Budgeted Task Scheduling for Crowdsourced Knowledge Acquisition

Tao Han[1,2], Hailong Sun[1,2](✉), Yangqiu Song[3], Zizhe Wang[1,2], Xudong Liu[1,2]
[1]State Key Laboratory of Software Development Environment
[2]School of Computer Science and Engineering, Beihang University, Beijing, China
[3]Department of Computer Science and Engineering,
Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong
{hantao,sunhl,wangzz,liuxd}@act.buaa.edu.cn,yqsong@cse.ust.hk

ABSTRACT

Knowledge acquisition (e.g. through labeling) is one of the most successful applications in crowdsourcing. In practice, collecting as specific as possible knowledge via crowdsourcing is very useful since specific knowledge can be generalized easily if we have a knowledge base, but it is difficult to infer specific knowledge from general knowledge. Meanwhile, tasks for acquiring more specific knowledge can be more difficult for workers, thus need more answers to infer high-quality results. Given a limited budget, assigning workers to difficult tasks will be more effective for the goal of specific knowledge acquisition. However, existing crowdsourcing task scheduling cannot incorporate the specificity of workers' answers. In this paper, we present a new framework for task scheduling with the limited budget, targeting an effective solution to more specific knowledge acquisition. We propose novel criteria for evaluating the quality of specificity-dependent answers and result inference algorithms to aggregate more specific answers with budget constraints. We have implemented our framework with real crowdsourcing data and platform, and have achieved significant performance improvement compared with existing approaches.

CCS CONCEPTS

• **Information systems → Crowdsourcing**;

KEYWORDS

Crowdsourcing, task scheduling, knowledge acquisition

1 INTRODUCTION

Crowdsourcing [10] has received increasing attention in both academia and industries. There have been many public crowdsourcing platforms such as Amazon Mechanical Turk and CrowdFlower providing a labor market for requesters to hire workers to process various tasks. One of the core problems of crowdsourcing is that crowdsourcing workers are not always reliable. For instance, some workers may provide incorrect answers due to lack of expertise and in a worse case some spammers may give wrong answers on purpose.As a result, requesters can often get unreliable task answers from workers. To tackle this problem, requesters usually assign the same task redundantly to a certain number of workers simultaneously. Then the returned answers will be aggregated based on inference algorithms, such as majority voting, DS (Dawid and Skene [5]), GLAD [18], Minimax Entropy [21, 23] and etc.

Most state-of-the-art inference algorithms employ minimal error rate between aggregated answers and the ground truth to measure the effectiveness. Intuitively, increasing redundancy can reduce the error rate, and Gao et al. [7] give the upper and lower bounds under the classic DS model, which measure the limit of methods no matter how large the redundancy is. However, more redundant tasks cause more costs. As crowdsourcing budget is always limited, it is necessary to allocate the budget properly. Intuitively, harder tasks need more redundancy while easier tasks need less. Thus, we need proper redundancy for different tasks to attain low error rate and low costs as well. We call this *budgeted task scheduling problem*, where it is necessary to estimate the quality of task answers and consider the remaining budget dynamically when scheduling tasks to more workers. In typical budgeted task scheduling problem, a requester submits a set of tasks together with a certain amount of budget, and the scheduler uses the budget incrementally by deciding which tasks to run in each turn until the budget is used up. For instance, Khetan and Oh [12] and Lahouti and Hassibi [13] present a crowdsourcing task scheduling to achieve minimal error rate with the limited budget.

Task scheduling mechanism is of great importance to effectively solve crowdsourcing problems with budget constraints. However, traditional task scheduling approaches only estimate the answer quality of the tasks using error rate as the criterion. For knowledge acquisition tasks, one of the most common types of crowdsourcing tasks, it is inappropriate to define result quality just with the error rate.

Take an image labeling task as an example. Generally, we consider such a task a classification problem.However, classes can be conceptually organized as a hierarchical taxonomy with the relations of hypernym and hyponym. For instance, if the object in the image is *walker hound*, in traditional error rate view, *foxhound* or *hound* is an error class. But as shown in Figure 1, we know that *walker hound* is a breed of *foxhound* and *foxhound* is a breed of *hound*. Thus we cannot simply treat *hound* or *foxhound* as incorrect results. Therefore, incorporating the knowledge about such conceptual relations can effectively improve the evaluation of the workers' answers. On the other hand, in crowdsourcing tasks, it

CIKM'17 , November 6–10, 2017, Singapore, Singapore

ACM ISBN 978-1-4503-4918-5/17/11…$15.00
https://doi.org/10.1145/3132847.3133002

is expected that the workers can label specific concepts instead of general ones. Obviously, *walker hound* is a better choice than the other two labels in the above example. This is because if we have the knowledge about *walker hound* being a breed of *foxhound*, we can easily generalize the labeled *walker hound* as *foxhound*, but we cannot generalize *foxhound* to *walker hound*. Since the error rate cannot be used to measure the result quality, existing solutions to *budgeted task scheduling problem* cannot be used directly in this scenario.

In this work, we are concerned with *budgeted task scheduling for crowdsourced knowledge acquisition* (BTSK). To solve the BTSK problem, there are mainly two challenges. The first challenge is how to measure the performance of inference methods. To address this challenge, we define two novel performance metrics including *hit rate* and *coherence* along with *accuracy* to measure the quality of inference results. The second challenge is how to estimate the answer quality of each task to determine which tasks should be assigned to more workers for processing. For that, we design an adaptive scheduling framework that can automatically determine which tasks need to solicit more labels, and we provide the quadratic form based estimation methods that can evaluate the answer quality well using the relations between labels. Meanwhile, we improve the existing result inference algorithms to incorporate the new performance metrics in our framework. Then we conducted an extensive set of real crowdsourcing experiments in CrowdFlower[1] and we have analyzed the behavior of workers in the crowdsourcing process. We also evaluated the performance of the improved inference algorithms in comparison with the state-of-the-arts.

Our main contributions are listed as follows.

- We are the first to formalize the budgeted task scheduling for crowdsourced knowledge acquisition problem, and we define novel criteria to measure the quality of results in this context.
- We design a model with a heuristic method to address the BTSK problem with two improved inference algorithms and quadratic form based estimation methods considering the relations between labels.
- We conducted a set of real crowdsourcing experiments to prove the reliability of the criteria and show the advantages of our methods in comparison with the state-of-the-art approaches.

The rest of this paper is organized as follows. Section 2 describes the problem studied in this work and presents three novel metrics of measuring the quality of the aggregated crowdsourcing results. In Section 3, we present the framework of our approach and three core components. And Section 4 describes the experimental evaluation. In Section 5, we analyze the related work and Section 6 concludes this work.

2 PROBLEM FORMULATION

In this section, we first formulate the budgeted task scheduling problem,then introduce two novel performance metrics.

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

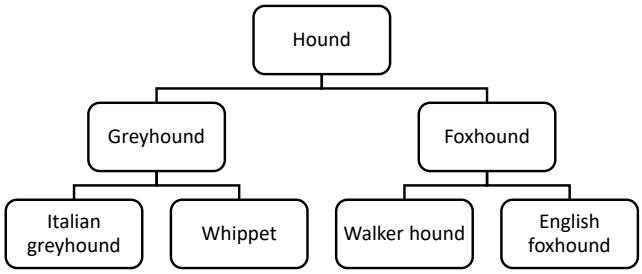

Figure 1: An example of label taxonomy

2.1 Budgeted Task Scheduling

We consider a typical crowdsourcing process. A requester first gives the budget B and the task set $\mathcal{T} = \{t_i | i = 1, 2, ..., n\}$ which needs to be labeled. Then the task set is sent to a crowdsourcing platform that provides a worker set $\mathcal{W} = \{w_j | j = 1, 2, ..., m\}$. For a micro-task t_i, it needs to be labeled for r times by distinct workers. Here r is called redundancy usually provided by the requester. If worker w_j gives task t_i a label l_{ij} from the label domain set $\mathcal{D} = \{x_g | g = 1, 2, ..., k\}$, w_j will get the payment p which satisfies

$$p = \frac{B}{rn}. \tag{1}$$

When the labeling process is finished, we suppose to consume all the budget and get the labeling results as a matrix $\mathbf{L}_r = \left(l_{ij}\right)_{n \times m}$ controlled by redundancy r and we fill the label matrix with zeros when l_{ij} is not defined. Finally, we use an inference method denoted as a function $f : \mathcal{D}^{n \times m} \rightarrow \mathcal{D}^n$, such as majority voting, to infer the ground truth label y_i for $\forall t_i \in \mathcal{T}$, and the inferred result is denoted by $\hat{y}_i$. Then the result set $\hat{\mathbf{Y}} = (\hat{y}_1, \hat{y}_2, ..., \hat{y}_n) = f(\mathbf{L}_r)$ is returned to the requester and we usually use error rate or accuracy as the performance metrics denoted as $v_a(\hat{\mathbf{Y}}, \mathbf{Y})$ to measure the quality of the inferred results.

However, since that different tasks have different difficulty levels, setting the same redundancy r to each task is unreasonable for budget consumption optimization. It brings forth the budgeted task scheduling problem as follows:

Definition 1 (Budgeted Task Scheduling Problem). *We denote task set by $\mathcal{T} = \{t_i | i = 1, 2, ..., n\}$ with size n , worker set by $\mathcal{W} = \{w_j | j = 1, 2, ..., m\}$ with size m, and label domain set by $\mathcal{D} = \{x_g | g = 1, 2, ..., k\}$ with size k. Given budget B, payment p per label, inference method function $f(\cdot)$ and inference performance metric function $v(\cdot)$, the budgeted task scheduling task is to obtain label matrix $\boldsymbol{L_R}$ step by step, determining redundancy $\hat{\boldsymbol{R}} = (\hat{r}_1, \hat{r}_2, ..., \hat{r}_n)$ to satisfy*

$$\sum_{i=1}^{n} \hat{r}_i = \frac{B}{p}, \tag{2}$$

where

$$\hat{\boldsymbol{R}} = \arg\max_{R} v(f(\boldsymbol{L_R}), \boldsymbol{Y}). \tag{3}$$

In practice, the payment can be variable for different tasks with various incentive methods and Eq.(2) should be $B = \sum_{ij} \hat{r}_i p_{ij}$. However, most crowdsourcing platforms using uniform payment and

incentive method can hardly have coherence performance in different situations. Thus, for simplicity, we use uniform payment scheme throughout this work.

Intuitively, harder tasks need greater redundancy. So budgeted task scheduling problem is essentially to estimate the difficulty of a task, then set greater redundancy for harder tasks. But accurately estimating task difficulty needs more labels, and it will lead to less accessible redundancy remaining, which is the main problem in budgeted task scheduling problem. Moreover, for knowledge acquisition scenario, how to define "hard" is still a problem. The performance metric v also needs a proper definition, which will be discussed in the next section.

2.2 Performance Metrics

In a typical crowdsourcing process, we often use error rate/accuracy as the inference performance metric. However, it is not suitable when additional knowledge is incorporated. When alternative labels have hypernym or hyponym relationships (see Figure 1), before the discussion, we first formulate the relationship on the taxonomy tree as a binary function:

$$M(x_g, x_h) = \begin{cases} 1 & \text{if } x_g = x_h \text{ or } x_g \in hypernym(x_h) \\ 0 & \text{otherwise} \end{cases}, \tag{4}$$

where function $hypernym(\cdot)$ follows the definition in Ref. [8]. In our BTSK problem, for instance in Figure 1, if we assume that the ground truth is "English foxhound", traditionally only the label "English foxhound" provided by a worker is considered correct, and all the other labels will be treated as errors. Actually the label "Foxhound" is not totally useless, it is consistent with the correct label "English foxhound" in the higher level of concepts, thus it is not reasonable to simply treat it as an error. In light of this characteristic of BTSK problem, we present a new performance measure called *hit rate* to relax the *accuracy* criterion, using the binary function (4). When a label is the ground truth or the hypernym of ground truth, we say it is a hit label which can be correct. The definition of the *hit rate* criterion is as follows:

$$v_h(\hat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{n} \sum_{i=1}^{n} M(\hat{y}_i, y_i). \tag{5}$$

However, *hit rate* does not tell how well a label is. For knowledge acquisition, we need to acquire specific knowledge. Thus it is necessary to define the *specificity score*. Intuitively, the truth label should be located in a leaf node in the taxonomy tree, which has the largest specificity score. Following this intuition, we define the *specificity score* in the following.

$$S(x_g) = \frac{D(x_g)}{D(x_g) + H(x_g)}, \tag{6}$$

where $D(x_g)$ is the length between the label node x_g and the root node defined by:

$$D(x_g) = \begin{cases} D(\varphi(x_g)) + 1 & \text{if } \varphi(x_g) \neq \emptyset \\ 1 & \text{otherwise} \end{cases}, \tag{7}$$

where $\varphi(x_g)$ is a function returning the parent node of x_g; and $H(x_g)$ is the maximum length between x_g and the hyponym of x_g

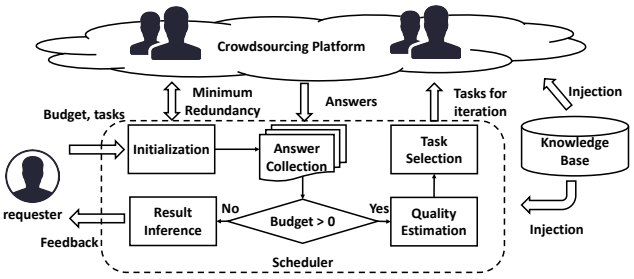

Figure 2: Knowledge Based Iterative Scheduling Framework

defined by

$$H(x_g) = \begin{cases} \max_{x_h \in \phi(x_g)} H(x_h) + 1 & \text{if } \phi(x_g) \neq \emptyset \\ 0 & \text{otherwise} \end{cases}, \tag{8}$$

where $\phi(x_g)$ is the function returning the children set of x_g.

Theorem 1. *$\forall x_g \in \mathcal{D}$, we have $S(x_g) \in (0, 1]$. And $\forall x_h \in hypernym(x_g)$, we have $S(x_h) < S(x_g)$.*

Proof 1. *From the definition of function $D(\cdot)$ and $H(\cdot)$, we have $\forall x_g \in \mathcal{D}, D(x_g) \geq 1, H(x_g) \geq 0$. According to Eq. 6, we can easily prove that $S(x_g) \in (0, 1]$. Then we denote $\forall x_g \in \mathcal{D}, \varphi(x_g) = x_{g'}$ and we have $x_g \in \phi(x_{g'})$. According to the definition,we have $D(x_{g'}) < D(x_g)$ and $H(x_{g'}) > H(x_g)$. Such that:*

$$\begin{aligned} S(x_{g'}) &= \frac{D(x_{g'})}{D(x_{g'}) + H(x_{g'})} < \frac{D(x_{g'})}{D(x_{g'}) + H(x_g)} \\ &< \frac{D(x_g)}{D(x_g) + H(x_g)} = S(x_g). \end{aligned} \tag{9}$$

It means $S(x_{g'}) < S(x_g)$. Because of the transitivity of hypernym, we have $\forall x_h \in hypernym(x_g), S(x_h) < S(x_g)$.

The properties in the Theorem 1 are very useful for describing the specificity concept. Using the definition of specificity score, we can measure how well a label x_g fits with the node x_h by

$$M_S(x_g, x_h) = \begin{cases} S(x_g)/S(x_h) & \text{if } M(x_g, x_h) = 1 \\ 0 & \text{otherwise} \end{cases}. \tag{10}$$

Meanwhile, we present a new performance metric of inference methods called *coherence* defined by:

$$v_c(\hat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{n} \sum_{i=1}^{n} M_S(\hat{y}_i, y_i). \tag{11}$$

3 APPROACH DESCRIPTION

In this section, we introduce the knowledge based iterative scheduling framework to deal with the BTSK problem. And we elaborate on three key issues on quality estimation, task selection and result inference respectively.

Algorithm 1: Knowledge based Iterative Scheduling

Input: $B, p, \mathcal{T} = \{t_i\}, r_{min}, r_{max}, M, \alpha$
Output: $\hat{\mathbf{Y}}$

```
R initialized by r_min;
S^(0) ← initialState(T, R);
B ← B − npr_min;
τ ← 1;
while B> 0 do
    E ← QualityEstimation(S^(τ−1), M);
    T^(τ) ← TaskSelection(E, R, r_max, α);
    R^(τ) ← Crowdsourcing(T^(τ));
    S^(τ) ← inputData(R^(τ), S^(τ−1));
    for t_i in T^(τ) do
        r_i ← r_i + 1;
    B ← B − p|T^(τ)|;
    τ ← τ + 1;
Ŷ ← Inference(S^(τ));
return Ŷ;
```

3.1 Knowledge Based Iterative Scheduling Framework

The knowledge based iterative scheduling framework is shown in Figure 2. A requester first submits the budget and the unlabeled tasks to the scheduling system. The scheduling process consists of five steps:

• **Initialization:** initializing the scheduler with the minimum redundant labels for each task.

• **Answer Collection:** collecting and storing the Answers from crowdsourcing workers.

• **Quality Estimation:** evaluating the quality of the received answers when there is budget left.

• **Task Selection:** determining which tasks should be further sent out for more labels in next iteration.

• **Result Inference:** inferring the truth label by aggregating workers' answers in the event all the budget has been used up.

The scheduling system interacts with the crowdsourcing platform. Both of them have knowledge base injected to capture the relationship among alternative labels helping workers and scheduling system work better. The details of the crowdsourcing scheduling framework are shown in Algorithm 1. The input is the total budget B, payment p per label, task set $\mathcal{T}$, minimum redundancy number r_{min}, maximum redundancy number r_{max}, relation binary function M (Eq. 4), and the parameter range α, adjusting the ratio of tasks which need to go another iteration so as to get new labels. The output is the inferred results of the ground truth of each task. First, we initialize the State $\mathcal{S}$ with r_{min} labels for each task. Then we execute the iterations until the budget is used up. Functions "QualityEstimation" is the implementation of quality estimation part returned to the estimation score vector $\mathbf{E}$. Function "TaskSelection" is the implementation of task selection, which returns the task set $\mathcal{T}^{(\tau)}$ for getting one more label in τ-th iteration. And function "Crowdsourcing" is the process of the crowdsourcing platform to get labels. Function "InputData" is just to input the answer set $\mathcal{R}^{(\tau)}$ from the crowdsourcing platform into the state module $\mathcal{S}^{(\tau-1)}$ returned to $\mathcal{S}^{(\tau)}$ After iteration, function "Inference" implements the result inference module to generate the final result for the requester.

3.2 Quality Estimation

Quality estimation is to compute the quality of workers' answers for each task. A straightforward idea is to compute a weight γ_{ij} of label l_{ij}, then get the estimation score for task t_i with weighted sum:

$$e_i = \sum_{j=1}^{m} \gamma_{ij}. \tag{12}$$

Considering the measurement of specificity, we can use the specificity as the weight: $\gamma_{ij}^S = S(l_{ij})$. If $l_{ij} = 0$, which means w_j did not give a label to t_i, we set $S(l_{ij}) = 0$ and $\gamma_{ij}^S = 0$. So we can get a vector of weight $\mathbf{\Gamma}_i = (\gamma_{i1}, \gamma_{i2}, ..., \gamma_{im})$ for task t_i. Meanwhile, we can consider the confidence that workers have in their answers: $\gamma_{ij}^C = C_{ij}$, where C_{ij} means the confidence of worker w_j giving label to task t_i. No label means $C_{ij} = 0$. However, we find that workers have the tendency to give highly specific label while they show low confidence in doing that, which will be discussed in Section 4.1. Thus we can have the weight by combining both of the two factors: $\gamma_{ij}^{SC} = S(l_{ij})C_{ij}$.

In BTSK problem, if two labels have hypernym or hyponym relationship, we treat them as a good sign to infer a specific label. However, if two labels are located in two branches in the taxonomy tree, such as "Whippet" and "Waller hound" showed in Figure 1, it will make confusion for inference. With the above consideration, we propose the quadratic form based estimation score. We define a matrix $\mathbf{A}^{(i)} = (a_{jj'}^i)_{m\times m}$ for $\forall t_i \in \mathcal{T}$, implementing the relation between the labels:

$$a_{jj'}^i = M(l_{ij}, l_{ij'}), \tag{13}$$

where $M(\cdot,\cdot)$ is defined as Eq. (4). Then we have the quadratic form based estimation score for $\forall t_i \in \mathcal{T}$:

$$e_i = \mathbf{\Gamma}_i \mathbf{A}^{(i)} \mathbf{\Gamma}_i^T, \tag{14}$$

where the weight of each label is the product of specificity and confidence. In this way, we think highly of the answer quality with the labels containing hypernym or hyponym relation, which can easily reveal specific knowledge in tasks.

3.3 Task Selection

In this step, we need to determine which tasks should go for another iteration of crowdsourcing. To that end, we are facing a trade-off between the number of tasks selected for next iteration and the number of total iterations for processing all the tasks. Then we introduce a controlling parameter α, which makes the scale of next iteration as α percent tasks. The details of how to use α to control it are showed in Algorithm 2.

The input is the evaluation vector $\mathbf{E}$ from the evaluation module, redundancy vector $\mathbf{R}$, maximum redundancy number r_{max}, range α. And the output is the task set $\mathcal{T}$ containing the tasks going for another iteration. **isTaskValid** is to figure out which task is valid to get another iteration unless it already gets r_{max} labels. N is the number tasks for another iteration. In Line 6, we set N to $[\alpha n]$ and

Algorithm 2: Task Selection

Input: $\mathbf{E} = (e_1, e_2, ..., e_n)$, $\mathbf{R} = (r_1, r_2, ..., r_n)$, r_{max}, α
Output: $\mathcal{T}$

1 **for** *i from 1 to n* **do**
2 **if** $r_i < r_{max}$ **then**
3 **isTaskValid**$[i] \leftarrow 1$;
4 **else**
5 **isTaskValid**$[i] \leftarrow 0$;
6 $N \leftarrow \min([\alpha n], \sum_{i=1}^{n}$ **isTaskValid**$[i])$;
7 $N \leftarrow \max(1, N)$;
8 $\mathcal{T} \leftarrow \emptyset$;
9 $\mathbf{I} \leftarrow$ get the permutation vector of $\mathbf{E}$ sorting ;
10 **for** *i from 1 to n* **do**
11 **if** $N \leq 0$ **then**
12 **break**;
13 $i' \leftarrow \mathbf{I}[i]$;
14 **if** ***isTaskValid***$[i']$ **then**
15 $\mathcal{T} \leftarrow \mathcal{T} \bigcup t_{i'}$;
16 $N \leftarrow N - 1$;
17 **return** $\mathcal{T}$;

also make sure that N does not exceed the number of valid tasks. Then in Line 7, we ensure one task be selected in any iteration. Finally, we choose the N tasks with the lowest quality scores from the task set.

3.4 Result Inference

This module aims at inferring the ground truth using some inference methods, such as majority voting, DS (Dawid and Skene [5]), GLAD [18], Minimax Entropy [21, 23], and majority voting with knowledge (MWK) and Simplicity-ability Estimation model with External Knowledge (SEEK) [8]. In Section 2.2, we have presented the metric of specificity. On the one hand, a highly specific label implies a high risk of an incorrect label; on the other hand, a less specific label does not provide sufficient knowledge although it usually means a high hit rate. Therefore, we have the following designs to improve the state-of-the-art methods.

3.4.1 MWK+. In the MWK algorithm [8], the weight of each label conducts to its hyponym to reveal more specific label. We set a conduction rate coefficient $\beta \in [0, 1]$ to adjust the transfer process showed in Line 7 of Algorithm 3. In this algorithm, M is the relation binary function defined in Eq. (4). Larger conduction rate β means more intention to get a specific label. Extremely, when $\beta = 1$, it reduces to the MWK method, and when $\beta = 0$, it reduces to the majority voting method or its variant [8].

3.4.2 ProFull. We also propose a probabilistic inference algorithm, which we call ProFull. In probabilistic perspective, the evaluation of ground truth is based on the posterior probability over the labels. In fact, the probability of ground truth should distribute in the leaf node for a specific task. But in [8], all labels in label domain are given by workers. So it will result in the following consequence. For example in Figure 1, the ground truth is "Walker hound" with

Algorithm 3: Improved Majority Voting with External Knowledge

Input: $\mathbf{L} = (l_{ij})_{n \times m}$, M
Output: $\hat{\mathbf{Y}}$

1 **Initialization**:;
2 Worker i's ability parameter $\mu_j^{(0)} \leftarrow 1$;
3 Score for label x_g in task t_i as $\delta_{ig}^{(0)} \leftarrow \frac{\sum_j \mu_j^{(0)} I(l_{ij}=x_g)}{\sum_{g'} \sum_j \mu_j^{(0)} I(l_{ij}=x_{g'})}$;
4 **for** *τ from 1 to maxIter* **do**
5 **if** *ability error < tolerance* **then**
6 **break**;
7 $\delta'_{ig} \leftarrow \delta_{ig}^{(n)} + \sum_{g' \neq g} \beta M(g', g) \delta_{ig}^{(n)}$;
8 Update $\delta_{ig}^{(n+1)} \leftarrow \frac{\delta'_{ig}}{\sum_{g'} \delta'_{ig'}}$; $\mu_j^{(n+1)} \leftarrow \frac{\sum_{i,g} \delta_{ig}^{(n+1)} I(l_{ij}=x_g)}{\sum_{i',g'} \delta_{i'g'}^{(n+1)}}$;
9 $y_i = \arg\max_{x_g} \delta_{ig}$;
10 **return** $\hat{\mathbf{Y}}$;

two workers giving the label "Foxhound" and one worker giving the label "English foxhound". "Walker hound" cannot be revealed by workers so that "Foxhound" is the best label. However, if we can get the full taxonomy tree, the ground truth should distribute in leaf nodes. Thus, we present a probabilistic method based on the full taxonomy tree of labels. We assume that a certain worker w_j has a hit rate parameter π_j which means w_j labeling a label hits the ground truth with probability π_j. The probability of label l_{ij} over ground truth is formulated as:

$$\mathbb{P}(l_{ij}|y_i, \pi_j) = \begin{cases} \pi_j / H(y_i) & \text{if } M(l_{ij}, y_i) = 1 \\ (1 - \pi_j)/(k - H(y_i)) & \text{if } M(l_{ij}, y_i) = 0 \end{cases}. \quad (15)$$

Then we solve this problem using EM algorithm:

E-step: We compute probability $\mathbb{P}(y_i|\mathbf{L}, \boldsymbol{\pi})$ for $\forall y_i \in \mathcal{D}$ by

$$\mathbb{P}(y_i|\mathbf{L}, \boldsymbol{\pi}) \propto \mathbb{P}(y_i) \prod_i \mathbb{P}(l_{ij}|y_i, \pi_j), \quad (16)$$

where we set $\mathbb{P}(y_i)$ is uniform distribution;

M-step: we figure out the parameters $\boldsymbol{\pi}$ by

$$\boldsymbol{\pi} = \arg\max_{\boldsymbol{\pi}} Q(\boldsymbol{\pi}^{old}, \boldsymbol{\pi}), \quad (17)$$

where Q is the standard auxiliary function that

$$Q(\boldsymbol{\pi}^{old}, \boldsymbol{\pi}) = \sum_{ij} \sum_{y_i \in \mathcal{D}} \mathbb{P}(y_i|\mathbf{L}, \boldsymbol{\pi}^{old}) \ln \mathbb{P}(l_{ij}|y_i, \pi_j). \quad (18)$$

Let $\frac{\partial Q}{\partial \pi_j} = 0$, we have

$$\pi_j = \frac{\sum_i \sum_{M(l_{ij}, y_i)=1} \mathbb{P}(y_i|\mathbf{L}, \boldsymbol{\pi}^{old})}{\sum_i I_{ij}}, \quad (19)$$

where I_{ij} is the indicator variable whether worker w_j gives a label to task t_i.

If the distribution of ground truth on leaf nodes are dispersive, the inference can be hardly reliable. We had better get the hypernym with higher hit probability, extremely the hit probability of root node is 1. Thus, we set a parameter σ which is the minimum hit

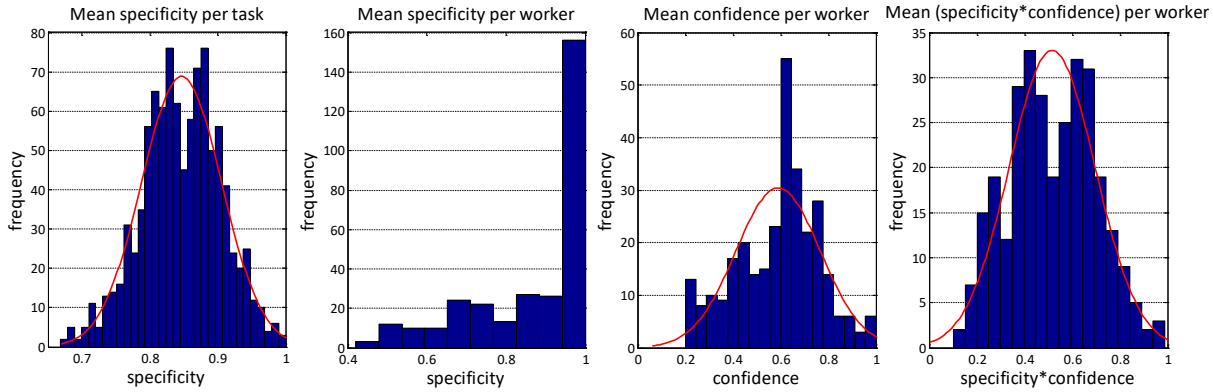

Figure 3: Distribution of specificity and confidence in real data

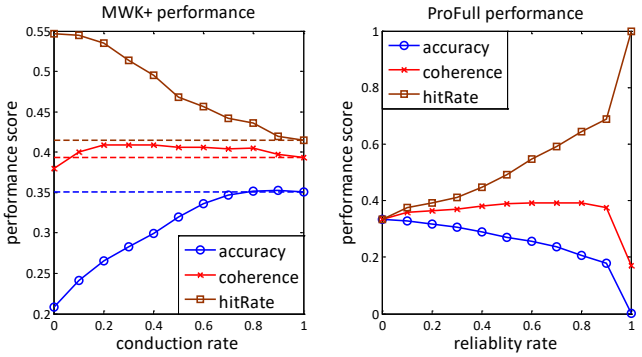

Figure 4: Performance of MWK+ and ProFull with their parameters respectively

probability of the result we need. Smaller σ means relaxing constraints with the aim of inferring more specific labels while larger σ indicates the priority of hit rate over specificity.

4 EVALUATION

In this section, we present the experimental results. We introduce the real data we use and analyze the workers' behavior in Section 4.1. Then about the inference methods, we test the performance of our MWK+ and ProFull algorithm and compare with other inference methods in Section 4.2. Next, we conduct a set of experiments on our scheduling to prove the effectiveness of our quadratic form scheduling in Section 4.3 and the robustness by range parameter in Section 4.4. Finally, we come to the budget issue and prove that our work can make remarkable budget saving in Section 4.5.

4.1 Datasets

We first extract a taxonomy tree from ImageNet [6] with 149 nodes with "dog" as the root node. We get images form the leaf nodes of the tree as data for labeling. In total, we get 984 images to implement crowdsourcing labeling tasks in crowdsourcing platform CrowdFlower[2]. To address the adaptive redundancy problem, we get 19 valid labels for each image and 303 workers participated in the labeling tasks. Meanwhile, we require worker give his or her confidence along with a label from the taxonomy tree and the label we require should best describe the image and as specific as possible.

We compute the statistics on the specificity of 18,696 labels and their confidence in Figure 3. The distribution of label specificity mean per task is close to Gaussian distribution but the label specificity mean per worker is not. It seems that worker tends to give high specificity label. Meanwhile, the confidence mean of worker tends to be in the middle. The mean of the product of label specificity and worker's confidence for each worker comes to Gaussian distribution again. It means the high specificity labels with low confidence are with high probabilities. It proves the usefulness of product of specificity and confidence from another perspective.

4.2 Evaluation of Result Quality

We compare our method with several other approaches. MV (Majority Voting) and MWW (Majority With Weight) are the basic inference in crowdsourcing. MWK (Majority With Knowledge) is the method presented by Han et al. [8], which incorporates external knowledge to deal with inference problem in BTSK problem. SEEK method presented by Han et al. [8] is hard to adjust parameter and always over time, and DS or Minimax Entropy is proved not fitting to the BTSK problem. Thus, we do not compare with SEEK, DS and Minimax Entropy methods. MWK+ is our methods based on MWK. MWC (Majority With Confidence) uses the confidence as the weight. ProFull is our probabilistic method using full taxonomy tree, solved by EM algorithm.

We incorporate two parameters into our inference methods MWK+ and ProFull respectively, as introduced in Section3.4. The first parameter is conduction rate β. In the left figure of Figure 4, the dashed line is the performance of the MWK method, and the solid line stands for the performance of the MWK+ method. We can see that the accuracy, coherence, and hitRate of MWK+ are all better than MWK when $\beta \in (0.8, 1)$. And when $\beta = 1$, the performance of MWK+ is the same as MWK. When $\beta = 0$, which means no weight conduction between labels, MWK+ comes to equal to MWW. The second one is reliable rate σ. In the right figure of Figure 4, when the σ increases, the accuracy decreases, and the hit rate increases.

[2]We have released our source code and data: https://github.com/crowdintelligence/BTSK

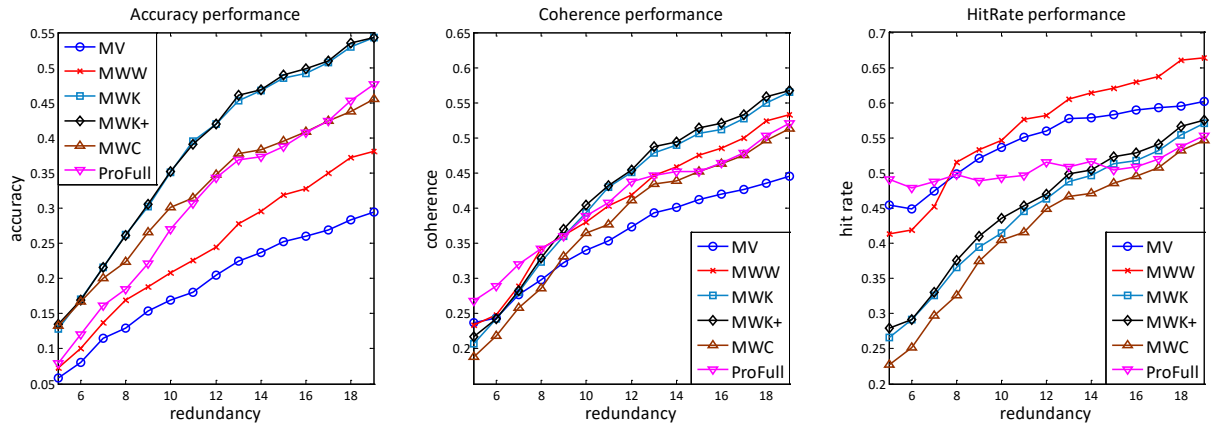

Figure 5: Performance of inference methods (MWK+:$\beta = 0.8$, ProFull:$\sigma = 0.5$)

Table 1: Scheduling methods performance on inference algorithms comparing with no scheduling method ($r_{mean} = 10$, $\alpha = 0.5$)

Inference Method		Uniform	Random	S-weight	C-weight	SC-weight	QF
			vs. Uniform				
MV	accuracy	0.170	-4.79%	0.00%	-1.80%	1.20%	**12.57%**
	coherence	0.340	-2.90%	-1.67%	-0.85%	0.04%	**5.45%**
	hit rate	0.537	-2.16%	-2.08%	0.00%	-0.19%	**1.89%**
MWW	accuracy	**0.208**	-6.98%	-1.46%	-6.34%	-6.34%	-13.17%
	coherence	**0.380**	-1.30%	-2.32%	-4.08%	-2.62%	-2.61%
	hit rate	0.547	0.82%	-1.86%	-2.60%	0.19%	**1.67%**
MWK	accuracy	0.351	-4.99%	3.19%	2.03%	8.41%	**13.62%**
	coherence	0.393	-2.79%	1.74%	1.84%	5.95%	**8.28%**
	hit rate	0.415	-1.27%	0.25%	1.72%	4.41%	**5.64%**
MWK+	accuracy	0.352	-5.06%	2.89%	0.87%	8.09%	**12.72%**
	coherence	0.405	-3.72%	0.15%	-0.18%	3.72%	**5.67%**
	hit rate	0.436	-2.94%	-2.33%	-0.70%	0.93%	**2.10%**
MWC	accuracy	0.301	-5.44%	0.68%	0.34%	6.08%	**9.46%**
	coherence	0.364	-4.65%	-0.54%	-0.68%	2.47%	**6.97%**
	hit rate	0.404	-4.37%	-1.51%	-1.76%	0.50%	**5.28%**
ProFull	accuracy	0.270	-4.32%	-4.51%	-1.13%	3.01%	**9.40%**
	coherence	0.389	-1.97%	-2.94%	-1.30%	-0.88%	**3.56%**
	hit rate	0.493	0.74%	-1.03%	-0.21%	-1.65%	**1.65%**

When $\sigma = 0$, hit rate meets to accuracy, meanwhile when $\sigma = 1$, hit rate equals to 1 and accuracy equals to 0 where we need the most reliable label which is the root label always fitting the task.

Figure 5 shows the performances of six compared methods in accuracy, coherence, and hit rate. We set $\beta = 0.8$ and $\sigma = 0.5$ for MWK+ and ProFull respectively. In accuracy, MWK+ performs best almost all the time. MWK method also has a good performance in accuracy, and it performs best when the number of redundancy is from 8 to 10. In terms of coherence, ProFull performs best when the number of redundancy is less than 9, and the MWK+ method performs best when the number of redundancy is larger than 9. MWK method has a close performance to MWK+ method. But in hit rate, MWK and MWK+ method performs worse than the others except MWC method. MWW method performs best and majority voting is second when the number of redundancy is larger than 8. When the number is less than 8, ProFull method performs best.

4.3 Effects of Quality Estimation

After the analysis of the inference methods behavior, we implement the whole scheduling by 6 ways respectively using 6 inference methods, as shown in Table 1. Each inference method is evaluated by three measures: accuracy, coherence and hit rate. We set appropriate parameters for MWK+ and ProFull where conduction rate $\beta = 0.8$ in MWK+, and reliable rate $\sigma = 0.5$ in ProFull.

The budget of all experiments in the table can get 10×984 labels for the scheduling to the assignment, which means each experiment has the same redundancy mean 10. The column with title "Uniform" is the benchmark which has no scheduling process and just assigns 10 redundancy for each task. This column shows the value of the performance measure for each inference method. The remaining five columns show the five scheduling methods, in which the percentage is the increase or decrease base on no scheduling method "Uniform." "Random" scheduling is assigning redundancy randomly. We implement each for 10 times and get the mean as the result. S-weight scheduling means using quality estimation part with the weight of specificity. C-weight scheduling means the weight of confidence and SC-weight means the weight of the product of specificity and confidence. QF (short for quadratic form) scheduling method considers the semantic relation between labels using quadratic form base quality estimation. For all the crowdsourcing scheduling approaches, we use the same task selection part whose range parameter is $\alpha = 0.5$. From Table 1 we can see that the "Random" scheduling is worse than no scheduling method "Uniform." "S-weight" scheduling and "C-weight" scheduling have about half to improvement and half to a setback. "SC-weight" scheduling improves based on most of the inference algorithms. However, all above do not consider the semantic relation of labels whereas "QF" scheduling does. The performance of "QF" scheduling is better than others and demonstrates the best result based on all inference methods except for MWW. In traditional accuracy measure with no scheduling "Uniform," MWK+ gets the best result 0.352 and "QF" improves it by 12.72%. Meanwhile, MWK+ also gets the best result 0.405 in coherence and "QF" improves it by 5.67%. In hit rate measurement, MWW gets the best result 0.547 and "QF" also improves it by 1.67%. These experiments show that "QF" evaluation has remarkable improvement and stable performance.

4.4 Effects of Task Selection

In quality estimation part, in fact, we want to infer which task has more potential to improve inferring the ground truth by another one label. If the prediction of "potential" with high probability accuracy, we will get a good quality estimation part. Moreover, predicting top 25% tasks to be labeled in the next iteration with most potential is more difficult than predicting top 50% because of the stricter requirement. However, the range parameter α plays the role of requirement. A smaller α means fewer tasks should go another iteration to get a label for each. Meanwhile, it also brings more meticulous scheduling to estimate answer quality of each task. If the prediction of "potential" is accuracy enough, it will show that smaller α results in better performance, otherwise results in unforeseeable noise to offset the tendency.

In Figure 6, we diversify the range α from 0.1 to 1 that task selection module would make a decision which α-percent tasks should get another a label. When α going to equal to 1, it means all the tasks go for next iteration until budget consumed and each task would get the same redundancy at last. So when $\alpha = 1$, Scheduling has the same performance of "Uniform". From Figure 6 we can see that the performance of all approaches except for MWW, become decreased more or less during α increasing, especially for MWK, MWK+ and MWC. "QF" makes a good prediction as expected. However, when α is less than 0.5, there is no sign to have any improvement. Moreover, the bottom left sub-figure in Figure 6 shows that the number of iterations has a negative correlation with range parameter α. The iterations depend on the crowdsourcing platform, and it will take a quite long time for an iteration. Thus, too small α is not a good choice because of the increasing iterations. We need set a proper range parameter to get a remarkable improvement of the result and have few increased iterations.

4.5 Evaluation of Costs

In this section, we present the results related to the budget issue, which is a very important problem in crowdsourcing. In section 4.3 and Section 4.4, we show the good performance of the "QF" scheduling to improve the crowdsourcing aggregation result with the limited budget. In other words, it means to achieve the same quality of the result, "QF" scheduling needs less budget than other methods. We set benchmark budget offering 10 redundancy for each task that obtaining 10×984 labels from crowdsourcing using "Uniform" method. In Figure 7, the dashed line uses the benchmark budget as the baseline, and we set redundancy mean from 7 to 12 using "QF" scheduling to draw the solid lines with range parameter $\alpha = 0.5$. The intersection of the dashed line and solid line in the same color (pointed by arrows) means the redundancy mean using "QF" scheduling and making the same quality of the result. The benchmark is with 10 redundancy, and the arrows pointing to the values of x-axis less than 10 mean we can save budget, vice versa. In the figure, we can see except for MWW, the rest inference approaches are saving budget more or less. In MWC inference method, our scheduling saves about 10% budget in any three performance measures. Note that using commercial platforms such as CrowdFlower, the transaction fee in the platform is 20% of the budget. So in comparison, our work can obtain a remarkable saving result in BTSK problem.

5 RELATED WORK

In crowdsourcing, task redundancy is a commonly used method to improve the result quality at the cost of increased payment. Thus given a certain amount of budget, how to scheduling tasks can largely affect the result quality. Several efforts [2, 11, 16, 17, 20] have been made on crowdsourcing task scheduling under budget constraints. Bansal et al. [2] present an active learning scheme for document selection that aims at maximizing the overall relevance label prediction accuracy for a given budget of available relevance judgments by exploiting system-wide estimates of label variance and mutual information. Karger et al. [11] propose a probabilistic model for general crowdsourcing in consideration of both task allocation and inference. And their main contribution is a theoretical analysis of the total budget for achieving certain collective quality. Yu et al. [20] propose a novel quality and budget aware spatial task allocation approach which jointly considers the workers' reputation and proximity to the task locations to maximize the expected quality of the results while staying within a limited budget. Tran-Thanh et al. [16] propose an algorithm called BudgetFix, it determines the number of interdependent micro-tasks and the price to pay for each task given budget constraints. Tran-Thanh et al. [17] present an algorithm called Budgeteer to solve the problem of task allocation in crowdsourcing systems with multiple

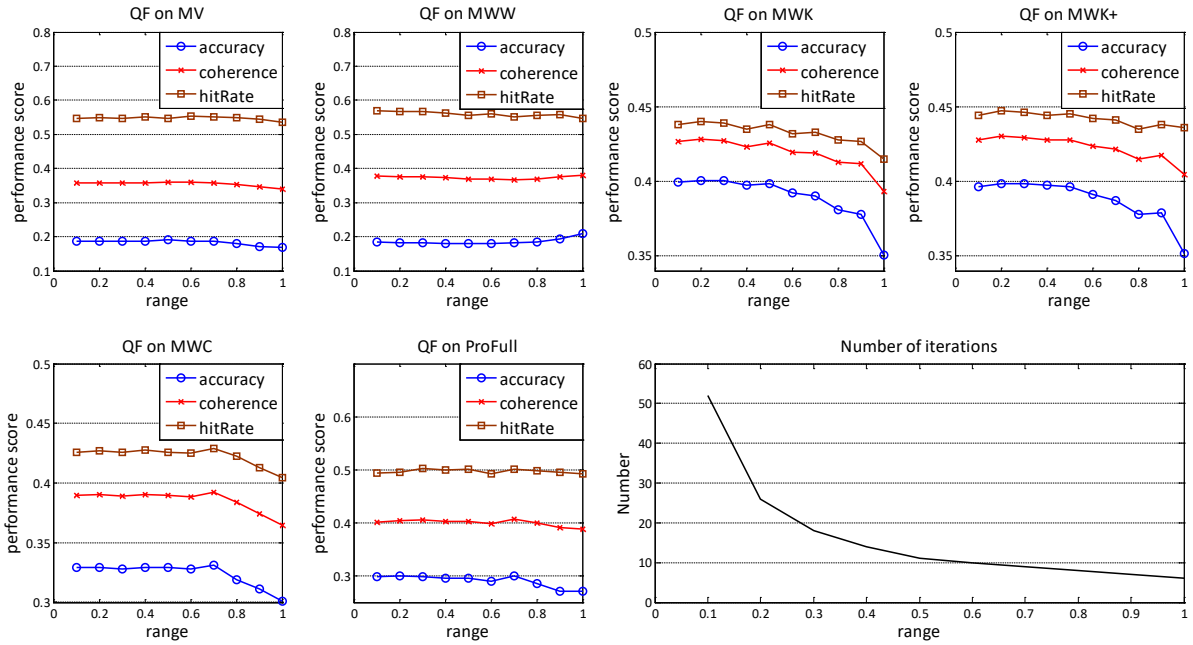

Figure 6: Performance of quadratic form scheduling over range α ($r_{mean} = 10$)

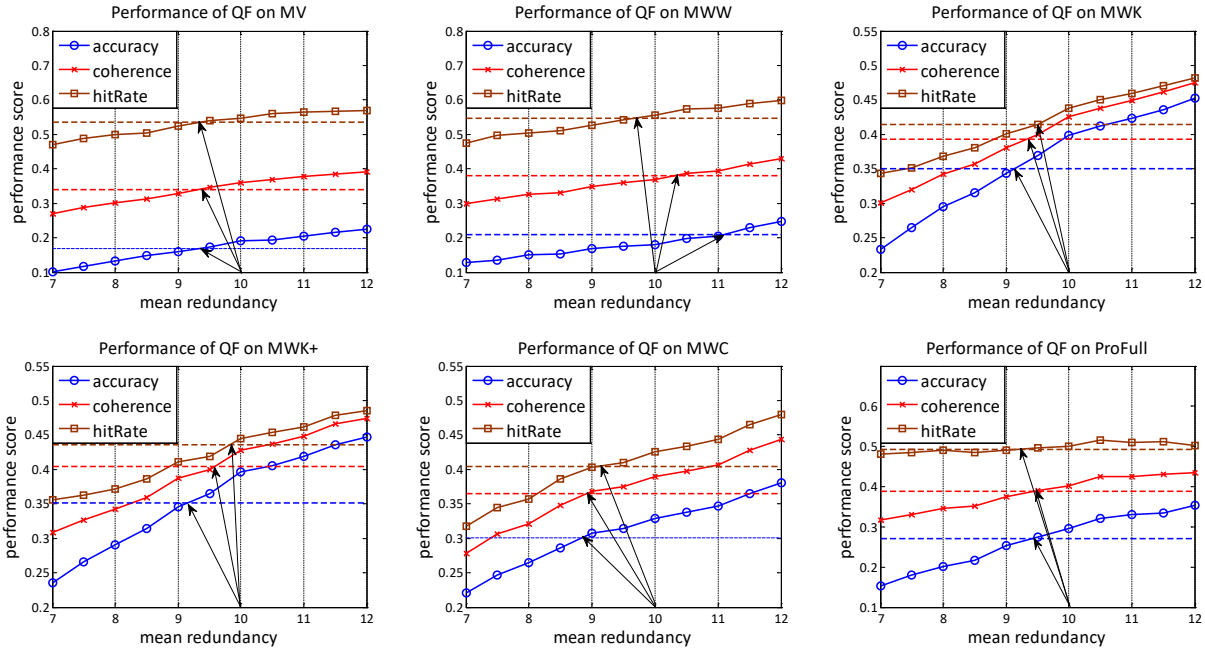

Figure 7: Budget saving of quadratic form scheduling comparing with uniform method ($\alpha = 0.5$, Uniform:$r = 10$)

complex workflows under a budget constraint, which calculates an efficient way to allocate budget to each workflow. However, existing work either targets general crowdsourcing applications or specifically focuses on particular crowdsourcing tasks, which cannot be straightforwardly employed to address our labeling tasks considering specific knowledge.

As labeling is one of the most feasible types of tasks that fit for crowdsourcing, a large amount of existing work [1, 5, 14, 19, 22] on crowdsourcing takes labeling as the target tasks. Most existing methods treat each answer as either true or false. Actually, there can be hierarchical semantic relations among workers' answers, in which case simply classify an answer as binary is not good for

acquiring specific knowledge. There have been a few studies [4, 15] on acquiring binary relationships to construct a taxonomy of concepts and using the taxonomy to classify items based on multi-label classification [3]. Han et al. [9] further propose to explicitly use specificity to measure the result quality in crowdsourcing task processing. However, existing studies on labeling tasks do not support task scheduling with limited budget, thus cannot find an optimal tradeoff between quality and costs.

In this work, we are concerned with designing a task scheduling framework in consideration of budget limit in the context of knowledge acquisition tasks. Our work is the first attempt to consider budgeted task scheduling in dealing with specific knowledge acquisition problems with crowdsourcing.

6 CONCLUSION

Budgeted task scheduling problem is of paramount importance in crowdsourcing. Most existing methods use error rate or accuracy to measure the result quality, which are not suitable for crowdsourcing tasks like knowledge acquisition as concerned in this work. We first formalize the budgeted tasks scheduling for crowdsourced knowledge acquisition (BTSK) problem. Then we propose novel criteria to measure the answer quality in the context of BTSK problems. Next, we present a knowledge based iterative scheduling framework to address the BTSK problem. There are mainly three key issues to address in our framework. Among them, *inference* is responsible for inferring the final results on the basis of the received answers from workers, and we design two algorithms including MWK+ and ProFull in this regard. Second, *quality estimation* concerns estimating the potential quality of a task so as to determine whether it is necessary to solicit more answers from extra workers, and we present a quadratic form based quality estimation method through considering the semantic specificity of an answer and the semantic relation between labels. Third, in *task selection*, we use the range parameter α to adjust the scheduling to get a better result with few iterations. We have implemented our scheduling framework in real crowdsourcing platform and analyzed the data. We find that almost half of the workers in platform often give most specific labels but with low confidence. We have evaluated our solution extensively in comparison with the state-of-the-art methods and the results demonstrate the advantages and effectiveness of our methods in the context of knowledge acquisition task.

ACKNOWLEDGEMENTS

This work was supported partly by the National Basic Research 973 Program of China under Grant No(s) 2014CB340304, 2015CB358700, and the State Key Laboratory of Software Development Environment under Grant No. SKLSDE-2017ZX-14. And the authors would thank the anonymous reviewers for the helpful comments and suggestions to improve this paper. The corresponding author of this paper is Hailong Sun.

REFERENCES

[1] Yoram Bachrach, Thore Graepel, Tom Minka, and John Guiver. 2012. 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] Piyush Bansal, Carsten Eickhoff, and Thomas Hofmann. 2016. Active Content-Based Crowdsourcing Task Selection. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 529–538.

[3] Jonathan Bragg, Mausam, and Daniel S. Weld. 2013. Crowdsourcing Multi-Label Classification for Taxonomy Creation. In *AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*.

[4] Lydia B Chilton, Greg Little, Darren Edge, Daniel S Weld, and James A Landay. 2013. Cascade: Crowdsourcing taxonomy creation. In *SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1999–2008.

[5] Alexander Philip Dawid and Allan M Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied statistics* (1979), 20–28.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*. 248–255.

[7] Chao Gao, Yu Lu, and Dengyong Zhou. 2016. Exact exponent in optimal rates for crowdsourcing. In *International Conference on Machine Learning (ICML)*.

[8] Tao Han, Hailong Sun, Yangqiu Song, Yili Fang, and Xudong Liu. 2016. Incorporating External Knowledge into Crowd Intelligence for More Specific Knowledge Acquisition. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16)*.

[9] Tao Han, Hailong Sun, Yangqiu Song, Yili Fang, and Xudong Liu. 2016. Incorporating External Knowledge into Crowd Intelligence for More Specific Knowledge Acquisition. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*. 1541–1547.

[10] Jeff Howe. 2006. The rise of crowdsourcing. *Wired magazine* 14, 6 (2006), 1–4.

[11] David R Karger, Sewoong Oh, and Devavrat Shah. 2014. Budget-optimal task allocation for reliable crowdsourcing systems. *Operations Research* 62, 1 (2014), 1–24.

[12] Ashish Khetan and Sewoong Oh. 2016. Achieving budget-optimality with adaptive schemes in crowdsourcing. In *Advances in Neural Information Processing Systems*. 4844–4852.

[13] Farshad Lahouti and Babak Hassibi. 2016. Fundamental Limits of Budget-Fidelity Trade-off in Label Crowdsourcing. In *Advances in Neural Information Processing Systems*. 5059–5067.

[14] Mahyar Salek, Yoram Bachrach, and Peter Key. 2013. Hotspotting-A Probabilistic Graphical Model For Image Object Localization Through Crowdsourcing.. In *AAAI*.

[15] Yuyin Sun, Adish Singla, Dieter Fox, and Andreas Krause. 2015. Building Hierarchies of Concepts via Crowdsourcing. In *IJCAI*. 844–851.

[16] Long Tran-Thanh, Trung Dong Huynh, Avi Rosenfeld, Sarvapali D Ramchurn, and Nicholas R Jennings. 2014. BudgetFix: budget limited crowdsourcing for interdependent task allocation with quality guarantees. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 477–484.

[17] Long Tran-Thanh, Trung Dong Huynh, Avi Rosenfeld, Sarvapali D Ramchurn, and Nicholas R Jennings. 2015. Crowdsourcing Complex Workflows under Budget Constraints.. In *AAAI*. 1298–1304.

[18] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*. 2035–2043.

[19] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *NIPS*. 2035–2043.

[20] Han Yu, Chunyan Miao, Zhiqi Shen, and Cyril Leung. 2015. Quality and budget aware task allocation for spatial crowdsourcing. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 1689–1690.

[21] Denny Zhou, Sumit Basu, Yi Mao, and John C Platt. 2012. Learning from the wisdom of crowds by minimax entropy. In *Advances in Neural Information Processing Systems*. 2195–2203.

[22] Dengyong Zhou, Sumit Basu, Yi Mao, and John C Platt. 2012. Learning from the wisdom of crowds by minimax entropy. In *NIPS*. 2195–2203.

[23] Dengyong Zhou, Qiang Liu, John Platt, and Christopher Meek. 2014. Aggregating ordinal labels from crowds by minimax conditional entropy. In *International Conference on Machine Learning*. 262–270.