning to predict the wisdom of crowds. *arXiv preprint arXiv:1204.3611*, 2012.
- [5] C.-J. Ho, S. Jabbari, and J. W. Vaughan. Adaptive task assignment for crowdsourced classification. In *ICML*, pages 534–542, 2013.
- [6] P. G. Ipeirotis. Analyzing the amazon mechanical turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students*, 17(2):16–21, 2010.
- [7] P. G. Ipeirotis. Demographics of mechanical turk. In *NYU Digital Working Paper CeDER-10-01*, 2010.
- [8] P. G. Ipeirotis, F. Provost, and J. Wang. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 64–67. ACM, 2010.
- [9] D. R. Karger, S. Oh, and D. Shah. Budget-optimal crowdsourcing using low-rank matrix approximations. In *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*, pages 284–291. IEEE, 2011.
- [10] D. R. Karger, S. Oh, and D. Shah. Iterative learning for reliable crowdsourcing systems. In *NIPS*, pages 1953–1961, 2011.
- [11] G. Kazai, J. Kamps, and N. Milic-Frayling. The face of quality in crowdsourcing relevance labels: Demographics, personality and labeling accuracy. In *Proceedings of the 21st ACM Conference on Information and Knowledge Management (CIKM 2012)*. ACM Press, New York NY, 2012.
- [12] G. Kazai, J. Kamps, and N. Milic-Frayling. An analysis of human factors and label accuracy in crowdsourcing relevance judgments. *Information Retrieval*, 16:138–178, 2013.
- [13] H. Li, B. Yu, and D. Zhou. Error Rate Analysis of Labeling by Crowdsourcing. In *ICML Workshop: Machine Learning Meets Crowdsourcing. Atlanta, Georgia, USA.*, 2013.
- [14] H. Li, B. Yu, and D. Zhou. Error rate bounds in crowdsourcing models. *arXiv preprint arXiv:1307.2674*, 2013.
- [15] Q. Liu, J. Peng, and A. Ihler. Variational inference for crowdsourcing. *NIPS*, 2012.
- [16] R. M. Mickey, O. J. Dunn, and V. Clark. *Applied statistics: analysis of variance and regression*. Wiley-Interscience, 2004.
- [17] T. Pfeiffer, X. A. Gao, Y. Chen, A. Mao, and D. G. Rand. Adaptive polling for information aggregation. In *AAAI*, 2012.
- [18] V. C. Raykar, S. Yu, L. H. Zhao, C. Florin, L. Bogoni, and L. Moy. Learning From Crowds. *Journal of Machine Learning Research*, 11:1297–1322, 2010.
- [19] J. Ross, L. Irani, M. Silberman, A. Zaldivar, and B. Tomlinson. Who are the crowdworkers?: shifting demographics in mechanical turk. In *CHI*, pages 2863–2872. ACM, 2010.
- [20] L. A. Schmidt. Crowdsourcing for human subjects research. *Proceedings of CrowdConf*, 2010.
- [21] P. Smyth, U. Fayyad, M. Burl, P. Perona, and P. Baldi. Inferring Ground Truth from Subjective Labelling of Venus Images. In *NIPS*, 1995.
- [22] R. Snow, B. O. Connor, D. Jurafsky, A. Y. Ng, D. Labs, and C. St. Cheap and Fast - But is it Good ? Evaluating Non-Expert Annotations for Natural Language Tasks. *EMNLP*, 2008.
- [23] L. STAHL and S. Wold. Analysis of variance (anova). *Chemometrics and intelligent laboratory systems*, 6(4):259–272, 1989.
- [24] P. Welinder, S. Branson, S. Belongie, and P. Perona. The Multidimensional Wisdom of Crowds. In *NIPS*, 2010.
- [25] J. Whitehill, P. Ruvolo, T. Wu, J. Bergsma, and J. Movellan. Whose Vote Should Count More : Optimal Integration of Labels from Labelers of Unknown Expertise. In *NIPS*, pages 2035–2043, 2009.
- [26] Y. Yan, G. M. Fung, R. Rosales, and J. G. Dy. Active learning from crowds. In *ICML*, pages 1161–1168, 2011.
- [27] D. Zhou, J. Platt, S. Basu, and Y. Mao. Learning from the Wisdom of Crowds by Minimax Entropy. In *NIPS*, 2012.