T2R2 東京工業大学リサーチリポジトリ
Tokyo Tech Research Repository

論文 / 著書情報
Article / Book Information

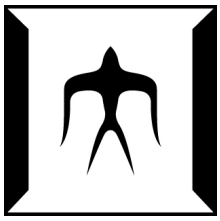
題目(和文)	群衆の叡智からの統計的学習
Title(English)	Statistical Learning from the Wisdom of Crowds
著者(和文)	ZHANGHAO
Author(English)	Hao Zhang
出典(和文)	学位:博士(工学), 学位授与機関:東京工業大学, 報告番号:甲第10239号, 授与年月日:2016年3月26日, 学位の種別:課程博士, 審査員:杉山 将,徳永 健伸,篠田 浩一,村田 剛志,石田 貴士
Citation(English)	Degree:Doctor (Engineering), Conferring organization: Tokyo Institute of Technology, Report number:甲第10239号, Conferred date:2016/3/26, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Powered by T2R2 (Tokyo Institute Research Repository)

Statistical Learning from the Wisdom of Crowds

Hao Zhang

March 2016



Department of Computer Science
Graduate School of Information Science and Engineering
Tokyo Institute of Technology

Thesis Committee:
Masashi Sugiyama, Chair
Takenobu Tokunaga
Koichi Shinoda
Tsuyoshi Murata
Takashi Ishida

Submitted in partial fulfillment of
the requirements for the degree of

Doctor of Engineering

Copyright © 2016 Hao Zhang

Keywords: crowdsourcing, machine learning, task-worker assignment, online decision making, uncertainty sampling

To my parents

Abstract

Labeled data is essential to many machine learning applications. In recent years, *crowdsourcing* has emerged as a useful tool for combining human intelligence by asking a crowd of low-paid workers to complete a group of labeling tasks. This doctoral thesis is devoted to developing statistical methods of learning from the wisdom of crowds.

Since workers in crowds are usually non-experts, the collected labels often contain a significant amount of noises. To cope with this, many crowdsourcing methods jointly estimate the true labels and workers' reliability on the collected labels. However, they do not focus on how to collect these labels for the purpose of wisely using the total budget of label collection. This motivates us to consider another important problem called *task-worker assignment*, which is to repeatedly make decisions on which task is to assign to which worker.

The task-worker assignment problem naturally includes two aspects: worker selection and task selection. Worker selection aims to select reliable workers given any task. A challenge in this scenario is how to deal with the diversity of tasks and the involved trade-off between exploration and exploitation. In this thesis, we propose the *bandit-based task assignment* (BBTA) method, which is a contextual bandit formulation for worker selection in crowdsourcing. We also theoretically investigate the regret bounds for BBTA, and experimentally show its practical usefulness.

As the other important part of task-worker assignment, task selection focuses on selecting an appropriate task at each round. Although worker selection has been successfully addressed by BBTA, task selection has not been thoroughly investigated yet. In this thesis, we further borrow ideas of uncertainty sampling in active learning, and develop several task selection strategies. Experimental results

show that the *least confidence* strategy significantly improves the performance of task-worker assignment in crowdsourcing.

Finally, we conclude that the proposed strategies of worker selection and task selection for crowdsourcing are successful and worth further investigating in the future.

Acknowledgments

First and foremost, I would like to express my deep gratitude to my supervisor, Professor Masashi Sugiyama, for his patient guidance, inspiring discussions, and honest criticism (which was so lenient). He has provided me with a wonderful study environment since I was a research student. I also have learned a lot from him, for example, diligent efforts, efficient planning, and positive thoughts. He has taught me not only how to conduct good research, but also how to enjoy doing research, which was far more important in my Ph.D. study, and will also definitely influence my future career.

Many thanks to my secondary supervisor, Professor Atsushi Fujii, who was always warm-hearted whenever I needed help.

I would like to thank my thesis committee members: Professor Takenobu Tokunaga, Professor Koichi Shinoda, Professor Tsuyoshi Murata, and Professor Takashi Ishida, for reviewing my thesis and giving me many insightful comments.

I was fortunate enough to be a member of Sugiyama Lab, where I shared a lot of happy memories with my colleagues and friends: Dr. Yao Ma, Dr. Gang Niu, Dr. Marthinus Christoffel du Plessis, Hideko Kawakubo, Kishan Wimalawarne, Hiroyuki Tanaka, Tomoya Sakai, Ikko Yamane, Voot Tangkaratt, Hiroaki Shiino, Jie Luo, Hongyi Ding.

Special thanks to our lab secretaries: Yuko Kawashima and Ayako Tamai, who helped me a lot to deal with many (and sometimes complex) daily affairs.

I gratefully acknowledge the financial support for my Ph.D. study from the Japanese government MEXT scholarship and the CREST program, without which I could not have led a normal life in Japan.

Most importantly, I would like to thank my parents for all the love and support they have given to me during my Ph.D. study and all the time.

Contents

Abstract **v**

Acknowledgments **vii**

List of Figures **xii**

List of Tables **xiii**

1 Introduction **1**
1.1 Learning from Labeled Data 1
1.1.1 Machine Learning and Big Data 1
1.1.2 Supervised Learning 2
1.1.3 Labeled Data versus Unlabeled Data 4
1.2 The Wisdom of Crowds 4
1.2.1 Crowdsourcing Marketplaces 5
1.2.2 Crowds versus Experts 5
1.2.3 Crowdsourcing Research 6
1.3 Online Decision Making in Crowdsourcing 10
1.3.1 Task-Worker Assignment 10
1.3.2 Dynamic Procurement 13
1.4 Contributions 13
1.4.1 Worker Selection in Crowdsourcing 14
1.4.2 Task Selection in Crowdsourcing 15
1.5 Organization 16

2 Worker Selection in Crowdsourcing **19**
2.1 Introduction 19
2.2 Bandit-Based Task Assignment 22
2.2.1 Problem Formulation 22
2.2.2 Pure Exploration Phase 23
2.2.3 Adaptive Assignment Phase 26

2.3 Theoretical Analysis . . . 28
2.3.1 Regret . . . 29
2.3.2 Regret Bounds . . . 30
2.4 Existing Methods . . . 31
2.4.1 IEThresh . . . 32
2.4.2 CrowdSense . . . 32
2.4.3 Optimistic Knowledge Gradient . . . 33
2.4.4 Discussion . . . 35
2.5 Experiments . . . 37
2.5.1 Benchmark Data . . . 37
2.5.2 Real Data . . . 50
2.6 Proof of Theoretical Results . . . 56

3 Task Selection in Crowdsourcing 61
3.1 Introduction . . . 61
3.2 Task Selection . . . 62
3.2.1 Problem Setting and Notation . . . 62
3.2.2 Task Selection Strategies . . . 63
3.2.3 Least Confidence . . . 64
3.2.4 Margin Sampling . . . 64
3.2.5 Information Density . . . 65
3.3 Experiments . . . 65
3.3.1 Benchmark Data . . . 66
3.3.2 Real Data . . . 70

4 Conclusions and Future Work 77
4.1 Conclusions . . . 77
4.2 Future Work . . . 78
4.2.1 Within BBTA . . . 78
4.2.2 Beyond BBTA . . . 80

Bibliography 83

List of Figures

1.1 A simplified framework of supervised learning with expert labels. 3
1.2 A simplified framework of supervised learning with crowd labels. 7
1.3 The task-worker assignment problem in crowdsourcing. 11
1.4 Organization of this thesis. 17

2.1 Distribution of true accuracy of simulated workers for three benchmark datasets with three worker models. 38
2.2 Results on three benchmark datasets with the spammer-hammer worker model. 40
2.3 Results on three benchmark datasets with the one-coin worker model. 41
2.4 Results on three benchmark datasets with the one-coin worker model (malicious). 42
2.5 Illustration of the regret and its bound in Theorem 2.1 as the function of N' on benchmark data. 45
2.6 Results of changing N' on three benchmark datasets with the spammer-hammer worker model. 47
2.7 Results of changing N' on three benchmark datasets with the one-coin worker model. 48
2.8 Results of changing N' on three benchmark datasets with the one-coin worker model (malicious). 49
2.9 Results on three benchmark datasets with various numbers of contexts. 51
2.10 Distribution of true accuracy of workers for RTE data ($N = 800, K = 164, S = 1$). 52
2.11 Results on RTE data ($N = 800, K = 164, S = 1$). 53
2.12 Distribution of true accuracy of workers for GH data ($N = 204, K = 42, S = 2$). 54
2.13 Results on GH data ($N = 204, K = 42, S = 2$). 55

3.1 Results of different task selection strategies on three benchmark datasets with the spammer-hammer worker model. 67

3.2 Results of different task selection strategies on three benchmark datasets with the one-coin worker model. 68
3.3 Results of different task selection strategies on three benchmark datasets with the one-coin worker model (malicious). 69
3.4 Results of varying β in ID strategy on three benchmark datasets with the spammer-hammer worker model. 71
3.5 Results of varying β in ID strategy on three benchmark datasets with the one-coin worker model. 72
3.6 Results of varying β in ID strategy on three benchmark datasets with the one-coin worker model (malicious). 73
3.7 Results on two real-world datasets. 74

List of Tables

1.1 A toy example of task selection. 12

2.1 Descriptions of datasets used in this thesis. 37

Chapter 1

Introduction

The recent rise of *crowdsourcing* has enabled us to harness the *wisdom of crowds* to complete tasks that are difficult to tackle by machines alone. The focus of this doctoral thesis is to develop statistical approaches to learning from the wisdom of crowds. In this chapter, we state the motivation and objective of this research.

1.1 Learning from Labeled Data

Machine learning can be defined as a vast set of methods that can automatically learn from and make predictions or decisions on *data*. It sits at the intersection of *computer science* and *statistics*, and at the core of *artificial intellegence* and *data science* (Jordan and Mitchell, 2015).

1.1.1 Machine Learning and Big Data

Over the past decade, machine learning has been one of the most dramatically progressing technical fields. One of the drivers of this rapid growth is the well-known phenomenon which is often referred to as *big data*.

We are already in the era of big data. For example, Twitter users send nearly 350 thousand tweets per minute[1], the engine of Criteo (a large online display advertising company) analyzes more than 230 terabytes of data every day[2], and

[1] https://www.domo.com/learn/data-never-sleeps-3-0

[2] http://www.criteo.com/what-we-do/technology/

the warehouse of Facebook currently stores more than 300 petabytes of data, with an incoming daily rate of about 600 terabytes[3].

To handle large-scale data, scientists and engineers often seek help from machine learning methods to make predictions or decisions, or draw insights from the data. Broadly, machine learning methods are categorized into three paradigms:

- *Supervised learning*: The *learner* is presented with *labeled data* and makes predictions for *unseen* data.
- *Unsupervised learning*: The *learner* tries to learn hidden structural properties of *unlabeled data*.
- *Reinforcement learning*: The *learner* interacts with a dynamic *environment* and aims to maximize the cumulative *rewards* over a series of *actions*.

In practice, the most widely used machine learning methods are supervised learning methods (Hastie et al., 2009; Mohri et al., 2012; Murphy, 2012).

1.1.2 Supervised Learning

In supervised learning problems, the *learner* is presented with a collection of $(\boldsymbol{x}, y)$ pairs called *training set*, and the goal is to learn a *model* mapping from $\boldsymbol{x}$ to y and predict the corresponding y for any given $\boldsymbol{x}$. Here, $\boldsymbol{x}$ stands for a *data instance*, y represents the corresponding *label*, and we say the training set consists of *labeled data*.

In different real-world applications, a data instance $\boldsymbol{x}$ could be in various forms, such as an image, a text message, an audio clip, etc. The label y could be discrete or continuous. When y is discrete, the problem is called *classification*, and when y is continuous, the problem is called *regression*. A variant of regression is called *ordinal regression*, where the label space of y has some natural ordering, such as movie ratings (e.g. one to five stars).

In classification problems, the label y takes on one of C classes. When $C = 2$, it is called the *binary classification* problem; when $C > 2$, it is called the *multi-class classification* problem.

[3] https://code.facebook.com/posts/229861827208629

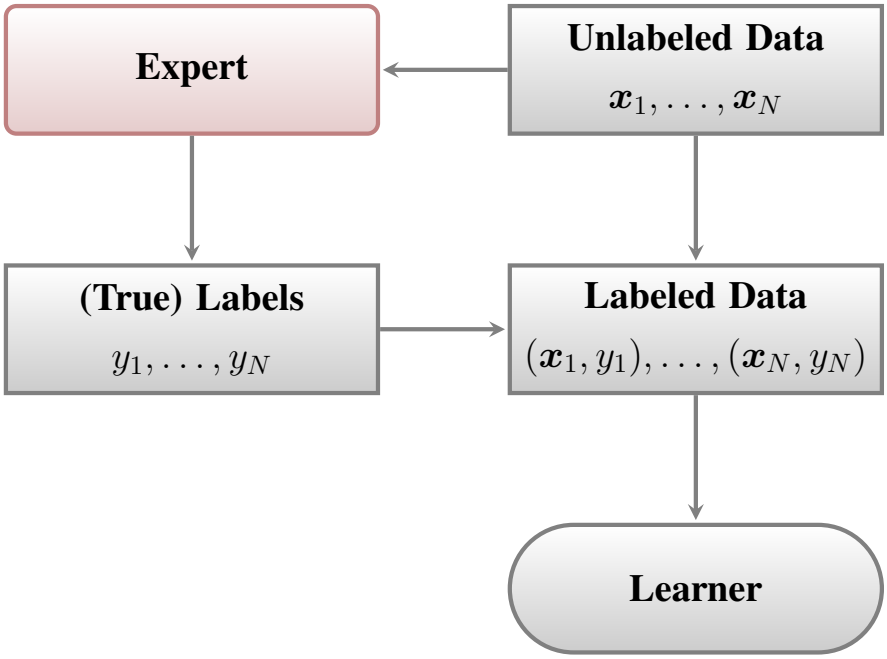


Figure 1.1: A simplified framework of supervised learning with expert labels. The total number of data instances is N. Labels from the expert are treated as true labels.

Now consider a small example of *image classification*, where we suppose the goal is to detect whether it contains a human or not in any given image. This is a binary classification problem, in which case the label of each image takes on one of two values (e.g. $+1$ and -1). In such a problem, usually we first have a set of images from the data source, and then each image is given a label (e.g. $+1$ or -1) by the *expert*. After we obtain the labeled images as the training set, we run the learner on this set to train our desired model. This model is often supposed to produce an output indicating whether it contains a human or not, given any image as an input.

Figure 1.1 illustrates a simplified framework of supervised learning with labels given by the expert. The total number of data instances

$$\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$$

is denoted by N. Originally, they are unlabeled data. The labels from the expert

are usually treated as true labels

$$y_1, \ldots, y_N.$$

Once a set of labeled data is available, the learner can be trained on this data set.

1.1.3 Labeled Data versus Unlabeled Data

Labeled data is essentially important to the learner in supervised learning. In some application cases, labels come at no cost. For example, in our email system, we mark spam flags on junk emails, and the system learns from these flags to better classify emails as spam or not. In this case, we provide spam flags (i.e. labels) for free, but in many other cases, labels need to be explicitly collected from humans at some cost. Traditionally, this can be done by hiring the expert (possibly a small group of experts) who can provide correct labels, like the case shown in Figure 1.1. However, this way of obtaining labeled data could be expensive and inefficient. For example, in some document classification problem, the expert has to label thousands of documents as "relevant" or "not relevant" to a certain topic, which could be tedious and time-consuming (Settles, 2009).

On the other hand, unlabeled data is usually redundant and far easier to obtain in practice (Zhou et al., 2004). However, it is difficult and hopeless for the learner to learn a mapping from $\boldsymbol{x}$ to y only with unlabeled data.

Then here is the question: is there any inexpensive and efficient way of obtaining the labels of unlabeled data? Fortunately, *crowdsourcing* (Howe, 2008) has given an answer to this question.

1.2 The Wisdom of Crowds

Crowdsourcing is a modern business term. Howe (2008) defined this term as follows:

> Crowdsourcing is the act of taking a task traditionally performed by a designated agent (such as an employee or a contractor) and outsourcing it by making an open call to an undefined but large group of people.

Recently, there is a rapidly increasing interest in utilizing crowdsourcing to produce labeled data for machine learning. Popular crowdsourcing platforms such as Amazon Mechanical Turk[4], CrowdFlower[5], Upwork[6] (formerly Elance-oDesk), Microworkers[7] and Lancers[8] provide the marketplaces where *requesters* can post labeling *tasks* and gather labels from crowds of *workers* in a short time at relatively low cost.

1.2.1 Crowdsourcing Marketplaces

In typical crowdsourcing marketplaces, there are mainly two roles: requesters and workers. A requester can post a group of micro-tasks, such as *image labeling* tasks in computer vision (Sorokin and Forsyth, 2008) and *recognizing textual entailment* tasks in natural language processing (Snow et al., 2008). Then the requester collects labels from workers and pays for them, possibly within a pre-fixed *budget* that she is willing to afford.

Workers in crowdsourcing marketplaces are also known as labelers or annotators. A worker can complete several labeling tasks and earn a tiny amount of money for every single task. For example, in a recognizing textual entailment task, a worker is presented a pair of sentences: *text* and *hypothesis*; then she is asked for a label, specifying whether the hypothesis can be implied by the information in the text; once she gives the required label, she gets paid 10 cents.

1.2.2 Crowds versus Experts

Compared with collecting labels from experts, crowdsourcing provides us with a far more efficient and inexpensive way of generating large amounts of labeled data. However, since most workers in crowds are non-experts, the crowdsourced labels typically contain a significant amount of errors (Kittur et al., 2008; Wais et al., 2010). This is a trade-off between cost/time and quality.

[4]https://www.mturk.com/mturk/
[5]http://www.crowdflower.com/
[6]https://www.upwork.com/
[7]https://microworkers.com/
[8]http://www.lancers.co.jp/

Fortunately, crowdsourcing services are still successful, because there exists a phenomenon of the *wisdom of crowds* (Surowiecki, 2005; Sheng et al., 2008; Liu et al., 2013):

> Properly combining a group of untrained people can be as good as the experts in many application domains.

Then how to properly combine the wisdom of crowds is what to investigate in crowdsourcing research.

1.2.3 Crowdsourcing Research

Now we describe the general problem setting in crowdsourcing research.

General Problem Setting

Figure 1.2 shows a simplified framework of supervised learning with crowd labels. Instead of hiring the expert, the requester asks K semi-skilled workers to perform labeling tasks. Here, the term "task" refers to the process of labeling a data instance. In the case shown in Figure 1.2, we have N labeling tasks, i.e., there are N data instances that need to be labeled.

Since workers are non-experts, the labels collected from them are usually (highly) noisy. A common solution is to collect labels from multiple workers for each task, which is often referred to as *repeated labeling* (Ipeirotis et al., 2013), and then aggregate these labels in a *reasonable* way. As shown in Figure 1.2, the labels of N tasks from K workers form a $N \times K$ matrix

$$
\begin{array}{cc}
 & \text{Worker} \\
 & \begin{array}{cccccc} 1 & 2 & \cdots & j & \cdots & K \end{array} \\
\text{Task} & \begin{array}{c} 1 \\ 2 \\ \vdots \\ i \\ \vdots \\ N \end{array}
\begin{pmatrix}
y_{1,1} & y_{1,2} & \cdots & y_{1,j} & \cdots & y_{1,K} \\
y_{2,1} & y_{2,2} & \cdots & y_{2,j} & \cdots & y_{2,K} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
y_{i,1} & y_{i,2} & \cdots & y_{i,j} & \cdots & y_{i,K} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
y_{N,1} & y_{N,2} & \cdots & y_{N,j} & \cdots & y_{N,K}
\end{pmatrix},
\end{array}
$$

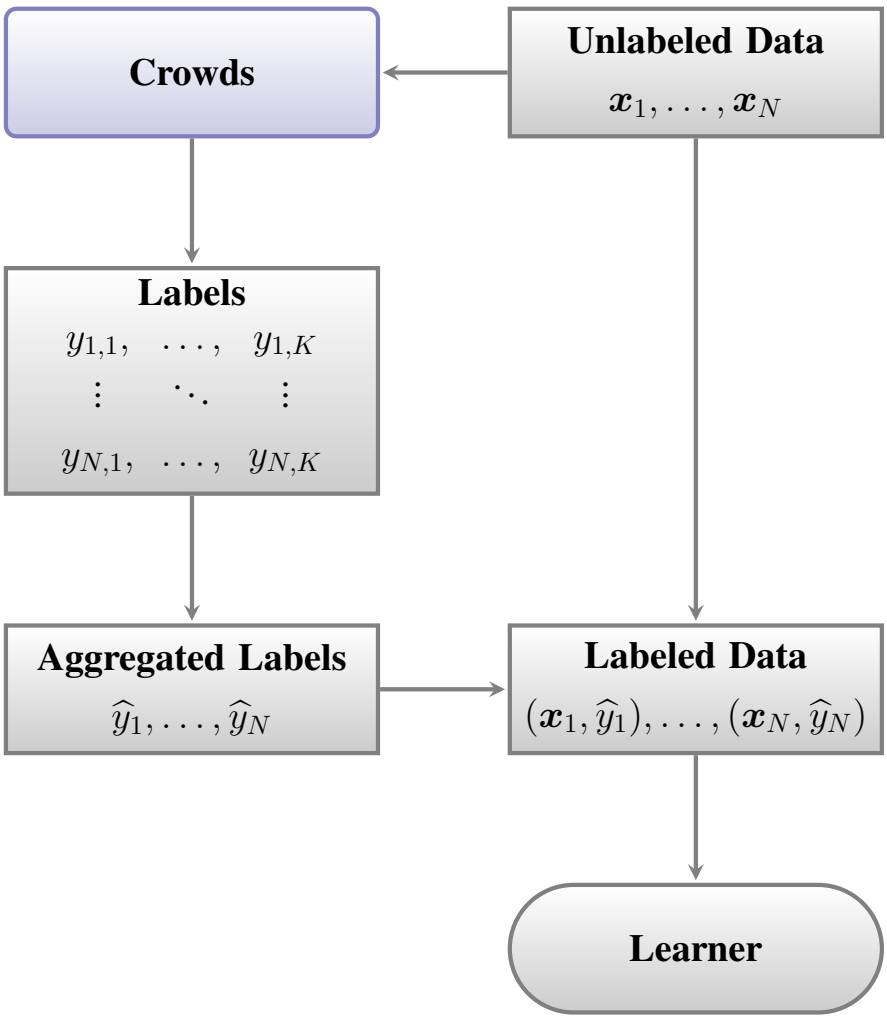


Figure 1.2: A simplified framework of supervised learning with crowd labels. The total number of data instances is N, and the crowds comprise K workers. Each data instance is labeled by multiple workers, and its true label is estimated by aggregating worker labels.

where the entry $y_{i,j}$ represents the label of task i from worker j. Note that some entries of the matrix might be missing, which implies that the requester does not necessarily collect labels from *all* workers for each task.

After label collection, the requester performs label aggregation to estimate the true label for each task (e.g. $\widehat{y}_i$ is the estimate of the true label y_i). Then the aggregated labels are used for learning.

It is worth noting that some crowdsourcing methods (e.g. Raykar et al., 2010; Yan et al., 2010) aim to learn a classifier. They directly feed the collected labels (i.e. the $N \times K$ matrix) to the learner, and merge the label aggregation phase into the learning phase, instead of separating them into two explicit phases (as shown in Figure 1.2). This is a straightforward way when the learning problem is specified. However, in most of cases, crowdsourcing is used for producing large-scale labeled data as knowledge bases, rather than learning a specific classifier. A well-known example is ImageNet[9], which is a large-scale image database (Deng et al., 2009). It is a useful resource for computer vision researchers to develop new algorithms for various applications. One important step of constructing such a large database is to ask humans to label whether it contains a certain object for each image. This is achieved by using the crowdsourcing service of Amazon Mechanical Turk. Finally, the collected labels are aggregated to produce the large labeled dataset.

Label Aggregation

Now let us consider the question: what could be a *reasonable* way of aggregating labels? A naive solution is to utilize *majority voting*: for any task, *equally* treat each worker and take the majority vote of their labels as the estimate of the true label. However, the labeling performance of different workers can vary widely (Raykar et al., 2010). Some workers might be more *reliable* than others in terms of giving correct labels. Hence, a more reasonable way of aggregating labels is to give workers different *weights* based on their *reliability*. This is known as *weighted voting*.

Weighted voting is often explicitly or implicitly used for label aggregation in crowdsourcing. However, as well as the true labels, the weights (i.e. workers'

[9] http://www.image-net.org/

reliability) are also *unknown*. Hence, the primary interests of crowdsourcing research are in learning workers' reliability and aggregating noisy labels (Dawid and Skene, 1979; Smyth et al., 1995; Whitehill et al., 2009; Ipeirotis et al., 2010; Raykar et al., 2010; Welinder and Perona, 2010; Welinder et al., 2010; Yan et al., 2010, 2011; Wauthier and Jordan, 2011; Kajino et al., 2012; Liu et al., 2012; Zhou et al., 2012, 2014).

Offline-Style Methods

Many crowdsourcing methods are *offline-style*, running on the *collected* labeled data *after* the entire process of label collection. We refer to these methods as *offline-style methods* in this thesis.

To the best of our knowledge, the earliest seminal offline-style method is *Dawid-Skene's method* (Dawid and Skene, 1979). Although the term of crowdsourcing did not exist yet at the time it was proposed, Dawid-Skene's method coped with a similar problem to crowdsourcing. Specifically, a model was presented to classify the *unknown* true conditions of patients by aggregating the diagnoses from multiple clinicians. The error rate of each clinician is captured by a confusion matrix. This confusion matrix is also *unknown*, and can be interpreted as the *reliability* of the clinician.

Technically, Dawid-Skene's method employed the *Expectation-Maximization* (EM) (Dempster et al., 1977) algorithm to jointly estimate the reliability of clinicians and the true conditions of patients. The simplified iterative procedure of Dawid-Skene's method is described as follows:

(i) Estimate the true condition of each patient by weighted voting, using current estimates of clinicians' reliability as weights.

(ii) Estimate clinicians' reliability, given current estimates of true conditions of patients.

(iii) Repeat steps (i) and (ii) until the estimates converge.

In Step (i) of the first-round iteration, the true condition of each patient can be initialized by using majority voting instead of weighted voting. This is because we have no estimates of clinicians' reliability as weights at the very beginning.

Based on the idea of Dawid-Skene's method, significant efforts have been made to study more sophisticated models for crowdsourcing. For example, Raykar et al. (2010) focused on learning a classifier, with estimating workers' reliability and true labels as a byproduct of their proposed method; Welinder et al. (2010) incorporated more factors into the model, such as task difficulty, worker biases, etc.; Zhou et al. (2012) also took task confusability into account as well as workers' reliability in the *multi-class* setting, and later extended their work to the *ordinal* setting (Zhou et al., 2014).

Offline-style methods perform well in practice. However, as discussed above, they usually run on the *collected* labeled data after the whole label collection process, but do not focus on *how to collect* these labels. From the perspective of a requester, it is necessary to consider how to *adaptively* collect labels from workers, for the purpose of intelligently using the total budget. This motivates us to consider another important problem in crowdsourcing research, which is called *task-worker assignment* or task routing (Law and von Ahn, 2011). The task-worker assignment problem in crowdsourcing falls into the broad category of *online decision making* (Slivkins and Vaughan, 2014).

1.3 Online Decision Making in Crowdsourcing

Online decision making in crowdsourcing consists of many challenging problems, concerning the repeated decisions that the requesters or systems make on assigning tasks to specific workers, filtering out unreliable workers, setting dynamic prices of tasks, or others (Slivkins and Vaughan, 2014). In this section, we introduce two important problems: *task-worker assignment* and *dynamic procurement*.

1.3.1 Task-Worker Assignment

Generally, there are two classes of crowdsourcing marketplaces: *pull* and *push*. In *pull* marketplaces, the system takes a passive role and only sets up the environment for workers to find tasks themselves (e.g. Galaxy Zoo[10]); whereas in *push* marketplaces, the system takes complete control over which tasks are assigned to

[10]http://www.galaxyzoo.org/

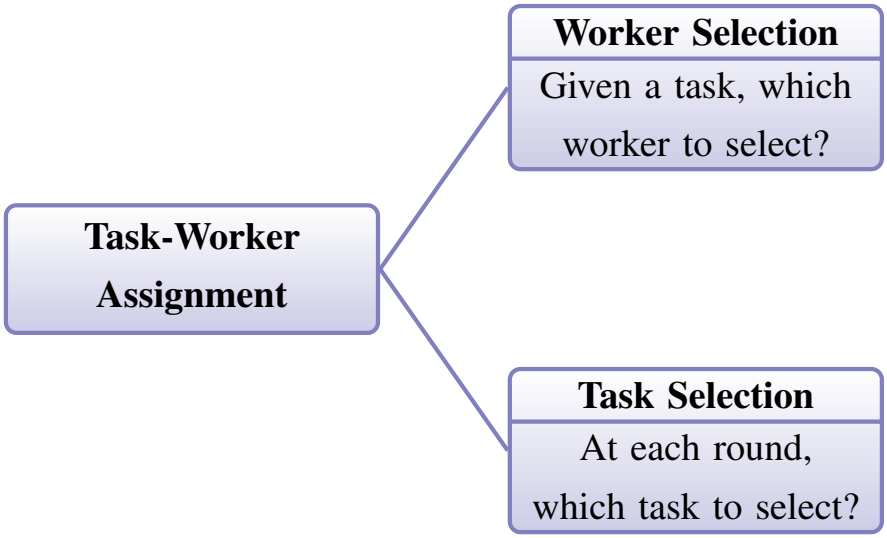


Figure 1.3: The task-worker assignment problem in crowdsourcing. It contains two aspects: worker selection and task selection.

whom (e.g. data annotation team in Microsoft Bing group (Chen et al., 2015)) (Law and von Ahn, 2011). In this thesis, we focus on the *task-worker assignment* problem in *push* marketplaces.

The goal of task-worker assignment is to collect and aggregate reliable labels from workers, by means of reasonably determining *which task* is assigned to *which worker* in a sequence of trials.

The task-worker assignment problem naturally involves two aspects: *worker selection* and *task selection*, as illustrated in Figure 1.3.

Worker Selection

Worker selection is an important part of the task-worker assignment problem, which is aimed at selecting *reliable* workers given any task.

Suppose we have selected a task, and then which worker to select? On one hand, selecting a reliable worker is always favored; on the other hand, workers' reliability is unknown and is what to estimate over time. This is the well-known phenomenon called the *exploration-exploitation trade-off* (Thompson, 1933; Robbins, 1952; Bubeck and Cesa-Bianchi, 2012). Here, *exploration* means gathering

Task	Label 1	Label 2	Label 3
1	+1	+1	?
2	-1	+1	?
3	-1	-1	?

Table 1.1: A toy example of task selection.

more information about which workers are reliable, while *exploitation* is to select reliable workers based on the information at hand.

The existing methods for worker selection in crowdsourcing adopt different strategies to deal with the trade-off between exploration and exploitation (Donmez et al., 2009; Chen et al., 2013; Ertekin et al., 2014). In contrast to *offline-style* methods, we refer to these methods as *online-style* methods, as they simultaneously learn workers' reliability and collect labels in an *online* (i.e. sequential) manner. Specifically, *IEThresh* (Donmez et al., 2009) builds upon *interval estimation* (Kaelbling, 1990), and sequentially selects workers whose current *upper confidence intervals* are higher than a pre-fixed threshold; similarly, *CrowdSense* (Ertekin et al., 2014) sequentially samples worker subsets according to a special exploration/exploitation criterion for measuring workers' current labeling performances; *OptKG* (Chen et al., 2013) formulates the problem as a Bayesian Markov decision process (MDP), and obtains a computationally efficient approximate policy for collecting labels from workers. More details of these existing methods are further described in Section 2.4.

Task Selection

Task selection is the other involved aspect of the task-worker assignment problem. The purpose of task selection is to select an *appropriate* task at each round.

What is an *appropriate* task? A possible measure for this could be *uncertainty*. Table 1.1 shows a toy example of task selection according to uncertainty. Suppose we have already collected two labels for each of the three tasks. Now we are going to collect one more label for one task. Then which task to select? In this toy example, both Task 1 and Task 3 have consistent labels respectively (i.e. all positive and all negative), while Task 2 has one positive and one negtive. Suppose we

use majority voting for label aggregation. Then intuitively we are more uncertain about the aggregated label of Task 2 than those of the other two tasks. Thus we select Task 2 to collect a label for it.

A good strategy of task selection can help us efficiently use the budget when assigning tasks to workers. For example, if we are already confident in the aggregated label of one certain task, we should not require more labels for this task, otherwise it would be a waste of budget. As well as worker selection, task selection is also worth further investigating as the other important part of the task-worker assignment problem.

1.3.2 Dynamic Procurement

Dynamic procurement focuses on making repeated decisions on setting prices of tasks (Slivkins and Vaughan, 2014). Here, the price means a tiny amount of money (e.g. \$0.01) that a worker obtains after completing a task. Unlike the standard setting of crowdsourcing, where the price of each task is fixed, dynamic procurement deals with *changeable* prices of tasks in crowdsourcing.

Slivkins and Vaughan (2014) described the basic model of dynamic procurement as follows: at each round, the requester posts a task and sets its price, and then waits until some worker takes this task or a pre-specified time runs out. The goal is to maximize the total number of completed tasks with the constrains on budget and waiting time.

Some researchers have made considerable efforts on this problem (Badanidiyuru et al., 2012, 2013a,b). However, Slivkins and Vaughan (2014) pointed out that whether the obtained results are optimal is not clear.

Dynamic procurement is also a challenging problem, together with task-worker assignment. However, more details of this problem are beyond the scope of this thesis. It is considered as a promising direction of our future work.

1.4 Contributions

This thesis contributes to developing statistical learning approaches for the task-worker assignment problem in crowdsourcing. This problem naturally involves

worker selection and task selection. We independently study and combine them. In this section, we briefly explain our contributions.

1.4.1 Worker Selection in Crowdsourcing

Worker selection is an important part of the task-worker assignment problem in crowdsourcing. A good strategy of worker selection can help the requester wisely use the budget.

Heterogeneous Crowdsourcing

In recent crowdsourcing marketplaces, the *heterogeneity* of labeling tasks is increasing. We refer to this setting as *heterogeneous* crowdsourcing. In such a scenario, a worker may be reliable at only a subset of tasks with a certain *context*. For example, when completing *name entity recognition* tasks (Finin et al., 2010; Ritter et al., 2011) in natural language processing, a worker may be good at recognizing names of sports teams, but not be familiar with cosmetics brands. Here, "sports" or "cosmetics" is called a context. A worker might have various reliability depending on different contexts, due to their individual knowledge domains. Hence, it is reasonable to model context-dependent reliability for workers in heterogeneous crowdsourcing. However, the existing online-style methods described in Section 1.3.1 are designed for the *homogeneous* setting, where multiple contexts (i.e. there exists more than one context) and context-dependent reliability of workers are not taken into account.

Bandit Problems

As introduced in Section 1.3.1, there exists a trade-off between exploration and exploitation in worker selection. To address this trade-off, we formulate our worker selection problem as a *multi-armed bandit* problem, more specifically, a *contextual bandit* problem (Langford and Zhang, 2008; Li et al., 2010; Bubeck and Cesa-Bianchi, 2012).

Multi-armed bandit problems (Auer et al., 2002a,b; Cesa-Bianchi and Lugosi, 2006; Bubeck and Cesa-Bianchi, 2012) are basic examples of online decision making with *limited feedback* and can naturally handle the exploration-

exploitation trade-off. At each round, we allocate one unit of budget to one of a set of actions and obtain some observable reward (loss) given by the environment. The goal is to maximize/minimize the cumulative reward/loss in the whole allocation sequence. To achieve this goal, we must balance the exploitation of actions that are good in the past and the exploration of actions that may be better in the future.

A practical extension of the basic bandit problem is the contextual bandit problem, where each round is marked by a context from a given set. Then the interest is in finding good mappings from contexts to actions rather than identifying good actions.

Bandit-Based Task Assignment

We propose a new method called *bandit-based task assignment* (BBTA) for worker selection in heterogeneous crowdsourcing. BBTA is a contextual bandit formulation, where a *context* can be interpreted as the type or required skill of a task, and each arm of the bandit represents a worker. The feedback after pulling an arm depends on the current context. This corresponds to the context-dependent reliability of workers. BBTA focuses on the strategy of worker selection in heterogeneous crowdsourcing. That is, given a task with a certain context, BBTA tries to select a suitable worker who tends to be good at this task.

We provide theoretical analysis for BBTA, indicating that the performance of our strategy converges to that of the optimal one as the budget goes to infinity.

We also conduct extensive experiments on both benchmark data and real data, and demonstrate that BBTA can outperform other online-style methods in heterogeneous crowdsourcing.

1.4.2 Task Selection in Crowdsourcing

Since task selection is the other important aspect of the task-worker assignment problem, we further investigate the strategies of task selection in heterogeneous crowdsourcing. In particular, we extend BBTA by using different strategies of selecting tasks. The idea of these strategies is borrowed from *query strategies* in *active learning* (Settles, 2009).

Usually, query strategies in active learning are designed for *probabilistic* inference. In this thesis, we introduce these strategies into our task-worker assignment problem, where the weighted voting mechanism is *non-probabilistic*. Although we do not have *probabilities* of labels as in probabilistic inference, we instead define a *confidence score* for each task to describe how confident we are in the aggregated label of this task.

In the original BBTA, we simply use the task selection strategy directly from weighted voting. To investigate this problem more, we further develop several strategies for task selection and embed them into the original BBTA. One of these strategies called *margin sampling* is equivalent to the original one used in BBTA.

We experimentally evaluate the proposed strategies for BBTA, and demonstrate that the performance of BBTA can be further improved by adopting appropriate task selection strategies such as the *least confidence* strategy.

1.5 Organization

This thesis is comprised of four chapters. Figure 1.4 shows the organization of this thesis.

In Chapter 1, we introduced important concepts closely related to our research, including machine learning, big data, supervised learning, labeled data in Section 1.1, crowdsourcing, label aggregation, workers' reliability, weighted voting in Section 1.2, and push marketplaces, task-worker assignment, worker selection, task selection in Section 1.3.

Chapter 2 covers our first contribution on worker selection in crowdsourcing. In section 2.1, we introduce the background and motivation of worker selection. Section 2.2 describes our proposed bandit-based task assignment (BBTA) method for worker selection in heterogeneous crowdsourcing. Then we theoretically investigate its regret bounds in Section 2.3. For the purpose of comparison, we look into the details of the existing online-style methods for crowdsourcing in Section 2.4 and experimentally evaluate them together with the proposed method in Section 2.5. Finally, the proof of theoretical results is provided in Section 2.6.

In Chapter 3, we introduce our second contribution on task selection in crowdsourcing. Section 3.1 presents the motivation of task selection. In Section 3.2, we

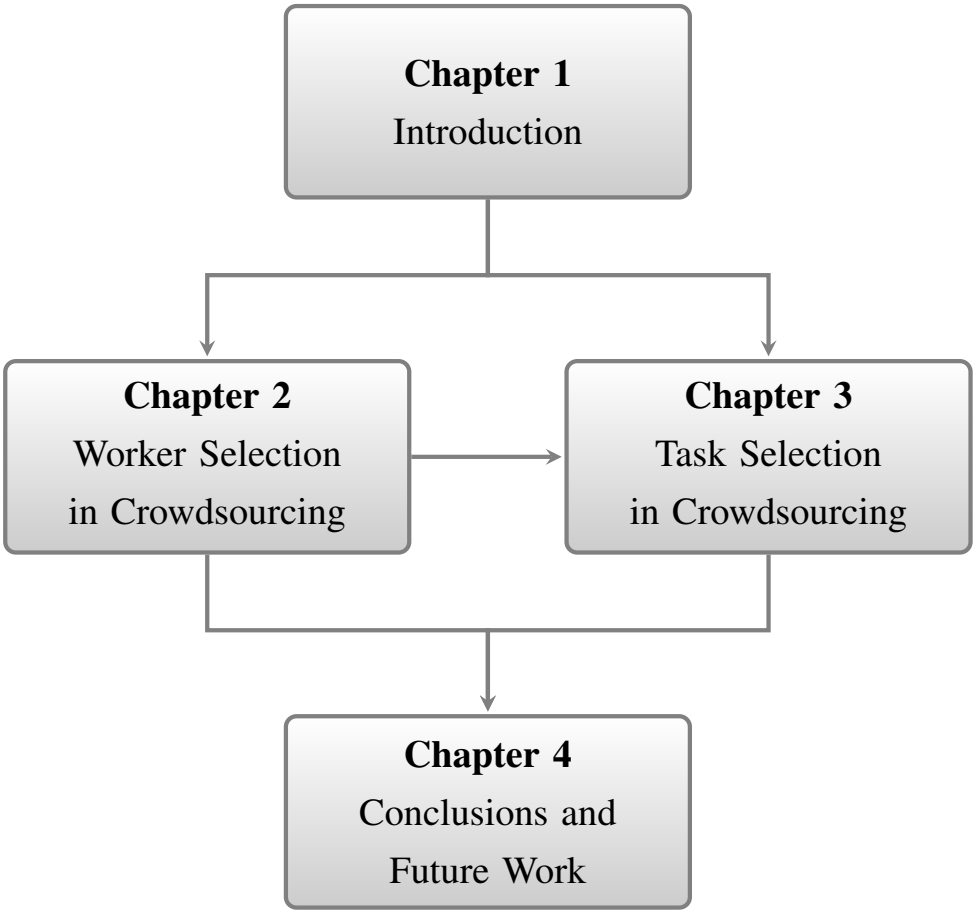


Figure 1.4: Organization of this thesis.

develop the strategies of task selection for crowdsourcing. Finally, Section 3.3 shows the experimental results of embedding different task selection strategies into BBTA.

In the end, we present conclusions and future directions in Chapter 4.

Chapter 2

Worker Selection in Crowdsourcing

Worker selection is an important part of the task-worker assignment problem in crowdsourcing. In this chapter, we propose the *bandit-based task assignment* (BBTA) method for worker selection in *heterogeneous* crowdsourcing. We also theoretically investigate the regret bounds for BBTA, and experimentally demonstrate its practical usefulness.

2.1 Introduction

The quantity and quality of labeled data significantly affect the performance of machine learning algorithms. However, collecting reliable labels from experts is usually expensive and time-consuming. The recent emergence of *crowdsourcing* services such as *Amazon Mechanical Turk*[1] (MTurk) enables us to cheaply collect huge quantities of labeled data from crowds of workers for many machine learning tasks, e.g., natural language processing (Snow et al., 2008) and computer vision (Welinder and Perona, 2010). In a crowdsourcing system, a requester asks workers to complete a set of labeling tasks by paying them a tiny amount of money for each label.

The primary interest of crowdsourcing research has been how to cope with different reliability of workers and aggregate the collected noisy labels (Dawid and Skene, 1979; Smyth et al., 1995; Whitehill et al., 2009; Ipeirotis et al., 2010;

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

Raykar et al., 2010; Welinder et al., 2010; Yan et al., 2010, 2011; Wauthier and Jordan, 2011; Kajino et al., 2012; Liu et al., 2012; Zhou et al., 2012, 2014). Usually, the *weighted voting* mechanism is implicitly or explicitly utilized for label aggregation, with workers' reliability as weights. Many existing methods use the *Expectation-Maximization* (EM) algorithm (Dempster et al., 1977) on collected labeled data to jointly estimate workers' reliability and true labels. However, how to *adaptively* collect these labels is often neglected. Since the total budget for a requester to pay the workers is usually limited, it is necessary to consider how to intelligently use the budget to assign tasks to workers.

This leads another important line of crowdsourcing research, which is called the task routing or *task-worker assignment* problem. There are two classes of crowdsourcing marketplaces: *push* and *pull*. While in *pull* marketplaces the system takes a passive role and only sets up the environment for workers to find tasks themselves, in *push* marketplaces the system takes complete control over which tasks are assigned to whom (Law and von Ahn, 2011). In this chapter we focus on *push* marketplaces.

Worker selection is an important part of the task-worker assignment problem. It is aimed at selecting a suitable worker given any task. Most of the existing methods for worker selection run in an online mode, and we refer to these methods as *online-style* methods, which often simultaneously learn workers' reliability and collect labels (Donmez et al., 2009; Chen et al., 2013; Ertekin et al., 2014). To deal with the exploration (i.e. learning which workers are reliable) and exploitation (i.e. selecting the workers considered to be reliable) trade-off in worker selection, *IEThresh* (Donmez et al., 2009) and *CrowdSense* (Ertekin et al., 2014) dynamically sample worker subsets according to workers' labeling performances. However, this is not enough in recent *heterogeneous* crowdsourcing where a worker may be reliable at only a subset of tasks with a certain type. For example, the workers considered to be good at previous tasks in IEThresh and CrowdSense may be bad at next ones. Therefore, it is more reasonable to model context-dependent reliability for workers in heterogeneous crowdsourcing. Here, context can be interpreted as the type for required skill of a certain task in heterogeneous crowdsourcing. Another issue in IEThresh and CrowdSense is that the budget is not pre-fixed. That is, the requester will not know the total budget until

the whole assignment process is over. This makes those two methods not so practical for crowdsourcing. *OptKG* (Chen et al., 2013) runs within a pre-fixed budget and formulates the problem as a Markov decision process (MDP). However, it is difficult to give theoretical guarantees for OptKG when heterogeneous workers are involved.

In contrast to the term "heterogeneous crowdsourcing", we refer to the setting where the existence of multiple contexts (i.e. there exists more than one context) is not taken into account as "homogeneous crowdsourcing".

In recent crowdsourcing marketplaces, as the *heterogeneity* of tasks is increasing, many researchers started to focus on heterogeneous crowdsourcing. Goel et al. (2014) studied the problem of incentive-compatible mechanism design for heterogeneous markets. The goal is to properly price the tasks for worker trustfulness and maximize the requester utility with the financial constraint. Ho and Vaughan (2012) and Ho et al. (2013) studied the problem of task-worker assignment in heterogeneous crowdsourcing. However, it is another variant of problem setting, where workers arrive online and the requester must assign a task (or sequence of tasks) to each new worker as she arrives (Slivkins and Vaughan, 2014). While in our problem setting, the requester completely controls which task to pick and which worker to select at each round.

From a technical perspective, the most similar problem setting to ours is that of OptKG, where we can determine a task-worker pair at each step. For the purpose of extensive comparison, we also include two heuristic methods IEThresh and CrowdSense as well as OptKG in our experiments. These three online-style methods are further detailed in Section 2.4.

In this chapter, we propose a *contextual bandit* formulation for worker selection in heterogeneous crowdsourcing. Our method models context-dependent reliability for workers by using *weight*, which depends on the *context* of a certain task. The term *context* here can be interpreted as the type or required skill of a task. For label aggregation, we adopt *weighted voting*, which is a common solution used for aggregating noisy labels in crowdsourcing. Our method consists of two phases: the *pure exploration* phase and the *adaptive assignment* phase. In the pure exploration phase, we explore workers' reliability in a batch mode and initialize their weights as the input of the adaptive assignment phase. On the other

hand, the adaptive assignment phase performs in an online mode. It includes a bandit-based strategy, where we sequentially select a worker for a given labeling task with the help of the *exponential weighting scheme* (Auer et al., 2002b; Stoltz, 2005; Cesa-Bianchi and Lugosi, 2006; Arora et al., 2012). This scheme is considered as a standard tool for bandit problems. The whole method runs within a limited budget. Moreover, we also investigate the regret bounds of our strategy of worker selection theoretically, and demonstrate its practical usefulness experimentally.

2.2 Bandit-Based Task Assignment

In this section, we describe our proposed bandit-based task assignment (BBTA) method.

2.2.1 Problem Formulation

Suppose we have N unlabeled tasks with indices $\mathcal{I} = \{1, \ldots, N\}$, each of which is characterized by a context s from a given context set $\mathcal{S}$, where $|\mathcal{S}| = S$. Let $\{y_i^*\}_{i=1}^N$ be the unknown true labels of tasks, where $y_i^* \in \{-1, 1\}$. Each time given a task, we ask one worker from a pool of K workers for a (possibly noisy) label, consuming one unit of the total budget T. Our goal is to find suitable task-worker assignment to collect as many reliable labels as possible within the limited budget T. Finally, we aggregate the collected labels to estimate the true labels $\{y_i^*\}_{i=1}^N$.

Let $y_{i,j}$ be the individual label of task i (with context s_i) given by worker j. If the label is missing, we set $y_{i,j} = 0$. For simplicity, we omit the subscript i of context s_i, and refer to the context of the current task as s. We denote the weight of worker j for context s by w_j^s (> 0), corresponding to the context-dependent reliability of worker j. Note that w_j^s is what we learn dynamically in the method. Then an estimate of the true label y_i^* is calculated by the weighted voting mechanism as

$$\widehat{y}_i = \mathrm{sign}(\overline{y}_i), \text{ where } \overline{y}_i = \frac{\sum_{j=1}^K w_j^s y_{i,j}}{\sum_{j'=1}^K w_{j'}^s}. \tag{2.1}$$

Our proposed method consists of the *pure exploration* phase and the *adaptive assignment* phase. The pseudo code is given in Algorithm 1, and the details are explained below.

2.2.2 Pure Exploration Phase

Pure exploration performs in a batch mode, and the purpose is to know which workers are reliable at which labeling tasks. To this end, we pick N' tasks for each of S distinct contexts ($SN' \ll N$) and let *all* K workers label them (Line 11-12 in Algorithm 1). We denote the index set of N' tasks with context s as $\mathcal{I}_1^s$ in this phase.

Since we have no prior knowledge of workers' reliability in this phase, we give all of them the same weight when aggregating their labels (equivalent to majority voting):

$$\widehat{y}_i = \mathrm{sign}(\overline{y}_i), \text{ where } \overline{y}_i = \frac{1}{K}\sum_{j=1}^{K} y_{i,j}. \tag{2.2}$$

In the standard crowdsourcing scenario, all of the true labels are usually unknown. As in many other crowdsourcing methods, we have the prior belief that most workers perform reasonably well. To evaluate an individual label $y_{i,j}$, using the weighted vote $\widehat{y}_i$ is a common solution (Donmez et al., 2009; Ertekin et al., 2014). We denote the cumulative loss by $L_{j,0}^s$ and initialize it for the next phase as

$$L_{j,0}^s = \sum_{i \in \mathcal{I}_1^s} \mathbb{1}_{y_{i,j} \neq \widehat{y}_i}, \text{ for } j = 1, \ldots, K \text{ and } s \in \mathcal{S},$$

where $\mathbb{1}_\pi$ denotes the indicator function that outputs 1 if condition π holds and 0 otherwise. This means that when a worker gives an individual label $y_{i,j}$ inconsistent or consistent with the weighted vote $\widehat{y}_i$, this worker suffers a loss 1 or 0. It is easy to see that cumulative losses correspond to workers' reliability. They are used for calculating workers' weights in the next phase. The budget for the next phase is

$$T_2 = T - T_1, \text{ where } T_1 = SKN' < T.$$

T_1 represents the budget consumed in this phase.

Algorithm 1 Bandit-Based Task Assignment (BBTA)

Input:

N: The number of tasks, each of which has a context s from $\mathcal{S}$

K: The number of workers

T: The total budget

N': The number of tasks for each context in the pure exploration phase

Initialization:

$y_{i,j} = 0$ for $i = 1, \ldots, N$ and $j = 1, \ldots, K$.

$|\overline{y}_i| = 0$ for $i = 1, \ldots, N$.

$t^s = 0$ for $s \in \mathcal{S}$.

Pure Exploration Phase

Pick N' tasks for each of S distinct contexts.

Collect labels from *all* workers for these tasks and use Eqn. 2.2 for label aggregation.

Calculate cumulative losses for $j = 1, \ldots, K$ and $s \in \mathcal{S}$:

$$L^s_{j,0} = \sum_{i \in \mathcal{I}^s_1} \mathbb{1}_{y_{i,j} \neq \widehat{y}_i}.$$

Set the budget as $T_2 = T - SKN'$ and the index set of available tasks as $\mathcal{I}_2 = \mathcal{I} \setminus \bigcup_{s \in \mathcal{S}} \mathcal{I}^s_1$ for the adaptive assignment phase.

15: # Adaptive Assignment Phase
16: **for** $t = 1$ **to** T_2 **do**
17: Pick the task with the lowest confidence score $i_t = \arg\min_{i \in \mathcal{I}_2} |\overline{y}_i|$.
18: Observe its context s, and update $t^s \leftarrow t^s + 1$.
19: Set $\eta^s_{t^s} = \sqrt{\frac{\ln K}{t^s K}}$, and calculate the weights for $j = 1, \ldots, K$:

$$w^s_{j,t^s} = \exp(-\eta^s_{t^s} L^s_{j,t^s-1}).$$

20: Draw the worker j_t from the distribution $(p_{1,t}, \ldots, p_{K,t})$, where

$$p_{j,t} = \frac{w^s_{j,t^s}}{\sum_{j'=1}^{K} w^s_{j',t^s}}.$$

21: Obtain label y_{i_t,j_t} and calculate $\widehat{y}_{i_t}$ by the weighted voting mechanism (Eqn. 2.1).
22: Set $l_{j_t,t} = \mathbb{1}_{\widehat{y}_{i_t} \neq y_{i_t,j_t}}$, and for $j = 1, \ldots, K$ calculate $\widetilde{l}_{j,t} = \frac{l_{j,t}}{p_{j,t}} \mathbb{1}_{j=j_t}$ and update $L^s_{j,t^s} = L^s_{j,t^s-1} + \widetilde{l}_{j,t}$.
23: Update confidence scores $|\overline{y}_i|$ of all tasks with context s by using Eqn. 2.1.
24: **if** task i_t is already labeled by all workers **then**
25: $\mathcal{I}_2 \leftarrow \mathcal{I}_2 \setminus \{i_t\}$.
26: **end if**
27: **end for**
28: **Output:** $\widehat{y}_i = \text{sign}(\overline{y}_i)$ for $i = 1, \ldots, N$

2.2.3 Adaptive Assignment Phase

In the adaptive assignment phase, task-worker assignment is determined for the remaining $N - SN'$ tasks in an online mode within the remaining budget T_2. At each round t of this phase, to determine a task-worker pair, we need to further consider which task to pick and which worker to select for this task.

Confidence Score

According to the weighted voting mechanism (Eqn. 2.1), the magnitude $|\overline{y}_i|$ ($\in [0, 1]$) corresponds to the *confidence score* of $\widehat{y}_i$. For task i, the confidence score $|\overline{y}_i|$ will be 1 if and only if we have collected labels from *all* workers and *all* of them are consistent. On the other hand, when the sum of (normalized) weights for positive labels is equal to that for negative labels, or we have no labels for task i, the confidence score $|\overline{y}_i|$ is 0. That is, we are not confident in the aggregated label (i.e. the confidence score is low) because the collected labels are significantly inconsistent, or insufficient, or both. If the confidence score of a task is lower than those of others, collecting more labels for this task is a reasonable solution. Thus we pick task i_t with the lowest confidence score as the next one to label (Line 17 in Algorithm 1):

$$i_t = \arg\min_{i \in \mathcal{I}_2} |\overline{y}_i| \,, \tag{2.3}$$

where $\mathcal{I}_2$ is the index set of current available tasks in this phase. This idea is similar to *uncertainty sampling* (Lewis and Gale, 1994) in active learning.

Bandit Formulation

Given the picked task, selecting a worker reliable at this task is always favored. On the other hand, workers' reliability is what we are dynamically learning in the method. There exists a trade-off between exploration (i.e. learning which worker is reliable) and exploitation (i.e. selecting the worker considered to be reliable) in worker selection. To address this trade-off, we formulate our worker selection problem as a *multi-armed bandit* problem, more specifically, a *contextual bandit* problem (Langford and Zhang, 2008; Li et al., 2010; Bubeck and Cesa-Bianchi, 2012).

Multi-armed bandit problems (Auer et al., 2002a,b; Cesa-Bianchi and Lugosi, 2006; Bubeck and Cesa-Bianchi, 2012) are basic examples of sequential decision making with *limited feedback* and can naturally handle the exploration-exploitation trade-off. At each round, we allocate one unit of budget to one of a set of actions and obtain some observable reward (loss) given by the environment. The goal is to maximize/minimize the cumulative reward/loss in the whole allocation sequence. To achieve this goal, we must balance the exploitation of actions that are good in the past and the exploration of actions that may be better in the future. A practical extension of the basic bandit problem is the contextual bandit problem, where each step is marked by a context from a given set. Then the interest is in finding good mappings from contexts to actions rather than identifying good actions.

In most contextual bandits, the context (also known as side information) is provided as a feature vector which influences the reward/loss at each step, whereas in the heterogeneous crowdsourcing setting, we consider the simple situation of contextual bandits, where the context is marked by some discrete value, corresponding to the task type. Then our interest is to find a good mapping from task types to appropriate workers rather than the best worker for all tasks. Note that the task types are observable and provided by the environment.

The adaptive assignment phase includes our strategy of worker selection, where selecting a worker corresponds to taking an action. The objective is to collect reliable labels via the strategy of adaptively selecting workers from the worker pool.

We further assume the scheme that each worker adopts to assign labels as a *black box*. Then the labeled status of all tasks could abruptly change over time. This means the exact sequence of tasks in the adaptive assignment phase is unpredictable, given that we calculate confidence scores for all tasks based on their labeled status. Thus we consider the task sequence as the external information in this phase.

Given the task with context s, we calculate the weights (s-dependent) of workers as follows (Line 19 in Algorithm 1):

$$w^s_{j,t^s} = \exp(-\eta^s_{t^s} L^s_{j,t^s-1}), \text{ for } j = 1, \ldots, K,$$

where t^s is the appearance count of context s and $\eta^s_{t^s}$ is the learning rate related to t^s. This calculation of weights by using cumulative losses is due to the *exponen-*

tial weighting scheme (Auer et al., 2002b; Stoltz, 2005; Cesa-Bianchi and Lugosi, 2006; Arora et al., 2012), which is a standard tool for sequential decision making under adversarial assumptions. Following the exponential weighting scheme, we then select a worker j_t from the discrete probability distribution on workers with each probability $p_{j,t}$ proportional to the weight w^s_{j,t^s} (Line 20 in Algorithm 1). Since workers' cumulative losses do not greatly vary at the beginning but gradually differ from each other through the whole adaptive assignment phase, this randomized strategy can balance the exploration-exploitation trade-off in worker selection, by exploring more workers at earlier steps and doing more exploitation at later steps.

Then we ask worker j_t for an individual label y_{i_t,j_t} and calculate the weighted vote $\widehat{y}_{i_t}$ by using Eqn. 2.1 (Line 21 in Algorithm 1). With the weighted vote $\widehat{y}_{i_t}$, we obtain the loss of the selected worker j_t:

$$l_{j_t,t} = \mathbb{1}_{\widehat{y}_{i_t} \neq y_{i_t,j_t}} \text{ (Line 22 in Algorithm 1).}$$

Recall that we do not have any assumption on how each worker gives a label. Then we have no stochastic assumption on the generation of losses. Thus we can consider this problem as an adversarial bandit problem. Note that we can only observe the loss of the selected worker j_t, and for other workers, the losses are unobserved. This is called *limited feedback* in bandit problems. Here, we decided to give an unbiased estimate of loss for any worker from the above distribution:

$$\widetilde{l}_{j,t} = \frac{l_{j,t}}{p_{j,t}} \mathbb{1}_{j=j_t}.$$

It is easy to see that the expectation of $\widetilde{l}_{j,t}$ with respect to the selection of worker j_t is exactly the loss $l_{j,t}$. Finally, we update the cumulative losses and the confidence scores of tasks with the same context as the current one (Line 22-23 in Algorithm 1).

The above assignment step is repeated T_2 times until the budget is used up.

2.3 Theoretical Analysis

In this section, we theoretically analyze the proposed bandit-based task assignment (BBTA) method.

2.3.1 Regret

The behavior of a bandit strategy is studied by means of *regret* analysis. Usually, the performance of a bandit strategy is compared with that of the optimal one, to show the "regret" for not following the optimal strategy. In our worker selection problem, we use the notion of regret to investigate how well the proposed strategy can select better workers from the whole worker pool, by comparing our strategy with the optimal one.

The reason why we use regret as the evaluation measure is that the whole worker selection process is working in an online mode. From the perspective of a requester, the objective is to maximize the average accuracy of the estimated true labels with the constraint of the budget. Ideally, this is possible when we have complete knowledge of the whole process, e.g. the reliability of each worker and the budget for each context. However, in the setting of worker selection, since we can not know beforehand any information about the worker behaviors and the coming contexts in the future, it is not meaningful to try to maximize the average accuracy. Instead, the notion of regret can be used as an evaluation measure for the strategy of worker selection in the proposed method, which evaluates a relative performance loss compared with the optimal strategy. As a general objective for online problems (Hazan, 2006; Shalev-Shwartz, 2007; Hazan, 2011; Shalev-Shwartz, 2012), minimizing the regret is a common and reasonable approach to guaranteeing the performance.

Specifically, we define the regret by

$$\overline{R}_T = \max_{g:\mathcal{S}\to\{1,\dots,K\}} \mathbb{E}\left[\sum_{t=1}^{T_2}(l_{j_t,t} - l_{g(s),t})\right],$$

where $g : \mathcal{S} \to \{1, \dots, K\}$ is the mapping from contexts to workers, s is the context of the task at step t, and $T_2 = T - SKN'$ is the budget for the adaptive assignment phase. The regret we defined is the difference of losses between our strategy and the expected optimal one. Note that the optimal strategy here is assumed to be obtained in hindsight under the same sequence of contexts as that of our strategy. The expectation is taken with respect to both the randomness of loss generation and worker selection $j_t \sim p_t$, where p_t is the current worker distribution at step t of our strategy. Since we consider the loss generation as

oblivious (i.e. the loss $l_{j_t,t}$ only depends on the current step t), we do not take the expectation over all sample paths.

2.3.2 Regret Bounds

We further denote $\mathcal{T}_2^s$ as the set of indices for steps with context s, and $T_2^s = |\mathcal{T}_2^s|$ as the total appearance count of context s in the adaptive assignment phase. The following theorem shows the regret bound of BBTA.

Theorem 2.1. *The regret of BBTA with* $\eta_{t^s}^s = \sqrt{\frac{\ln K}{t^s K}}$ *is bounded as*

$$\overline{R}_T \leq 2\sqrt{(T - SKN')SK \ln K} + \frac{SN'^2}{4}\sqrt{\frac{\ln K}{K}} + SN'.$$

Proof.

$$\begin{aligned}
\overline{R}_T &= \max_{g:\mathcal{S}\to\{1,\ldots,K\}} \mathbb{E}\left[\sum_{t=1}^{T_2}(l_{j_t,t} - l_{g(s),t})\right] \\
&= \sum_{s\in\mathcal{S}} \max_{j=1,\ldots,K} \mathbb{E}\left[\sum_{t\in\mathcal{T}_2^s}(l_{j_t,t} - l_{j,t})\right] \\
&\leq \sum_{s\in\mathcal{S}} \left(2\sqrt{T_2^s K \ln K} + \frac{N'^2}{4}\sqrt{\frac{\ln K}{K}} + N'\right) \\
&\leq 2\sqrt{(T - SKN')SK \ln K} + \frac{SN'^2}{4}\sqrt{\frac{\ln K}{K}} + SN',
\end{aligned}$$

where the first inequality is implied by Lemma 2.2 shown below and the second one is due to Jensen's inequality and $\sum_{s\in\mathcal{S}} T_2^s = T_2$. □

Now, let us consider the following setting. Suppose there is only one context ($S = 1$) for all tasks (i.e. in the homogeneous setting). Then we do not have to distinguish different contexts. In particular, t^s, $\eta_{t^s}^s$, L_{j,t^s}^s, w_{j,t^s}^s and T_2^s are respectively equivalent to t, η_t, $L_{j,t}$, $w_{j,t}$ and T_2 in this setting. For convenience, we omit the superscript s in this setting. The regret now is

$$\overline{R}'_T = \max_{j\in\{1,\ldots,K\}} \mathbb{E}\left[\sum_{t=1}^{T_2}(l_{j_t,t} - l_{j,t})\right],$$

which is the difference of losses between our strategy and the one that always selects the best worker in expectation. The following Lemma 2.2 shows a bound of $\overline{R}'_T$.

Lemma 2.2. *If there is only one context, then the regret of BBTA with* $\eta_t = \sqrt{\frac{\ln K}{tK}}$ *is bounded as*

$$\overline{R}'_T \leq 2\sqrt{(T - KN')K \ln K} + \frac{N'^2}{4}\sqrt{\frac{\ln K}{K}} + N'.$$

The proof of Lemma 2.2 is provided in Section 2.6.

Remark 1. Theorem 2.1 and Lemma 2.2 show sub-linear regret bounds with respect to T for BBTA, indicating that the performance of our strategy converges to that of the expected optimal one as the budget T goes to infinity.

Remark 2. The pure exploration phase is necessary for obtaining some knowledge of workers' reliability before adaptively assigning tasks, and N' controls the length of the pure exploration phase. In Theorem 2.1, it is easy to see that a larger N' makes the term $2\sqrt{(T - SKN')SK \ln K}$ (corresponding to the adaptive assignment phase) smaller but results in larger $\frac{N'^2}{4}\sqrt{\frac{\ln K}{K}} + N'$ (related to the pure exploration phase). Moreover, a longer pure exploration phase also consumes a larger proportion of the total budget. The pure exploration phase is effective (as we will see in Section 2.5), but how effective it can be is not only related to the choice of N' (even we could obtain the optimal N' that minimizes the bound), but also depends on many other factors, such as the true accuracy of workers and how different their labeling performances are from each others, which are unfortunately unknown in advance. A practical choice used in our experiments in Section 2.5 is to set a very small N' (e.g. $N' = 1$).

2.4 Existing Methods

In this section, we review the existing online-style methods for crowdsourcing and discuss similarities to and differences from the proposed method.

2.4.1 IEThresh

IEThresh (Donmez et al., 2009) builds upon *interval estimation* (IE) (Kaelbling, 1990), which is used in reinforcement learning for handling the trade-off between exploration and exploitation in action selection. Before each selection, the IE method estimates the *upper confidence interval* (UI) for each action a according to its cumulative reward:

$$\mathrm{UI}(a) = m(a) + t_{\frac{\alpha}{2}}^{(n-1)} \frac{s(a)}{\sqrt{n}},$$

where $m(a)$ and $s(a)$ are the mean and standard deviation of rewards that has been received so far by choosing action a, n denotes the number of times a has been selected so far, and $t_{\frac{\alpha}{2}}^{(n-1)}$ is the critical value for Student's t-distribution with $n-1$ degrees of freedom at significance level $\alpha/2$. In experiments, we set α at 0.05, following Donmez et al. (2009).

In IEThresh, taking the action a_j corresponds to selecting worker j to ask for a label. The reward is 1 if the worker's label agrees with the majority vote, and 0 otherwise. At each step, a new task arrives, UI scores of all workers are re-calculated, and a subset of workers with indices

$$\{j \mid \mathrm{UI}(a_j) \geq \epsilon \cdot \max_a \mathrm{UI}(a)\}$$

are asked for labels. It is easy to see that the pre-determined threshold parameter ϵ controls the size of worker subset at each step.

2.4.2 CrowdSense

Similarly to IEThresh, CrowdSense (Ertekin et al., 2014) also dynamically samples worker subsets in an online mode. The criterion that CrowdSense uses for handling exploration-exploitation in worker selection is described as follows.

Before assigning a new task i to workers, CrowdSense first calculates a quality estimate for each worker:

$$Q_j = \frac{a_j + K}{c_j + 2K},$$

where c_j is the number of times worker j has given a label, a_j represents how many of those labels were consistent with the weighted vote, and K is a smoothing parameter. In experiments, we set K at 100, which is empirically shown by Ertekin et al. (2014) as the best choice. Then a confidence score is calculated as

$$\text{Score}(S_i) = \sum_{j \in S_i} y_{i,j} Q_j,$$

where S_i is a subset of workers initially containing two top-quality workers and another random worker. Then CrowdSense determines whether to add a new worker l to S_i according to the criteria:

$$\frac{|\text{Score}(S_i)| - Q_l}{|S_i| + 1} < \epsilon,$$

where ϵ is a pre-determined threshold parameter. If the above is true, the candidate worker is added to S_i, and $\text{Score}(S_i)$ is re-calculated. After fixing the subset of workers for task i, CrowdSense aggregates the collected labels by the weighted voting mechanism, using Q_j as the weight of worker j.

2.4.3 Optimistic Knowledge Gradient

Optimistic Knowledge Gradient (OptKG) (Chen et al., 2013) uses an N-coin-tossing model, formulates the worker selection problem as a Bayesian Markov decision process (MDP), and obtains the optimal allocation sequence for any finite budget T via a computationally efficient approximate policy.

For better illustration, a simplified model with noiseless workers is first considered. In the model, task i is characterized by θ_i drawn from a known Beta prior distribution, $\text{Beta}(a_i^0, b_i^0)$. In practice, we may simply start from $a_i^0 = b_i^0 = 1$ if there is no prior knowledge (Chen et al., 2013). At each stage t with $\text{Beta}(a_i^t, b_i^t)$ as the current posterior distribution for θ_i, task i_t is picked, and its label, drawn from Bernoulli distribution $y_{i_t} \sim \text{Bernoulli}(\theta_{i_t})$, is acquired. Then $\{a_i^t, b_i^t\}_{i=1}^N$ are put into an $n \times 2$ matrix S^t, called a state matrix with row i as $S_i^t = (a_i^t, b_i^t)$. The state matrix is updated according to the following rule:

$$S^{t+1} = \begin{cases} S^t + (\boldsymbol{e}_{i_t}, \mathbf{0}) & \text{if } y_{i_t} = 1, \\ S^t + (\mathbf{0}, \boldsymbol{e}_{i_t}) & \text{if } y_{i_t} = -1, \end{cases}$$

where $\boldsymbol{e}_{i_t}$ is an n-dimensional vector with 1 at the i_t-th entry and 0 at all others. The state transition probability is calculated as

$$p(y_{i_t} = 1 \mid S^t, i_t) = \mathbb{E}[\theta_{i_t} \mid S^t] = \frac{a_{i_t}^t}{a_{i_t}^t + b_{i_t}^t}.$$

The stage-wise expected reward is defined as

$$R(S^t, i_t) = p_1 R_1(a_{i_t}^t, b_{i_t}^t) + p_2 R_2(a_{i_t}^t, b_{i_t}^t),$$

where

$$\begin{aligned}
&p_1 = p(y_{i_t} = 1 \mid S^t, i_t),\\
&p_2 = 1 - p(y_{i_t} = 1 \mid S^t, i_t),\\
&R_1(a_{i_t}^t, b_{i_t}^t) = h(I(a_{i_t}^t + 1, b_{i_t}^t)) - h(I(a_{i_t}^t, b_{i_t}^t)),\\
&R_2(a_{i_t}^t, b_{i_t}^t) = h(I(a_{i_t}^t, b_{i_t}^t + 1)) - h(I(a_{i_t}^t, b_{i_t}^t)),\\
&I(a_{i_t}^t, b_{i_t}^t) = p(\theta \geq 0.5 \mid \theta \sim \text{Beta}(a_{i_t}^t, b_{i_t}^t)),\\
&h(x) = \max(x, 1 - x).
\end{aligned}$$

Then the following optimization problem is solved:

$$V(S^0) = G_0(S^0) + \sup_{\pi} \mathbb{E}_{\pi} \left[\sum_{t=0}^{T-1} R(S^t, i_t) \right],$$

where $G_0(S^0) = \sum_{i=1}^{n} h(I(a_i^0, b_i^0))$. With the above definitions, the problem is formulated as a T-stage MDP associated with a tuple

$$\{T, \{\mathcal{S}^t\}, \mathcal{A}, p(y_{i_t} \mid S^t, i_t), R(S^t, i_t)\},$$

where

$$\mathcal{S}^t = \left\{ \{a_i^t, b_i^t\}_{i=1}^n : a_i^t \geq a_i^0, b_i^t \geq b_i^0, \sum_{i=1}^{n} (a_i^t - a_i^0) + (b_i^t - b_i^0) = t \right\},$$

and the action space is the set of indices for tasks that could be labeled next: $\mathcal{A} = \{1, \ldots, n\}$.

Since the computation of dynamic programming for obtaining the optimal policy of the above formulation is intractable, a computationally efficient and consistent approximate policy is employed, resulting in the criterion for task selection at

each stage t:

$$i_t = \arg\max_{i \in \{1,\ldots,n\}} \left(\max(R_1(a_i^t, b_i^t), R_2(a_i^t, b_i^t)) \right).$$

Finally, OptKG outputs the positive set

$$H_T = \{i : a_i^T \geq b_i^T\}.$$

Based on the simplified model above, workers' reliability can be further introduced. Specifically, worker j is characterized by a scalar parameter

$$\rho_j = p(y_{i,j} = y_i^* \mid y_i^*),$$

and is assumed drawn from a Beta prior distribution: $\rho_j \sim \text{Beta}(c_j^0, d_j^0)$. Then at each stage, a pair of worker and task is adaptively selected, according to a new criterion, which is derived in Appendix.5 in the original OptKG paper (Chen et al., 2013). In experiments, we follow Chen et al. (2013) and set $c_j^0 = 4$ and $d_j^0 = 1$ for each worker, which indicates that we have the prior belief that the average accuracy of workers is $4/5 = 80\%$.

2.4.4 Discussion

It is essential to model workers' reliability in crowdsourcing. All of the above online-style methods have done this in different ways, such as upper confidence intervals $\text{UI}(a_j)$ in IEThresh, worker qualities Q_j in CrowdSense, and the scalar parameter ρ_j in OptKG. However, there are also some practical issues in these methods, which are described as follows.

Issues in the Heterogeneous Setting

Although IEThresh and CrowdSense employ different criterions for dealing with the exploration-exploitation trade-off in worker selection, they share the similar mechanism of dynamically sampling worker subsets.

In particular, at each step, a new task comes, workers' reliability is learned according to their labeling performances on previous tasks, and a subset of workers considered to be reliable is sampled for labeling the new task.

However, in heterogeneous crowdsourcing, a worker in the subset who is reliable at previous tasks may be bad at new ones. On the other hand, the worker who is good at new tasks may have already been eliminated from the subset. Therefore, it is reasonable to model the context-dependent reliability for workers, and then match each task to workers who can do it best.

Issues of Non-Fixed Budget

Another drawback in IEThresh and CrowdSense is that the exact amount of total budget is not pre-fixed.

In these two methods, a threshold parameter ϵ is used for controlling the size of worker subsets. That is, ϵ determines the total budget. However, how many workers in each subset is unknown beforehand. Therefore, we will not know the exact amount of total budget until the whole worker selection process ends. This is not so practical in crowdsourcing, since in the worker selection problem, we pursue not only collecting reliable labels but also intelligently using the pre-fixed budget.

Moreover, both of IEThresh and CrowdSense lack of theoretical analyses about the relation between the budget and the performance.

Issues of Theoretical Guarantees

OptKG and the proposed method (BBTA) attempt to intelligently use a pre-fixed budget, and ask one worker for a label of the current task at each step. OptKG formulates the worker selection problem as a Bayesian MDP, and is proved to produce a consistent policy in homogeneous worker setting (i.e. the policy will achieve 100% accuracy almost surely when the total budget goes to infinity).

However, when the heterogeneous reliability of workers is introduced, more sophisticated approximation is also involved in OptKG, making it difficult to give theoretical analysis for OptKG in heterogeneous worker setting.

On the other hand, BBTA is a contextual bandit formulation designed for heterogeneous crowdsourcing, and the regret analysis demonstrates the performance of our worker selection strategy will converge to that of the optimal one when the total budget goes to infinity.

Name	# Tasks	# Workers	# Contexts
ionosphere	351	30	3
breast	569	40	4
pima	768	50	5
Recognizing Textual Entailment	800	164	1
Gender Hobby	204	42	2

Table 2.1: Descriptions of datasets used in this thesis.

2.5 Experiments

In this section, we experimentally evaluate the usefulness of the proposed bandit-based task assignment (BBTA) method. To compare BBTA with existing methods, we first conduct experiments on benchmark data with simulated workers, and then use real data for further comparison. The descriptions of all dataset are summarized in Table 2.1. All of the experimental results are averaged over 30 runs.

2.5.1 Benchmark Data

We perform experiments on three popular UCI benchmark datasets[2]: ionosphere ($N = 351$), breast ($N = 569$), and pima ($N = 768$). We consider data instances in these datasets as labeling tasks in crowdsourcing. True labels of all tasks in these datasets are available for evaluation.

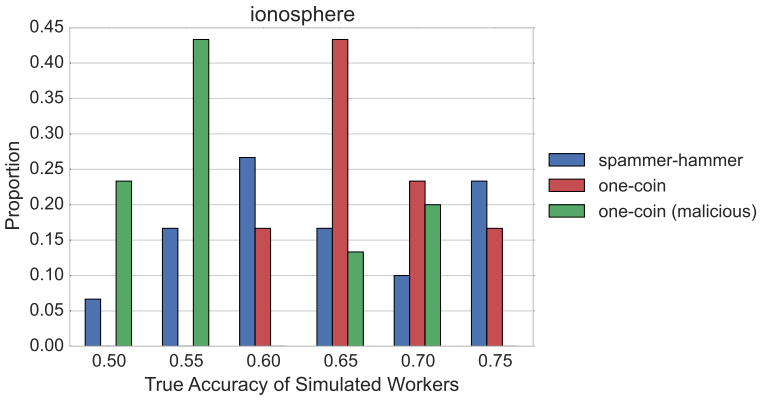
To simulate various heterogeneous cases in the real world, we use *k-means* to cluster these three datasets into $S = 3, 4, 5$ subsets respectively (corresponding to different contexts).

Since there are no crowd workers in these datasets, we then simulate workers ($K = 30, 40, 50$, respectively) by using the following worker models in heterogeneous setting:

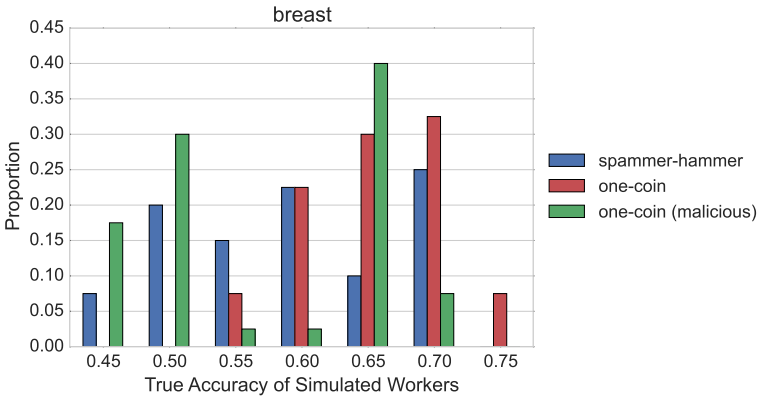
Spammer-Hammer Model

A hammer gives true labels, while a spammer gives random labels (Karger et al., 2011). We introduce this model into heterogeneous setting: each worker is a hammer on one subset of tasks but a spammer on others.

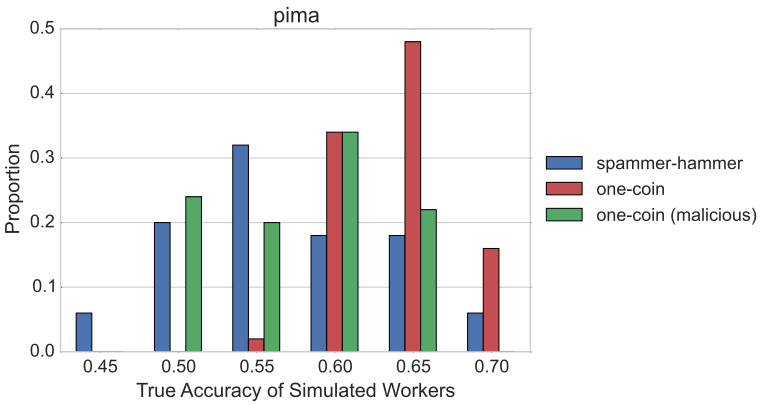
[2]http://archive.ics.uci.edu/ml/



(a) $N = 351, K = 30, S = 3$



(b) $N = 569, K = 40, S = 4$



(c) $N = 768, K = 50, S = 5$

Figure 2.1: Distribution of true accuracy of simulated workers for three benchmark datasets with three worker models.

One-Coin Model

Each worker gives true labels with a given probability (i.e. accuracy). This model is widely used in many existing crowdsourcing literatures (e.g. Raykar et al., 2010; Chen et al., 2013) for simulating workers. We use this model in heterogeneous setting: each worker gives true labels with higher accuracy (we set it to 0.9) on one subset of tasks, but with lower accuracy (we set it to 0.6) on others.

One-Coin Model (Malicious)

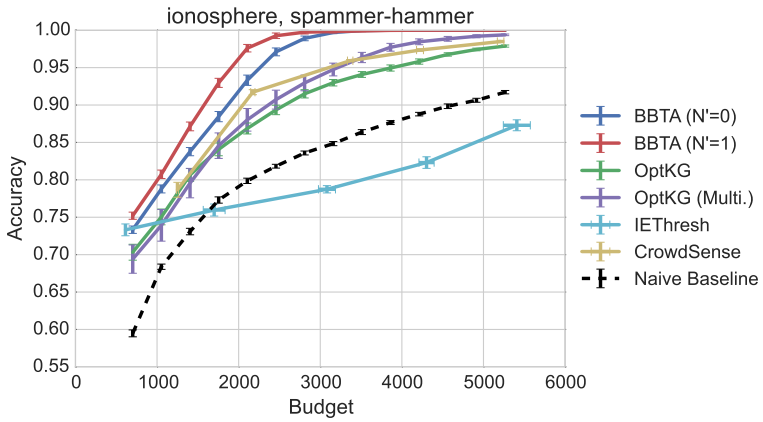
This model is based on the previous one, except that we add more malicious labels: each worker is good at one subset of tasks (accuracy: 0.9), malicious or bad at another one (accuracy: 0.3), and normal at the rest (accuracy: 0.6).

With the generated labels from simulated workers, we can calculate the true accuracy for each worker by checking the consistency with the true labels. Figure 2.1 illustrates the proportions of simulated workers with the true accuracy falling in the associated interval (e.g., 0.65 represents that the true accuracy is between 60% and 65%). It is easy to see that the spammer-hammer model and the one-coin model (malicious) create more adversarial environments than the one-coin model.

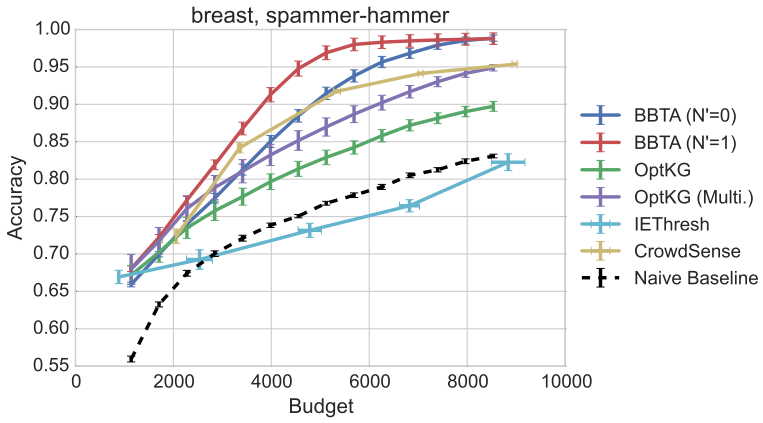
Comparison Experiments

We compare BBTA with three state-of-the-art methods: IEThresh, CrowdSense and OptKG in terms of accuracy. Accuracy is calculated as the proportion of correct estimates for true labels. Also, since the budget is limited, we expect a worker selection method can achieve its highest accuracy as fast as possible as the budget increases (high convergence speed).

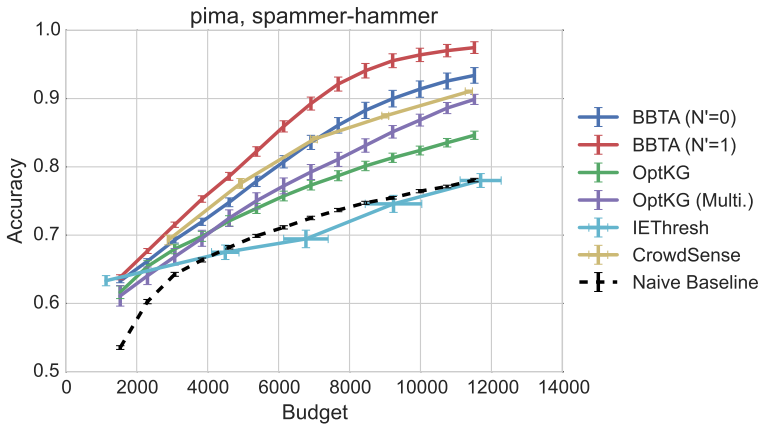
We set $N' = 0$ and 1 for BBTA to see the effectiveness of the pure exploration phase. We also implement a naive baseline for comparison: we randomly select a task-worker pair at each step, and use majority voting mechanism for label aggregation. Accuracy of all methods is compared at different levels of budget. For BBTA, OptKG, and the naive baseline, we set the maximum amount of budget at $T = 15N$. Since the budget is not pre-fixed in IEThresh and CrowdSense, we carefully select the threshold parameters for them, which affect the consumed budgets.



(a) $N = 351, K = 30, S = 3$

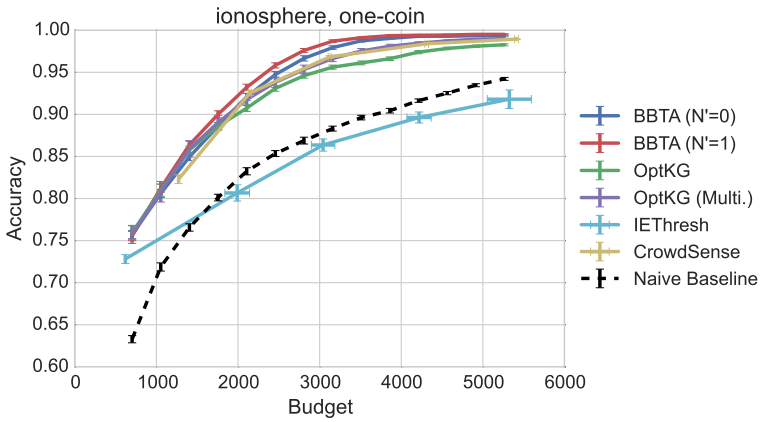


(b) $N = 569, K = 40, S = 4$

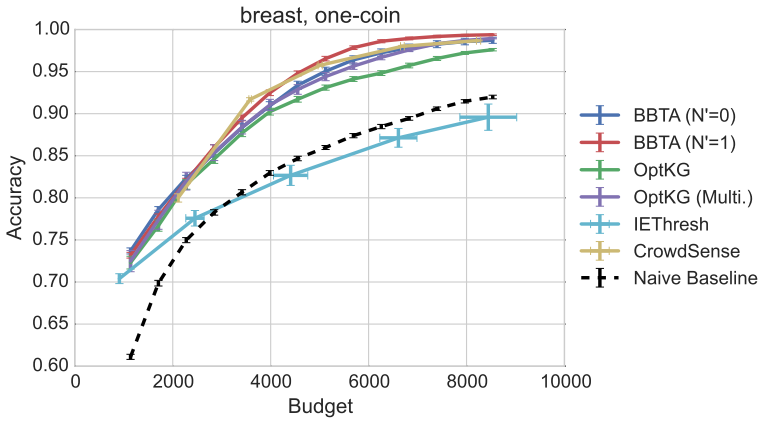


(c) $N = 768, K = 50, S = 5$

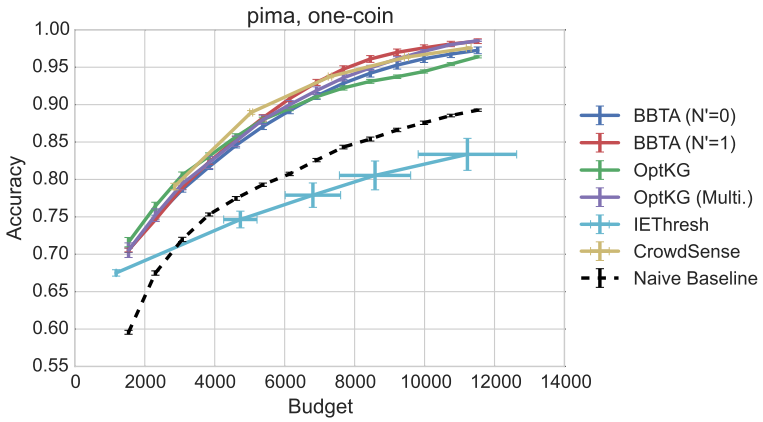
Figure 2.2: Results on three benchmark datasets with the spammer-hammer worker model.



(a) $N = 351, K = 30, S = 3$

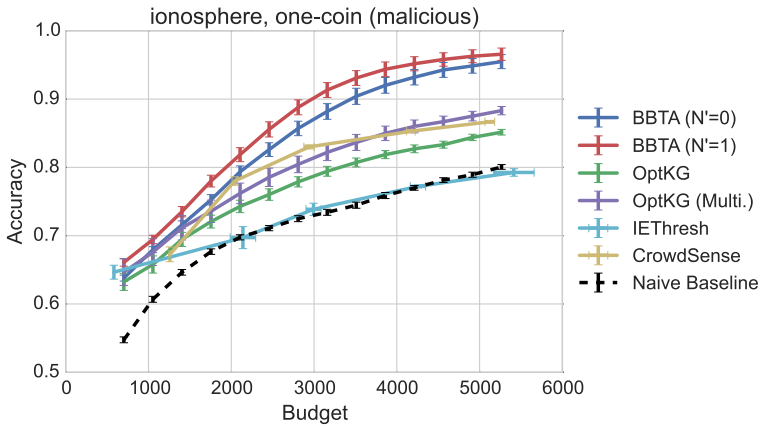


(b) $N = 569, K = 40, S = 4$

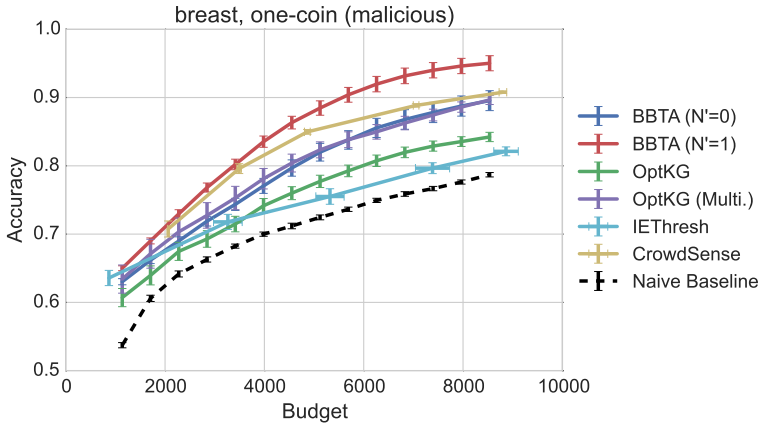


(c) $N = 768, K = 50, S = 5$

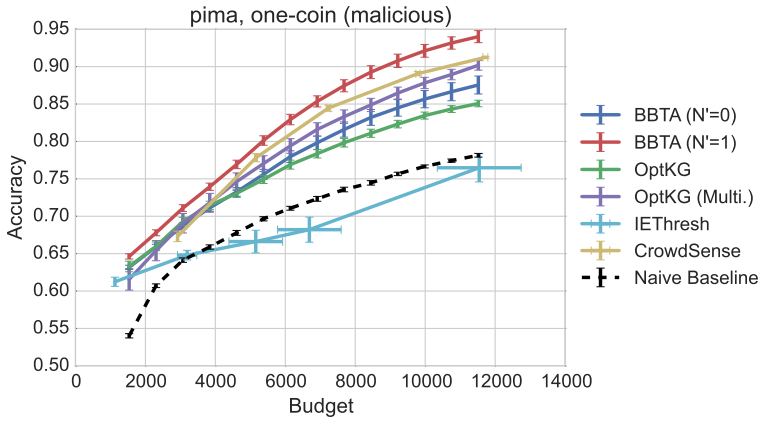
Figure 2.3: Results on three benchmark datasets with the one-coin worker model.



(a) $N = 351, K = 30, S = 3$



(b) $N = 569, K = 40, S = 4$



(c) $N = 768, K = 50, S = 5$

Figure 2.4: Results on three benchmark datasets with the one-coin worker model (malicious).

Additionally, we also try to introduce the state-of-the-art methods (designed for the homogeneous setting) into the heterogeneous setting. Specifically, we split the total budget and allocate a sub-budget to a context in proportion to the number of tasks with this context. In particular, for context s, we allocate the sub-budget $T \cdot |\text{tasks with context } s|/N$. Then we can run an instance of a homogeneous method for each context within the associated sub-budget. Since OptKG has the most similar problem setting to that of the proposed method, it is straightforward to run multiple instances of OptKG with a pre-fixed sub-budget for each context. However, for the two heuristic methods IEThresh and CrowdSense, it is difficult to figure out how to use them in this way, since the budget could not be pre-determined in their settings.

Figure 2.2, 2.3 and 2.4 show the averages and standard errors of accuracy as functions of budgets for all methods in nine cases (i.e. three datasets with three worker models). As we can see, BBTA with $N' = 1$ works better than that with $N' = 0$, indicating that the pure exploration phase helps in improving the performance.

It is also shown that BBTA ($N' = 1$) outperforms other methods in all six cases with the spammer-hammer model and the one-coin model (malicious). This demonstrates that BBTA can handle spamming or malicious labels better than others in more adversarial heterogeneous environments.

For the three cases with the one-coin model where there are more reliable labels, almost all methods have good performances. Nevertheless, IEThresh performs poorly in all cases, even worse than the naive baseline. The reason is that IEThresh samples a subset of reliable workers at each step, by calculating upper confidence intervals of workers based on their labeling performances on previous tasks. In heterogeneous settings, however, a worker reliable at previous tasks may be poor at next ones. This makes IEThresh learn workers' reliability incorrectly, resulting in poor sampling of worker subsets.

Although CrowdSense also adopts the mechanism of dynamically sampling worker subsets, its exploration-exploitation criterion gives it a chance of randomly selecting some workers who may be reliable at next tasks.

For OptKG, not surprisingly, OptKG (Multi.) which is aware of contexts outperforms the original OptKG. The tendency of either of them implies that they

may achieve the best accuracy as the budget goes to infinity, but the convergence speed is shown to be slower than those of BBTA and CrowdSense. In crowdsourcing, it is important to achieve the best accuracy as fast as possible, especially when the budget is limited.

Experiments with varying N'

Now we are curious about how N' affects the performance of BBTA. It is desirable to find a optimal N' to achieve the best performance.

Recall that the bound in Theorem 2.1 shows that our strategy of worker selection performs as well as the optimal one when the budget goes to infinity. In this bound, N' is considered as a constant together with T, S and K. This bound holds no matter how we choose N' in $[0, T/(SK)]$. Since we are going to check how the performance changes with varying N', a natural question to ask is: does the optimal N' that minimizes the bound also bring the best performance?

Let us first check whether the optimal N' exists. We rewrite the bound in Theorem 2.1 as the function of N' as

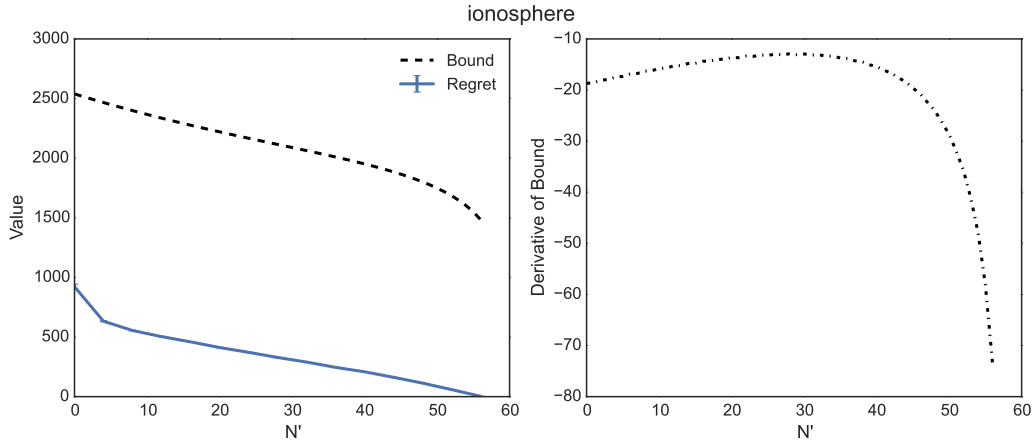
$$\Theta(N') = 2\sqrt{(T - SKN')SK \ln K} + \frac{SN'^2}{4}\sqrt{\frac{\ln K}{K}} + SN'.$$

And the derivative of $\Theta(N')$ with respect to N' is

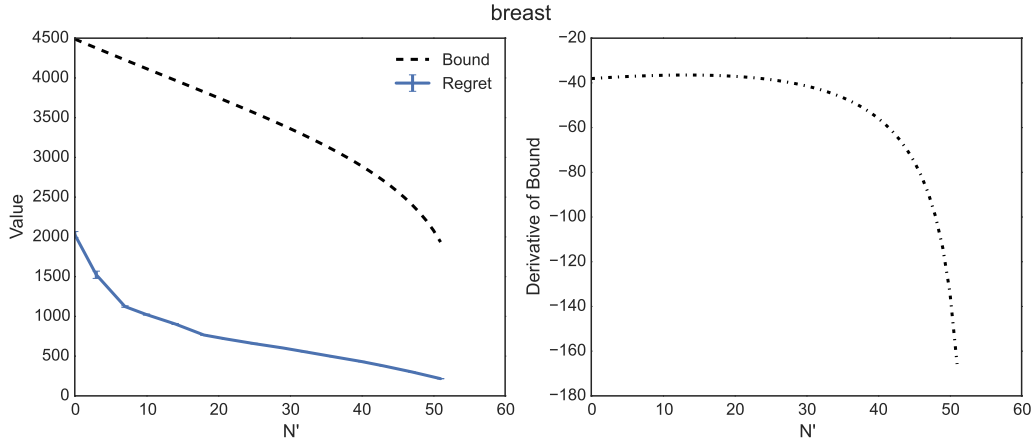
$$\Theta'(N') = -\frac{S^2K^2 \ln K}{\sqrt{(T - SKN')SK \ln K}} + \frac{SN'}{2}\sqrt{\frac{\ln K}{K}} + S.$$

For illustration, we plot the regret $\Theta(N')$ and its derivative $\Theta'(N')$ respectively in Figure 2.5 on the three benchmark datasets. We also plot experimental values of the regret. Recall that the regret we defined in Section 2.3.1 is the difference of cumulative losses between our strategy and the optimal one. However, the optimal strategy could not be captured in experiments unless it exactly performs. Hence, to calculate the regret, we assume the cumulative loss of the optimal strategy is zero. So the regret we plot is essentially the cumulative loss of our strategy. Then the actual regret which could not be plotted would be under the plotted regret.

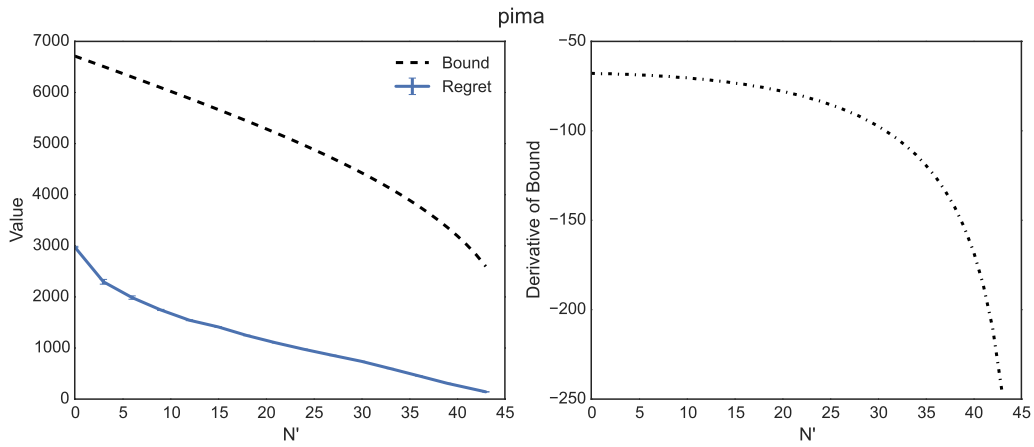
As shown in Figure 2.5, the regret bounds are quite loose. We can also observe that the values of bound and regret are both monotone decreasing with respect to N' in general, and the derivative is always negative and also in a decreasing trend.



(a) $N = 351, K = 30, S = 3$



(b) $N = 569, K = 40, S = 4$



(c) $N = 768, K = 50, S = 5$

Figure 2.5: Illustration of the regret and its bound in Theorem 2.1 as the function of N' on benchmark data.

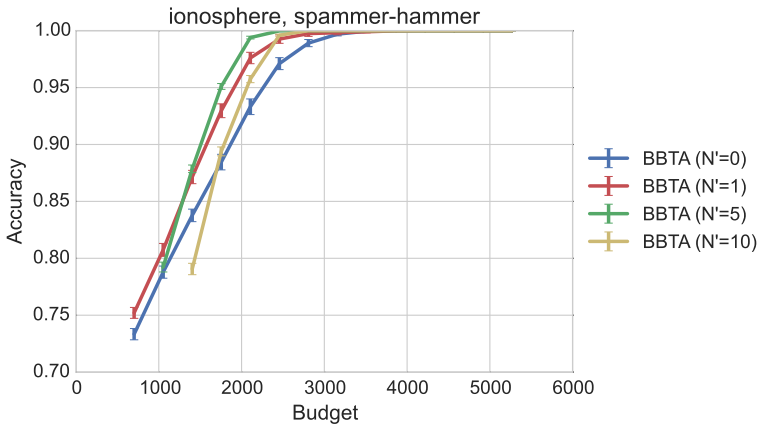
Although these results are only from three benchmark datasets, it can be implied that the optimal N' does not always exist (at least it does not exist in the cases of the three benchmark datasets). However, the results also suggest that a larger N' corresponds to a tighter bound in the cases of the three benchmark datasets. Then let us further check whether a larger N' also yields better performance.

We set $N' = 0, 1, 5, 10$, and the results are shown in Figure 2.6, 2.7 and 2.8. It can be seen that without the pure exploration phase (i.e. $N' = 0$), the performance is the worst in all nine cases. On the other hand, when we add the pure exploration phase ($N' > 0$), the performance is improved.

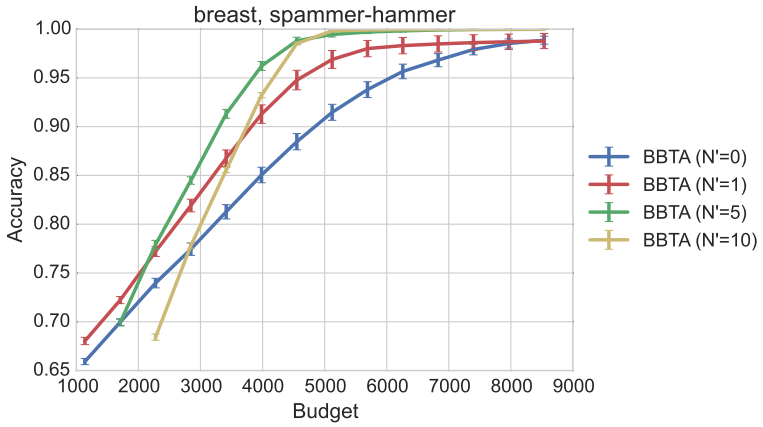
However, we are unable to conclude that the larger N' is, the better the performance is (e.g. $N' = 10$ does not always make the method achieve its highest accuracy fastest in all nine cases). Indeed, a larger N' means a longer pure exploration phase, which consumes a larger proportion of the total budget. For example, when $N' = 10$, the performance usually starts from a lower accuracy level than that when we choose other exploration lengths. Although its start level is lower, as the budget increases, the performance when $N' = 10$ can outperform all the others in most cases of the spammer-hammer and one-coin (malicious) models. However, it can only achieve the same level as the performance when $N' = 5$ in all cases of the one-coin model, but with a lower convergence speed.

In BBTA, we can only choose N' to affect the performance, and there are also some other factors such as the true reliability of workers and how different their labeling performances are from each others, of which we usually do not have prior knowledge in real-world crowdsourcing. If we could somehow know beforehand that the worker pool is complex (in terms of the difference of workers' reliability) as in the spammer-hammer and one-coin (malicious) models, setting a larger N' may help, otherwise a practical choice would be to set a small N'.

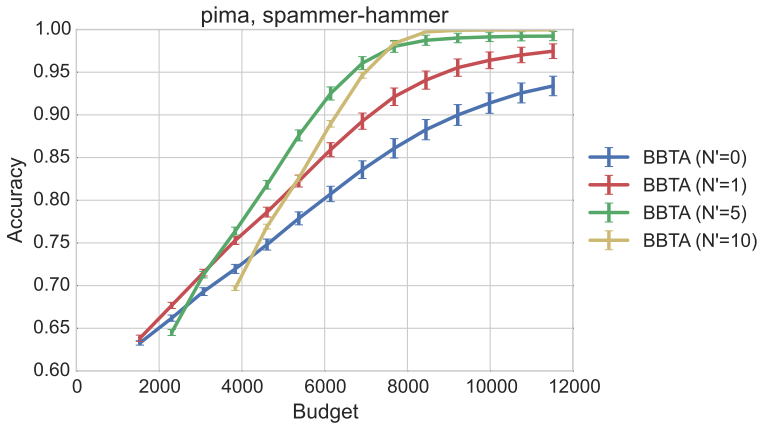
The results also imply that it is not reasonable to optimize N' by minimizing the current bound in Theorem 2.1. The bound is to show the fact that the performance of our worker selection strategy converges to that of the optimal one as the budget goes to infinity, but not the guidance for choosing N'. In order to find the optimal N', we need to further establish new theoretical guarantees. However, new theories for choosing N' is beyond the scope of this thesis, and is considered as a promising direction of our future work.



(a) $N = 351, K = 30, S = 3$

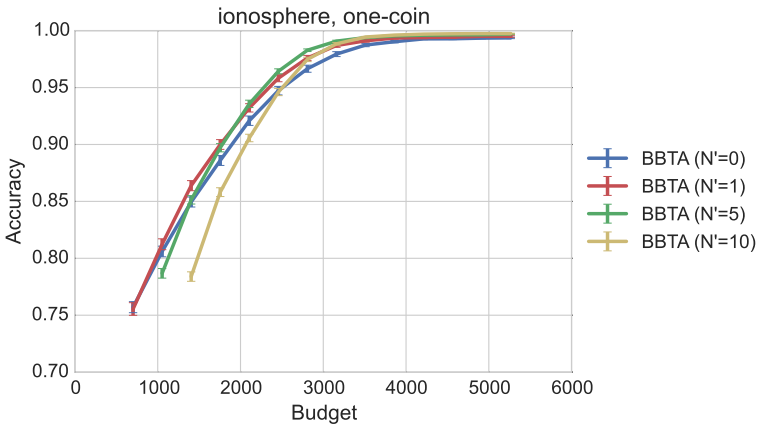


(b) $N = 569, K = 40, S = 4$

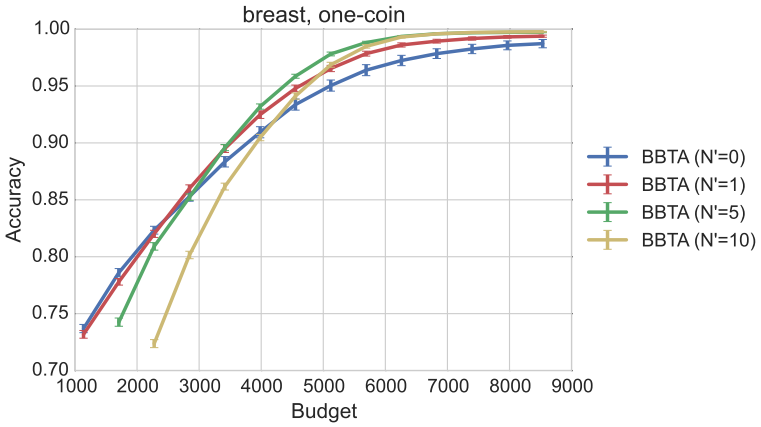


(c) $N = 768, K = 50, S = 5$

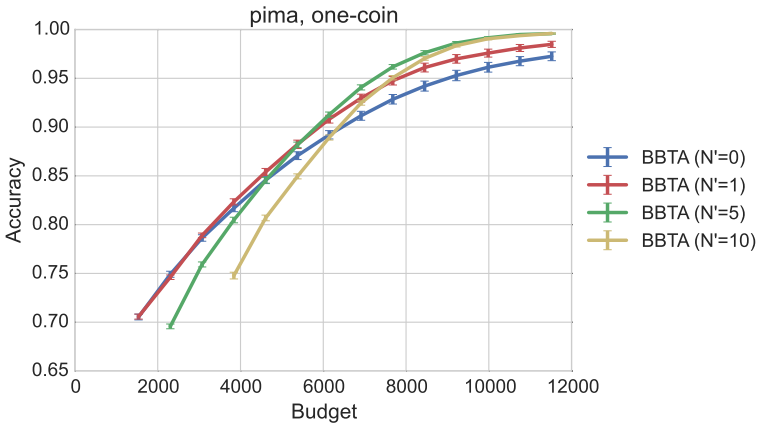
Figure 2.6: Results of changing N' on three benchmark datasets with the spammer-hammer worker model.



(a) $N = 351, K = 30, S = 3$

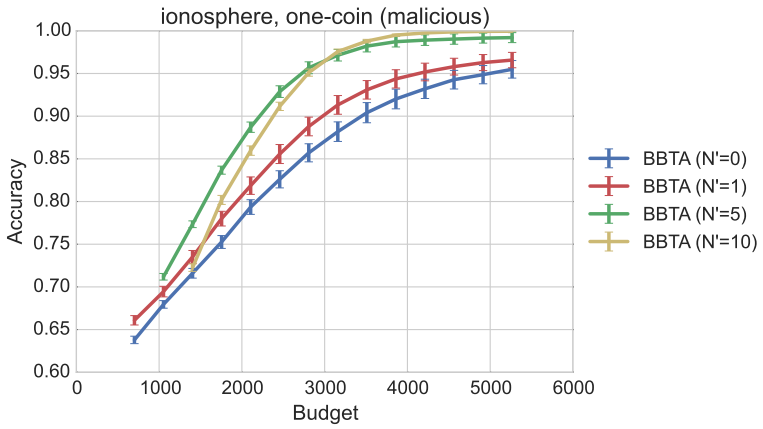


(b) $N = 569, K = 40, S = 4$

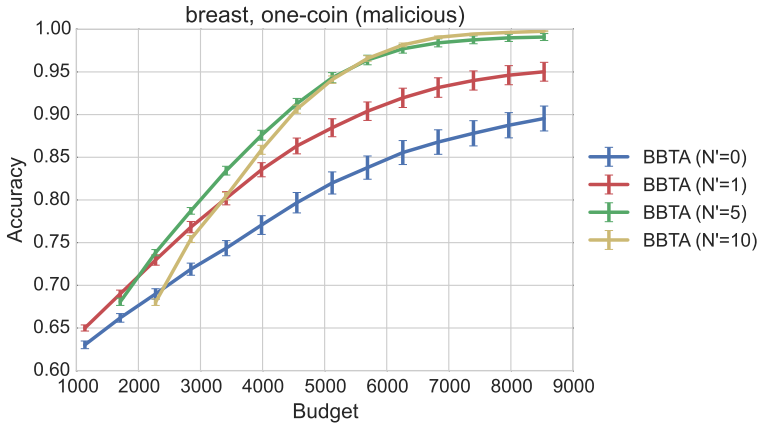


(c) $N = 768, K = 50, S = 5$

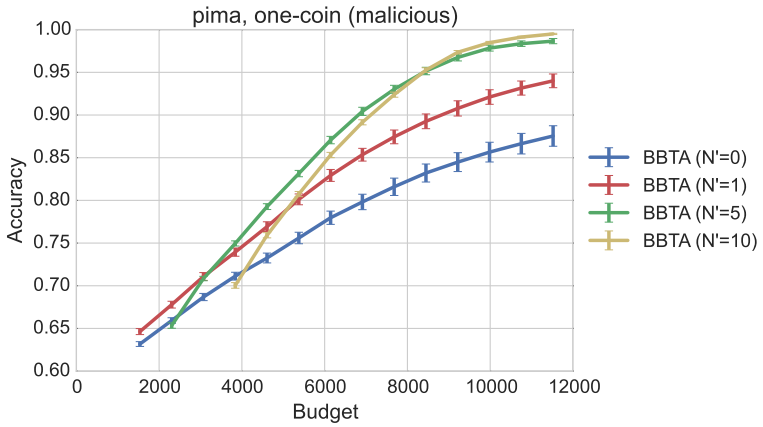
Figure 2.7: Results of changing N' on three benchmark datasets with the one-coin worker model.



(a) $N = 351, K = 30, S = 3$



(b) $N = 569, K = 40, S = 4$



(c) $N = 768, K = 50, S = 5$

Figure 2.8: Results of changing N' on three benchmark datasets with the one-coin worker model (malicious).

Experiments with various numbers of contexts

In previous experiments, we create $S = 3, 4, 5$ contexts for ionosphere, breast and pima respectively. Now we look into the relation between the number of contexts S and the accuracy on each single dataset. To this end, we create $S = 1, 2, 3, 4, 5$ contexts for each dataset and see how the accuracy changes as the number of context S increases (i.e. the heterogeneity of tasks increases).

We use the spammer-hammer model to generate worker labels, and then calculate the true accuracy for each worker. The distributions of true accuracy of workers on ionosphere, breast and pima are illustrated respectively in the box plots in Figure 2.9(a), 2.9(c), and 2.9(e). Broadly, on all these three datasets, workers are more unreliable (i.e. the overall true accuracy of workers is lower) when the number of contexts S is larger.

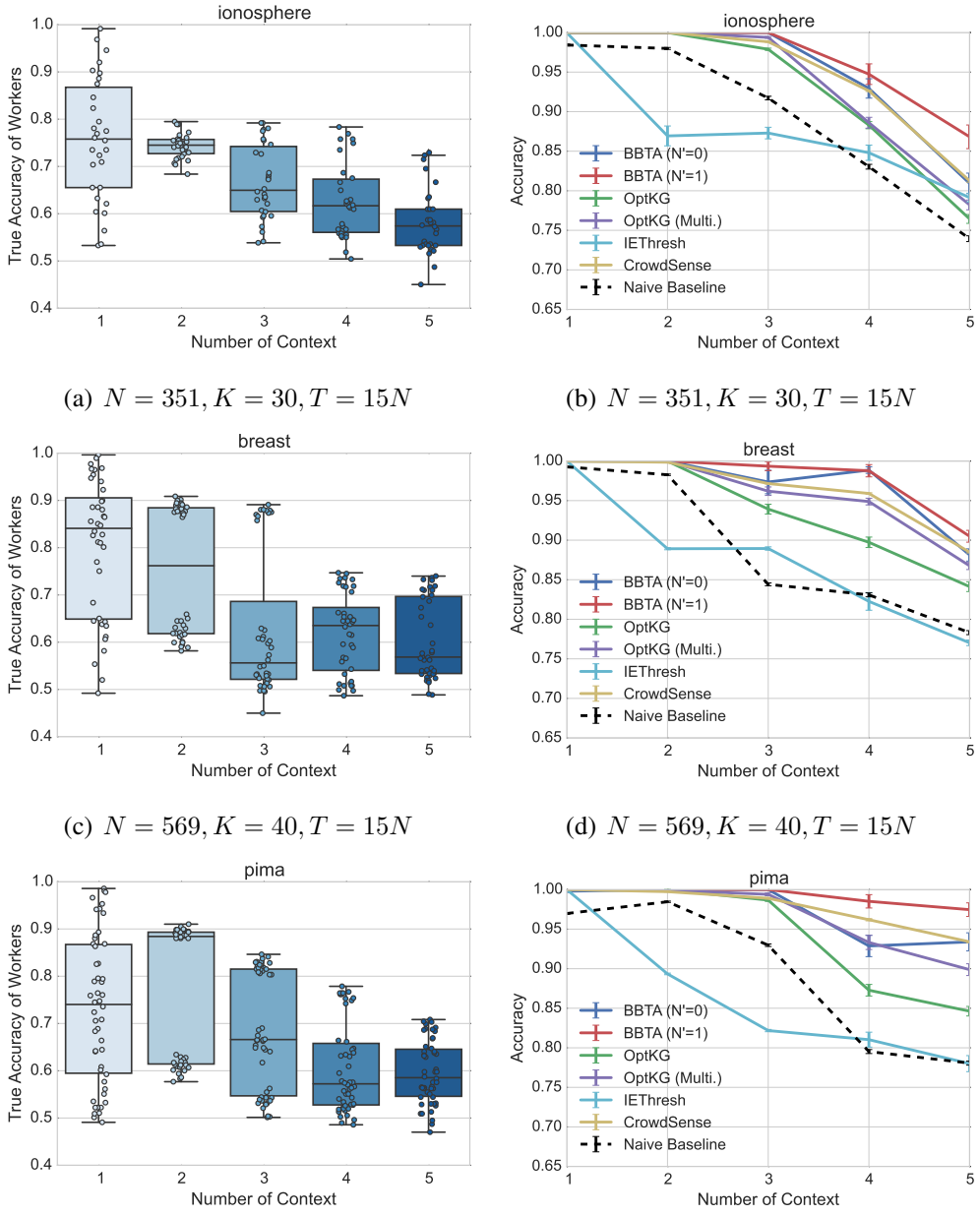
Figure 2.9(b), 2.9(d), and 2.9(f) show the comparison results. A general trend that we can observe is that the accuracy is decreasing as the number of contexts S increases for each method. In particular, when S is smaller (e.g. $S = 1, 2$), almost all the methods are comparable and perform very well. As S becomes larger, the accuracy of all methods goes down. This coincides with our intuition that when workers are more unreliable (as illustrated in the box plots), the performance is worse. We can also observe that the proposed method BBTA with $N' = 1$ still performs the best in general, and its performance decrease level is lower than that of other methods. This demonstrates again the usefulness of BBTA for handling the heterogeneity of tasks (i.e. multiple contexts).

2.5.2 Real Data

Next, we compare BBTA with the existing online-style methods on two real-world datasets: *Recognizing Textual Entailment* (RTE) and *Gender Hobby* (GH).

Recognizing Textual Entailment

We first use a real dataset from *Recognizing Textual Entailment* (RTE) tasks in natural language processing. This dataset is collected by Snow et al. (2008) using *Amazon Mechanical Turk* (MTurk). For each RTE task in this dataset, the worker is presented with two sentences and given a binary choice of whether the second



(a) $N = 351, K = 30, T = 15N$

(b) $N = 351, K = 30, T = 15N$

(c) $N = 569, K = 40, T = 15N$

(d) $N = 569, K = 40, T = 15N$

(e) $N = 768, K = 50, T = 15N$

(f) $N = 768, K = 50, T = 15N$

Figure 2.9: Results on three benchmark datasets with various numbers of contexts.

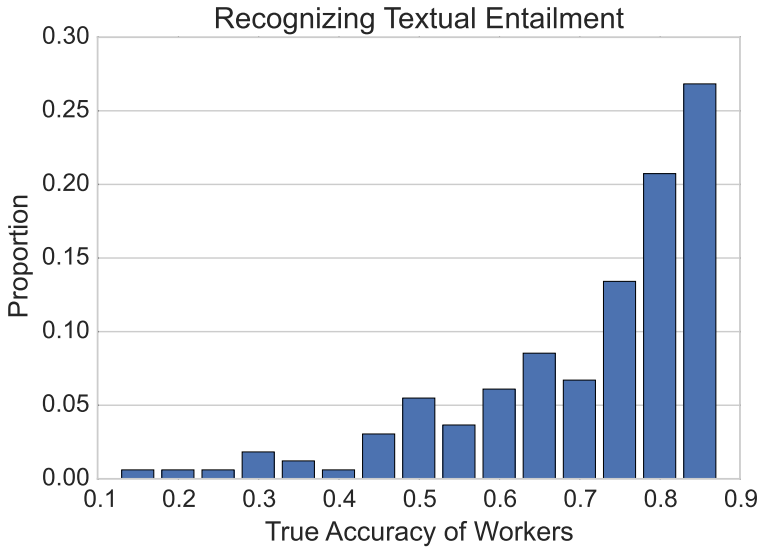


Figure 2.10: Distribution of true accuracy of workers for RTE data ($N = 800, K = 164, S = 1$).

sentence can be inferred from the first one. The true labels of all tasks are available and used for evaluating the performances of all online-style methods in our experiments.

In this dataset, there is no context information available, or we can consider all tasks have the same context. That is, this is a homogeneous dataset ($S = 1$). "Homogeneous" can be considered as a special case of "heterogeneous", so we expect the proposed method BBTA could also perform well in this homogeneous dataset. The numbers of tasks and workers in this dataset are $N = 800$ and $K = 164$ respectively.

The label matrix in the original RTE data is incomplete, and contains 8000 non-zero entries (i.e. the density of this matrix is $8000/(800 \times 164) \approx 6\%$). Then we decided to use a matrix completion algorithm[3] to fill the missing entries in the label matrix. This is to make sure that we can collect a label when any task is assigned to any worker in the experiments. After matrix completion, although not all of labels are from real workers, we still refer to this data set as "real"

[3]We use *GROUSE* (Balzano et al., 2010) for label matrix completion in our experiments.

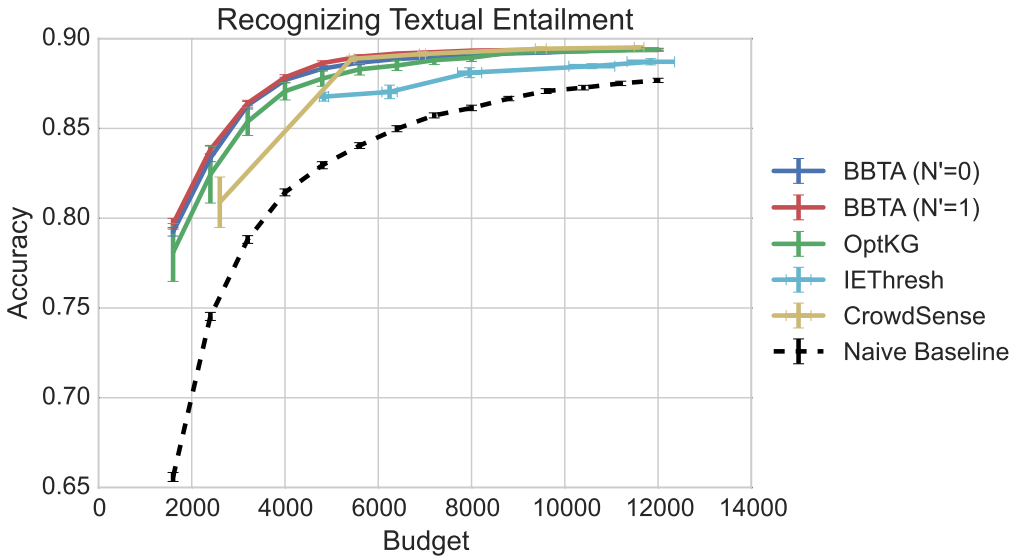


Figure 2.11: Results on RTE data ($N = 800, K = 164, S = 1$).

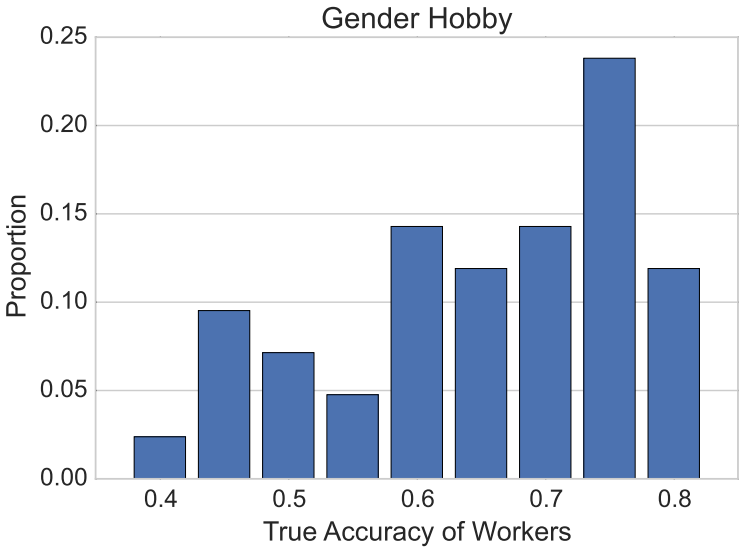
data, to distinguish it from benchmark data used in Section 2.5.1, where all labels are generated based on worker models. Then we calculate the true accuracy of workers for this dataset, as illustrated in Figure 2.10.

Figure 2.11 depicts the comparison results on the RTE data, showing that all methods work very well. The reason is that there is a significant proportion of reliable workers in this dataset as we can see in Figure 2.10, and finding them out is not a difficult mission for all methods.

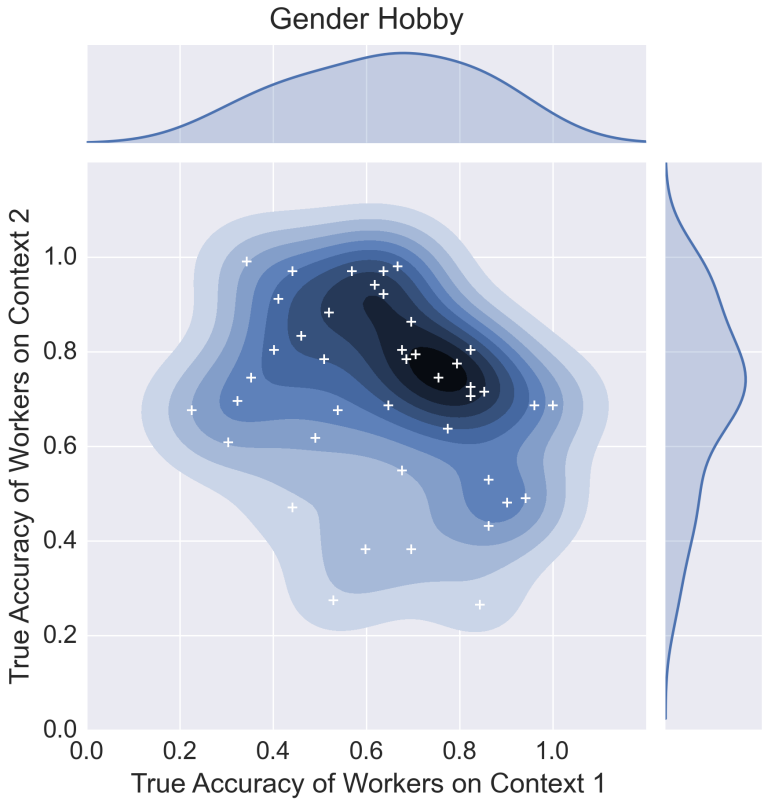
It is also shown in Figure 2.11 that BBTA with $N' = 1$ converges to the highest accuracy slightly faster than others. This is important in practice especially in the budget-sensitive setting, because achieving higher accuracy within a lower budget is always favored from the perspective of requesters in crowdsourcing.

Gender Hobby Dataset

The second real dataset we use is *Gender Hobby* (GH) collected from MTurk by Mo et al. (2013). Tasks in this dataset are binary questions that are explicitly divided into two contexts ($S = 2$): sports and makeup/cooking. This is a typical heterogeneous dataset, where there are $N = 204$ tasks (102 per context) and $K = 42$ workers.



(a)



(b)

Figure 2.12: Distribution of true accuracy of workers for GH data ($N = 204, K = 42, S = 2$).

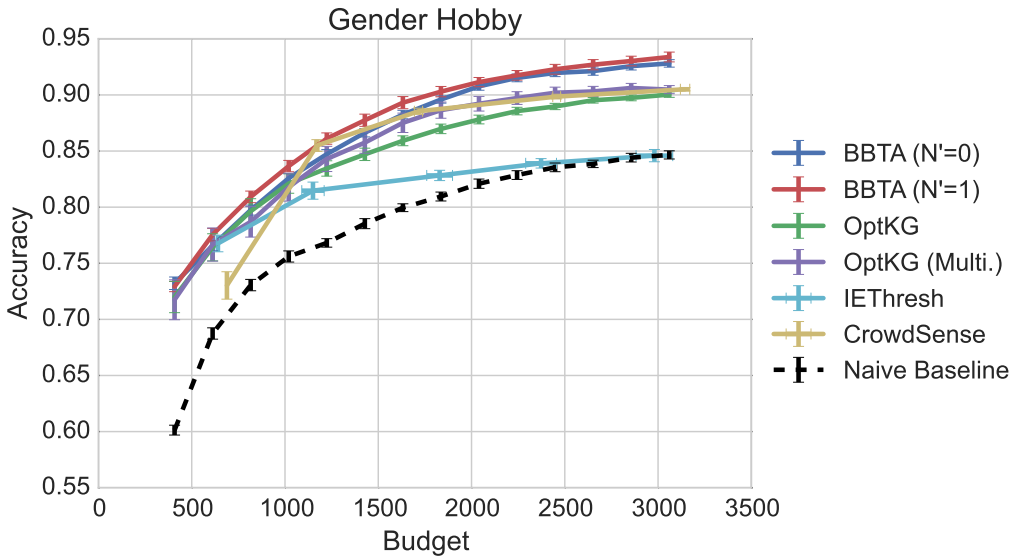


Figure 2.13: Results on GH data ($N = 204, K = 42, S = 2$).

The label matrix in the original GH data is also incomplete, and contains 4200 non-zero entries (i.e. the density of this matrix is $4200/(204 \times 42) \approx 49\%$). Then we decided to use the matrix completion algorithm again to fill the missing entries.

Figure 2.12(a) illustrates the distribution of the true accuracy of workers in this dataset after matrix completion. It is easy to see that the labels given by the workers in this dataset are more malicious than those in the RTE data (Figure 2.10) due to the increased diversity of tasks.

Figure 2.12(b) shows the joint distribution of the true accuracy of workers on two contexts in this dataset. We can see there are only few workers reliable (e.g. accuracy is higher than 80%) on both contexts. We can also see that the workers who are reliable on one context tend to be unreliable on the other. This reflects the characteristic of heterogeneity of this dataset.

Figure 2.13 plots the experimental results, showing that BBTA with $N' = 0$ and $N' = 1$ outperform others on this typical heterogeneous dataset.

Discussion

Now let us have more discussion about the usage of two real data sets (RTE and GH) in our experiments.

Recall that we need the complete matrix of worker labels (i.e. each worker gives a label for each task) to run any online-style method in our experiments. Ideally, we expect that all these labels are from real workers. However, this is not practical since the original RTE and GH are both incomplete (which is the common case in real-world crowdsourcing). In order to feasibly perform our experiments, we use the matrix completion algorithm to fill the missing entries (which is a standard way). Then the data sets used in our experiments actually consist of both real labels and generated labels, and the proportions of real labels in RTE and GH are 6% and 49% respectively. So it is true that RTE and GH used in our experiments are not completely "real". However, this is the best way we can consider to feasibly conduct our experiments on RTE and GH.

It is worth noting that the experimental results on RTE and GH could be treated differently, in terms of "how real the data is". For example, one could argue that the results on GH are much "closer" to the realistic case than those on RTE, because the proportion of real labels in GH is 49% while in RTE it is 6%.

2.6 Proof of Theoretical Results

In this section, we provide our proof for Lemma 2.2. This proof is following the framework of regret analysis for adversarial bandits by Bubeck (2010).

Recall that the unbiased estimator of loss is

$$\widetilde{l}_{j,t} = \frac{l_{j,t}}{p_{j,t}} \mathbb{1}_{j=j_t}.$$

Then we have

$$\mathbb{E}_{j\sim p_t}\left[\widetilde{l}_{j,t}\right] = \sum_{j=1}^{K}\left(p_{j,t}\frac{l_{j,t}}{p_{j,t}}\mathbb{1}_{j=j_t}\right) = l_{j_t,t},$$

$$\mathbb{E}_{j_t\sim p_t}\left[\widetilde{l}_{j',t}\right] = \sum_{j=1}^{K}\left(p_{j,t}\frac{l_{j',t}}{p_{j',t}}\mathbb{1}_{j'=j}\right) = l_{j',t},$$

which imply

$$\sum_{t=1}^{T_2} l_{j_t,t} - \sum_{t=1}^{T_2} l_{j',t} = \sum_{t=1}^{T_2}\mathbb{E}_{j\sim p_t}\left[\widetilde{l}_{j,t}\right] - \sum_{t=1}^{T_2}\mathbb{E}_{j_t\sim p_t}\left[\widetilde{l}_{j',t}\right]. \tag{2.4}$$

Now we rewrite $\mathbb{E}_{j\sim p_t}\left[\widetilde{l}_{j,t}\right]$ in the first term of (2.4) as

$$\mathbb{E}_{j\sim p_t}\left[\widetilde{l}_{j,t}\right] = \frac{1}{\eta_t}\ln \mathbb{E}_{j\sim p_t}\left[\exp\left(-\eta_t\left(\widetilde{l}_{j,t} - \mathbb{E}_{j'\sim p_t}\left[\widetilde{l}_{j',t}\right]\right)\right)\right] - \frac{1}{\eta_t}\ln \mathbb{E}_{j\sim p_t}\left[\exp\left(-\eta_t\widetilde{l}_{j,t}\right)\right]. \tag{2.5}$$

Then the first term of (2.5) is

$$\begin{aligned}
\frac{1}{\eta_t}\ln \mathbb{E}_{j\sim p_t}\left[\exp\left(-\eta_t\left(\widetilde{l}_{j,t} - \mathbb{E}_{j'\sim p_t}\left[\widetilde{l}_{j',t}\right]\right)\right)\right] &= \frac{1}{\eta_t}\ln \mathbb{E}_{j\sim p_t}\left[\exp\left(-\eta_t\widetilde{l}_{j,t}\right)\right] + \mathbb{E}_{j'\sim p_t}\left[\widetilde{l}_{j',t}\right] \\
&\leq \frac{1}{\eta_t}\mathbb{E}_{j\sim p_t}\left[\exp\left(-\eta_t\widetilde{l}_{j,t}\right) - 1\right] + l_{j_t,t} \\
&= \frac{1}{\eta_t}\mathbb{E}_{j\sim p_t}\left[\exp\left(-\eta_t\widetilde{l}_{j,t}\right) - 1 + \eta_t\widetilde{l}_{j,t}\right] \\
&\leq \frac{1}{\eta_t}\mathbb{E}_{j\sim p_t}\left[\frac{\eta_t^2\widetilde{l}_{j,t}^2}{2}\right] \\
&\leq \frac{\eta_t}{2p_{j_t,t}},
\end{aligned} \tag{2.6}$$

where the first and second inequalities are due to $\ln x \leq x - 1$ for $x > 0$ and $\exp(x) \leq 1 + x + x^2/2$ for $x \leq 0$ respectively, and the third inequality is because

$$\mathbb{E}_{j\sim p_t}\left[\widetilde{l}_{j,t}^2\right] = \sum_{j=1}^{K} p_{j,t}\left(\frac{l_{j,t}}{p_{j,t}}\mathbb{1}_{j=j_t}\right)^2 = \frac{l_{j_t,t}^2}{p_{j_t,t}} \leq \frac{1}{p_{j_t,t}}.$$

We define the potential function as

$$\Phi_t(\eta) = \frac{1}{\eta}\ln\frac{1}{K}\sum_{j=1}^{K}\exp\left(-\eta L_{j,t}\right).$$

Then the second term of (2.5) is

$$\begin{aligned}
-\frac{1}{\eta_t}\ln \mathbb{E}_{j\sim p_t}\left[\exp\left(-\eta_t \widetilde{l}_{j,t}\right)\right] &= -\frac{1}{\eta_t}\ln \sum_{j=1}^{K} p_{j,t}\exp\left(-\eta_t \widetilde{l}_{j,t}\right) \\
&= -\frac{1}{\eta_t}\ln \sum_{j=1}^{K} \frac{w_{j,t}}{\sum_{j'=1}^{K} w_{j',t}}\exp\left(-\eta_t \widetilde{l}_{j,t}\right) \\
&= -\frac{1}{\eta_t}\ln \frac{\sum_{j=1}^{K}\exp\left(-\eta_t L_{j,t-1} - \eta_t \widetilde{l}_{j,t}\right)}{\sum_{j'=1}^{K}\exp\left(-\eta_t L_{j',t-1}\right)} \\
&= -\frac{1}{\eta_t}\ln \frac{\sum_{j=1}^{K}\exp\left(-\eta_t L_{j,t}\right)}{\sum_{j'=1}^{K}\exp\left(-\eta_t L_{j',t-1}\right)} \\
&= \Phi_{t-1}(\eta_t) - \Phi_t(\eta_t). \qquad (2.7)
\end{aligned}$$

Combining (2.4), (2.5), (2.6), and (2.7), we obtain

$$\sum_{t=1}^{T_2} l_{j_t,t} - \sum_{t=1}^{T_2} l_{j',t} \le \sum_{t=1}^{T_2}\frac{\eta_t}{2p_{j_t,t}} + \sum_{t=1}^{T_2}\left(\Phi_{t-1}(\eta_t) - \Phi_t(\eta_t)\right) - \sum_{t=1}^{T_2}\mathbb{E}_{j_t\sim p_t}\left[\widetilde{l}_{j',t}\right]. \qquad (2.8)$$

Since we have

$$\mathbb{E}_{j_t\sim p_t}\left[\frac{1}{p_{j_t,t}}\right] = \sum_{j=1}^{K} p_{j,t}\frac{1}{p_{j,t}} = K,$$

then the expectation of the first term of (2.8) is

$$\mathbb{E}\left[\sum_{t=1}^{T_2}\frac{\eta_t}{2p_{j_t,t}}\right] = \frac{1}{2}\mathbb{E}\left[\sum_{t=1}^{T_2}\mathbb{E}_{j_t\sim p_t}\left[\frac{\eta_t}{p_{j_t,t}}\right]\right] = \frac{K}{2}\sum_{t=1}^{T_2}\eta_t. \qquad (2.9)$$

The second term of (2.8) is transformed into

$$\sum_{t=1}^{T_2}\left(\Phi_{t-1}(\eta_t) - \Phi_t(\eta_t)\right) = \Phi_0(\eta_1) + \sum_{t=1}^{T_2-1}\left(\Phi_t(\eta_{t+1}) - \Phi_t(\eta_t)\right) - \Phi_{T_2}(\eta_{T_2}). \qquad (2.10)$$

By the definition of the potential function, we have

$$
\begin{aligned}
\Phi_0(\eta_1) &= \frac{1}{\eta_1} \ln \frac{1}{K} \sum_{j=1}^{K} \exp\left(-\eta_1 L_{j,0}\right) \\
&\leq \frac{1}{\eta_1} \left(\frac{1}{K} \sum_{j=1}^{K} \exp\left(-\eta_1 L_{j,0}\right) - 1 \right) \\
&\leq \frac{1}{\eta_1 K} \left(\sum_{j=1}^{K} \left(\exp\left(-\eta_1 L_{j,0}\right) - 1 + \eta_1 L_{j,0} \right) \right) \\
&\leq \frac{1}{\eta_1 K} \sum_{j=1}^{K} \frac{\eta_1^2 L_{j,0}^2}{2} \\
&= \frac{\eta_1}{2K} \sum_{j=1}^{K} \left(\sum_{i \in \mathcal{I}_1} \mathbb{1}_{y_{i,j} \neq \widehat{y}_i} \right)^2 \\
&\leq \frac{\eta_1}{2K} \sum_{j=1}^{K} \left(N' \sum_{i \in \mathcal{I}_1} \left(\mathbb{1}_{y_{i,j} \neq \widehat{y}_i} \right)^2 \right) \\
&= \frac{\eta_1 N'}{2K} \sum_{i \in \mathcal{I}_1} \sum_{j=1}^{K} \left(\mathbb{1}_{y_{i,j} \neq \widehat{y}_i} \right)^2 \\
&\leq \frac{\eta_1 N'}{2K} \sum_{i \in \mathcal{I}_1} \frac{K}{2} = \frac{\eta_1 N'^2}{4}, \qquad (2.11)
\end{aligned}
$$

where the first and third inequalities are due to $\ln x \leq x - 1$ for $x > 0$ and $\exp(x) \leq 1 + x + x^2/2$ for $x \leq 0$ respectively, the fifth line of (2.11) is obtained by the definition of the cumulative loss $L_{j,0}$ in the pure exploration phase, the fourth inequality of (2.11) is an immediate result of using Jensen's inequality and $|\mathcal{I}_1| = N'$, and the last line of (2.11) is because we use majority voting in the pure

exploration phase. We also have

$$\begin{aligned}
-\Phi_{T_2}(\eta_{T_2}) &= -\frac{1}{\eta_{T_2}} \ln \frac{1}{K} \sum_{j=1}^{K} \exp\left(-\eta_{T_2} L_{j,T_2}\right) \\
&\leq \frac{\ln K}{\eta_{T_2}} - \frac{1}{\eta_{T_2}} \ln\left(\exp\left(-\eta_{T_2} L_{j',T_2}\right)\right) \\
&= \frac{\ln K}{\eta_{T_2}} + L_{j',0} + \sum_{t=1}^{T_2} \widetilde{l}_{j',t} \\
&\leq \frac{\ln K}{\eta_{T_2}} + N' + \sum_{t=1}^{T_2} \widetilde{l}_{j',t}. \qquad (2.12)
\end{aligned}$$

Now we combine (2.8), (2.9), (2.10), (2.11), and (2.12) and obtain

$$\mathbb{E}\left[\sum_{t=1}^{T_2} l_{j_t,t} - \sum_{t=1}^{T_2} l_{j',t}\right] \leq \frac{K}{2} \sum_{t=1}^{T_2} \eta_t + \frac{\eta_1 N'^2}{4} + \frac{\ln K}{\eta_{T_2}} + N' + \mathbb{E}\left[\sum_{t=1}^{T_2-1} \left(\Phi_t(\eta_{t+1}) - \Phi_t(\eta_t)\right)\right].$$

Since it is shown by Bubeck (2010) that $\Phi'(\eta) \geq 0$ and we set $\eta_t = \sqrt{\frac{\ln K}{tK}}$, we have

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T_2} l_{j_t,t} - \sum_{t=1}^{T_2} l_{j',t}\right] &\leq \frac{K}{2} \sum_{t=1}^{T_2} \sqrt{\frac{\ln K}{tK}} + \frac{N'^2}{4} \sqrt{\frac{\ln K}{K}} + \sqrt{T_2 K \ln K} + N' \\
&\leq 2\sqrt{(T - KN')K \ln K} + \frac{N'^2}{4} \sqrt{\frac{\ln K}{K}} + N', \qquad (2.13)
\end{aligned}$$

where the second inequality is because $T_2 = T - KN'$ and

$$\sum_{t=1}^{T_2} \frac{1}{\sqrt{t}} \leq 2\sqrt{T_2}.$$

(2.13) shows the bound about our strategy competing against *any* strategy that always selects one single worker, thus concluding the proof.

Chapter 3

Task Selection in Crowdsourcing

Task selection is the other important part of the task-worker assignment problem in crowdsourcing. In this chapter, we develop several strategies for task selection in crowdsourcing. We also embed these strategies into the *bandit-based task assignment* (BBTA) method, and show that the performance of BBTA can be further improved by choosing appropriate task selection strategies.

3.1 Introduction

The recent rise of crowdsourcing has provided us with a very useful tool for combining the wisdom of crowds. Especially in many machine learning tasks, such as image labeling in computer vision (Sorokin and Forsyth, 2008) and recognizing textual entailment in natural language processing (Snow et al., 2008), crowdsourcing has become a very efficient way of obtaining large amounts of labeled data.

Compared with collecting labels from the expert, crowdsourcing is a cheaper and faster choice for label collection. Although a single label from a worker is very cheap, the requester still wants to wisely use her (possibly limited) budget for label collection. This motivates us to consider the task-worker assignment problem in crowdsourcing.

Task-worker assignment is concerning online decision making on which task is to assign to which worker. It naturally involves two aspects: worker selection and task selection. A worker selection strategy is to determine which worker to select given any task. Chapter 2 proposed the *bandit-based task assignment*

(BBTA) method, and successfully addressed the problem of worker selection in crowdsourcing.

As well as worker selection, task selection is also involved in the task-worker assignment problem. It is the problem of deciding how to pick an appropriate task at each round. A good strategy of task selection may help us efficiently use the budget when assigning the tasks to workers. For example, if we are already confident in the aggregated label of one certain task, we should not require more labels for this task, otherwise it would be a waste of budget. Although a common uncertainty criterion is used for picking tasks in BBTA, task selection is worth further investigating.

In this chapter, we look into task selection strategies for task-worker assignment in heterogeneous crowdsourcing. In particular, we extend BBTA by using different strategies of picking tasks. The ideas of these strategies are borrowed from *query strategies* in *active learning* literature (Settles, 2009). We embed several task selection strategies into BBTA, one of which is equivalent to the uncertainty criterion used in the original BBTA. We experimentally evaluate these strategies for BBTA, and demonstrate that the performance of BBTA can be further improved by adopting appropriate task selection strategies such as the *least confidence* strategy.

3.2 Task Selection

In this section, we investigate several different strategies of selecting tasks for task-worker assignment in crowdsourcing.

3.2.1 Problem Setting and Notation

We assume there are N unlabeled tasks, K workers, and S contexts. The task is indexed by i, where $i \in [N] = \{1, 2, \ldots, N\}$. The worker is indexed by j, where $j \in [K] = \{1, 2, \ldots, K\}$. Each task is characterized by a context s, where $s \in [S] = \{1, 2, \ldots, S\}$. For simplicity, we consider binary labels, i.e., the label space $\mathcal{C} = \{-1, +1\}$. Each time given a task, we ask one worker from a pool of K workers for a (possibly noisy) label, consuming one unit of the total budget

T. The goal is to find suitable task-worker assignment to collect as many reliable labels as possible within the limited budget T. Finally, we estimate the true labels by aggregating the collected labels.

Let $y_{i,j}$ be the individual label of task i (with context s) given by worker j. If the label is missing (i.e. we did not collect the label of task i from worker j), we set $y_{i,j} = 0$. We denote the weight of worker j for context s by w_j^s, corresponding to the context-dependent reliability of worker j for context s. To avoid complex notation, we do not use the subscript for s henceforth, which implicitly represents the context of the current task i.

3.2.2 Task Selection Strategies

In this chapter, we develop three task selection strategies: *least confidence*, *margin sampling* and *information density*. The ideas of these strategies are borrowed from query strategies, which are the common frameworks for measuring the *informativeness* of data instances in active learning literature (Settles, 2009). For example, at each step of an active learning algorithm, the most *uncertain* instance is sampled for training, where the *uncertainty* is calculated according to some query strategy.

Usually, query strategies in active learning are designed for *probabilistic* models. In this chapter, we introduce these strategies into our task-worker assignment problem, where the weighted voting mechanism is *non-probabilistic*. Although we do not have *probabilities* of labels as in probabilistic models, we instead define a confidence score for each task, to describe how confident we are in the aggregated label of this task.

For convenience, we rewrite the weighted voting mechanism as

$$\widehat{y_i} = \underset{c \in \{-1,+1\}}{\arg\max} \frac{\sum_{j=1}^{K} w_j^s \cdot \mathbb{1}_{y_{i,j}=c}}{\sum_{j'=1}^{K} w_{j'}^s},$$

which is equivalent to Eqn. 2.1 in Section 2.2. We further define the *positive score* as:

$$\overline{y}_i^+ = \frac{\sum_{j=1}^{K} w_j^s \cdot \mathbb{1}_{y_{i,j}=+1}}{\sum_{j'=1}^{K} w_{j'}^s},$$

and similarly the *negative score* as:

$$\overline{y}_i^- = \frac{\sum_{j=1}^K w_j^s \cdot \mathbb{1}_{y_{i,j}=-1}}{\sum_{j'=1}^K w_{j'}^s}.$$

It is easy to see that $\overline{y}_i^+, \overline{y}_i^- \in [0, 1]$. For task i, the positive/negative score will be 1 if and only if we have collected labels from *all* workers and *all* of them are consistent. On the other hand, when we have no positive/negative label for task i, the positive/negative score will be 0.

At each round of the adaptive assignment phase in BBTA, we pick a task according to the *confidence score* $\overline{y}_i$, which is calculated by using the positive score and the negative score based on the specific task selection strategy we adopt. For convenience, the definition of $\overline{y}_i$ in this chapter is slightly different from that in Chapter 2. The details of task selection strategies are presented below.

3.2.3 Least Confidence

The strategy called *least confidence* (LC) corresponds to probably the most commonly used framework of *uncertainty sampling* in active learning (Lewis and Gale, 1994).

Specifically, at round t of the adaptive assignment phase, we pick the task with the index:

$$i_t = \arg\min_{i \in \mathcal{I}_2} \overline{y}_i, \text{ where } \overline{y}_i = \max(\overline{y}_i^+, \overline{y}_i^-).$$

This is a straightforward way to pick a task. Here, the confidence score $\overline{y}_i$ is simply set as the larger one between the positive score $\overline{y}_i^+$ and the negative score $\overline{y}_i^-$. That is, it is the score of the more probable label (-1 or $+1$). Then we pick the task i_t with the lowest confidence score, indicating that among all tasks we are the least confident in the aggregated label of task i_t, and thus we require one more label for this task at round t.

3.2.4 Margin Sampling

The strategy called *Margin sampling* (MS) is originally used for multi-class uncertainty sampling in active learning (Scheffer et al., 2001). Instead of only consid-

ering the information about the most probable label (as in LC), MS also involves the information about the remaining label(s).

We introduce this strategy into our task-worker assignment problem as

$$i_t = \arg\min_{i \in \mathcal{I}_2} \overline{y}_i, \text{ where } \overline{y}_i = \left| \overline{y}_i^+ - \overline{y}_i^- \right|.$$

It is easy to see that this strategy is equivalent to the uncertainty criterion (Eqn. 2.3) used in the original BBTA method. Basically, the confidence score $\overline{y}_i$ here is the absolute value of the difference between the sum of (normalized) weights for positive labels and that for negative labels. The uncertainty in MS can be interpreted as ambiguity. The intuition is that tasks with smaller margins are more ambiguous, and ambiguous tasks usually require more labels.

3.2.5 Information Density

It is argued by Settles and Craven (2008) that the informativeness of instances should not only be measured by the uncertainty, but also by the *representativeness* of the underlying distribution. This is the motivation of the strategy called *Information Density* (ID) (Settles and Craven, 2008).

Similarly, in our heterogeneous crowdsourcing setting, we can also consider the proportions of different task types as well as the confidence scores when picking tasks. We adopt the idea of ID and develop the task selection strategy as

$$i_t = \arg\max_{i \in \mathcal{I}_2} (1 - \overline{y}_i) \left(\frac{N^s}{N} \right)^\beta,$$

where $\overline{y}_i$ is calculated as in LC or MS, s is the type (i.e. context) of task i, and N^s is the total number of tasks with type s. The parameter β controls the relative importance of the density term (i.e. the proportion of tasks with context s). When β approaches 0, ID strategy degrades to LC or MS (depending on how we calculate $\overline{y}_i$). As β gets larger, the density term becomes more important.

3.3 Experiments

In this section, we experimentally evaluate different task selection strategies for BBTA. Since the margin sampling strategy is equivalent to the uncertainty crite-

rion in the original BBTA, for the purpose of fair comparison, we use the same experimental setup as that in Chapter 2.

We first conduct experiments on benchmark data with simulated workers, and then use real data for further comparison. All of the experimental results are averaged over 30 runs.

3.3.1 Benchmark Data

We again use three popular UCI benchmark datasets[1]: ionosphere ($N = 351$), breast ($N = 569$), and pima ($N = 768$). To simulate the heterogeneous scenario, we first use *k-means* to create $S = 3, 4, 5$ contexts for these three datasets respectively, and then generate crowd labels by using three worker models: spammer-hammer, one-coin, and one-coin (malicious). The numbers of simulated workers are $K = 30, 40, 50$ respectively.

See Section 2.5 of Chapter 2 for more details of the experimental setup.

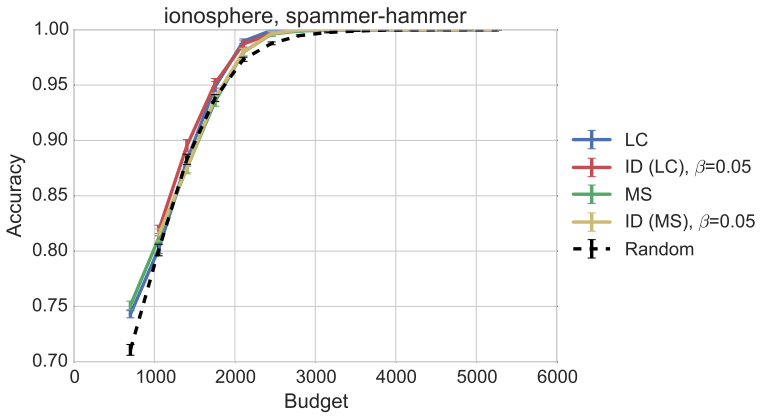
Comparison Experiments

We compare the least confidence (LC), margin sampling (MS) and information density (ID) strategies for BBTA in terms of accuracy. Accuracy is calculated as the proportion of correct estimates for true labels.

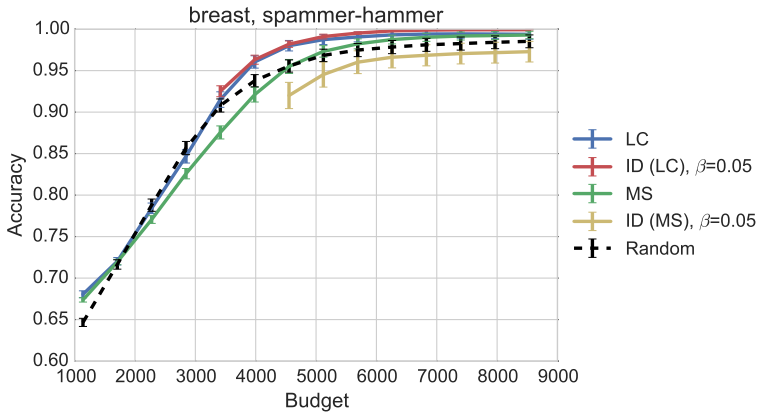
Recall that N' is the parameter for pure exploration in BBTA. We set $N' = 1$ for all strategies. As mentioned in Section 3.2.4, MS is equivalent to the uncertainty criterion used in the original BBTA method. For ID, we have ID (LC) and ID (MS), indicating that we use LC and MS respectively to calculate confidence scores. We set the parameter β to 0.05 in both ID (LC) and ID (MS), which is a reasonable choice as shown later. We also implement a random strategy as a baseline, where we randomly pick a task at each step without using any uncertainty criterion. Accuracy of all strategies is compared at different levels of budget. We set the maximum amount of budget at $T = 15N$.

Figure 3.4, 3.5 and 3.6 show the averages and standard errors of accuracy as functions of budgets for all strategies in nine cases (i.e. three datasets with three worker models). Generally, LC and ID (LC) can be considered as the best two

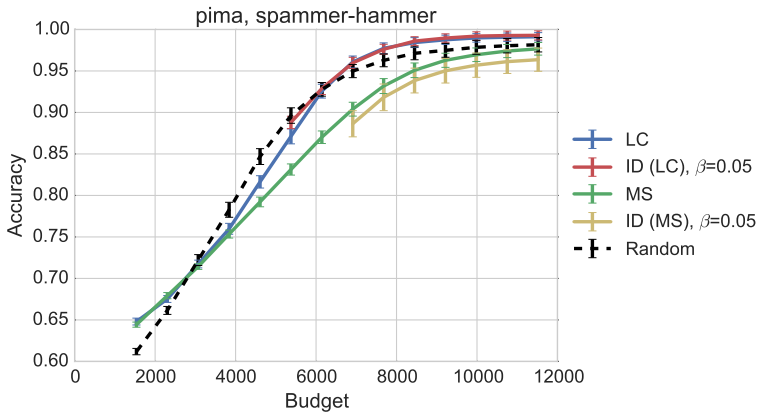
[1] http://archive.ics.uci.edu/ml/



(a) $N = 351, K = 30, S = 3$

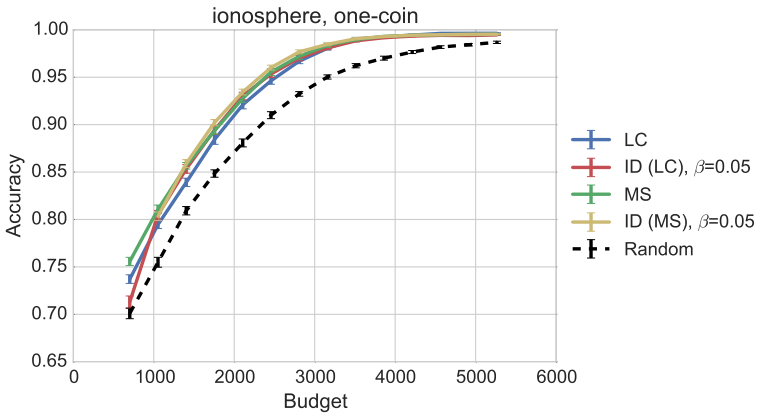


(b) $N = 569, K = 40, S = 4$

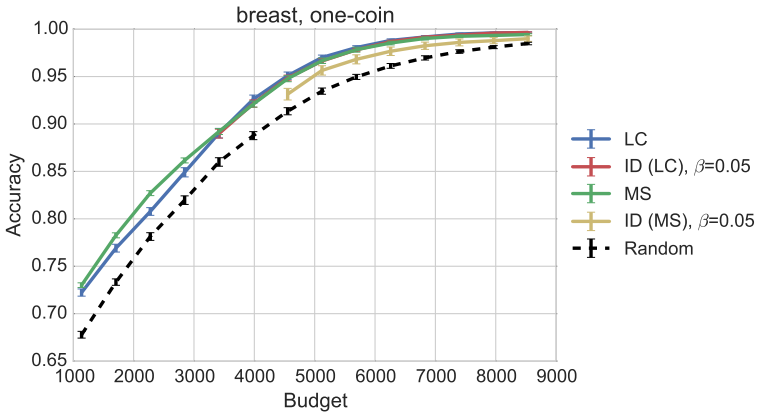


(c) $N = 768, K = 50, S = 5$

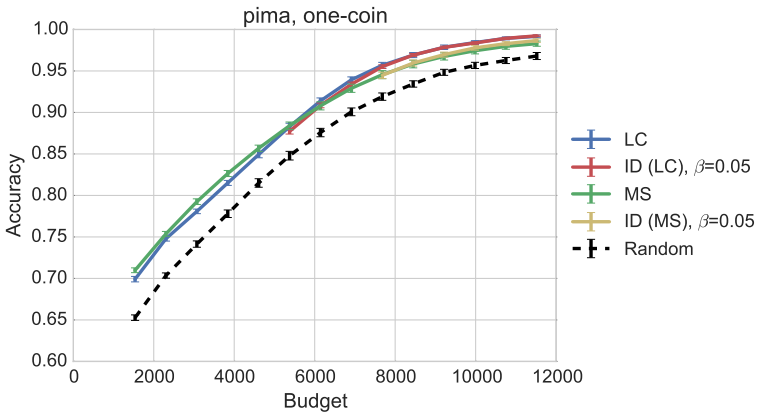
Figure 3.1: Results of different task selection strategies on three benchmark datasets with the spammer-hammer worker model.



(a) $N = 351, K = 30, S = 3$

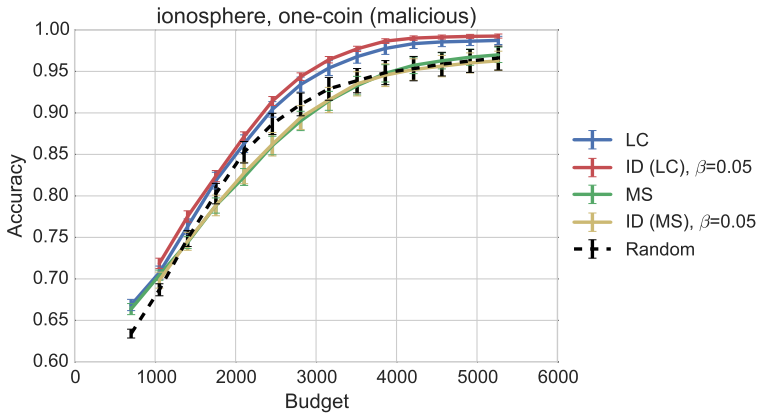


(b) $N = 569, K = 40, S = 4$

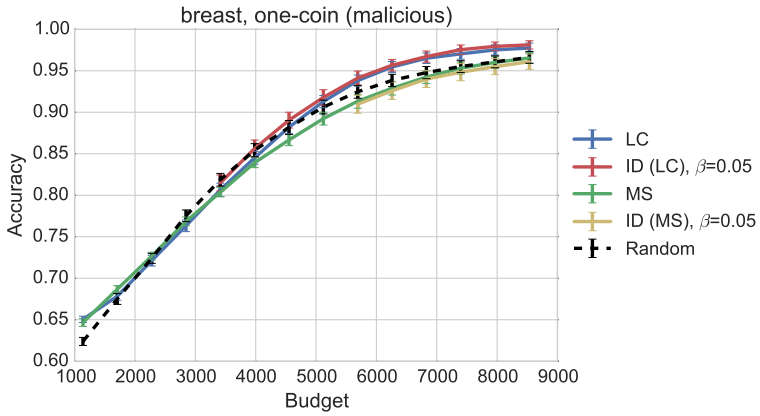


(c) $N = 768, K = 50, S = 5$

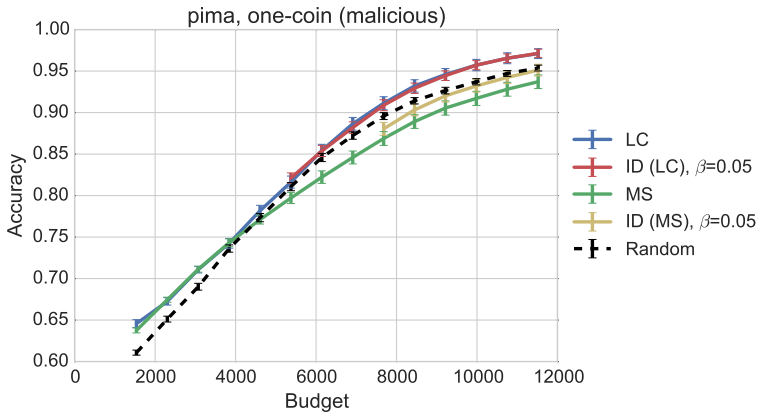
Figure 3.2: Results of different task selection strategies on three benchmark datasets with the one-coin worker model.



(a) $N = 351, K = 30, S = 3$



(b) $N = 569, K = 40, S = 4$



(c) $N = 768, K = 50, S = 5$

Figure 3.3: Results of different task selection strategies on three benchmark datasets with the one-coin worker model (malicious).

on these benchmark data, while ID (LC) is slightly better than LC in some cases. Surprisingly, MS and ID (MS) perform even worse than the random strategy in many cases. This implies that the uncertainty criterion used in the original BBTA may not be an appropriate task selection strategy.

Experiments with Varying β for the Information Density Strategies

We further check the effects of varying the parameter β in ID. The accuracy is calculated when we consume all the budget, corresponding to the rightmost points on the associated curves in Figure 3.1, 3.2 and 3.3. For some larger β, we do not have the associated accuracy. This is because there exist some tasks that have never been assigned to any worker when β is larger. We consider these cases as *assignment failure* and do not calculate the accuracy for them. We also plot LC, MS and the random strategy for reference (dash curves without error bars).

Figure 3.4, 3.5 and 3.6 show the results. Generally, ID (LC) is better than ID (MS). Roughly speaking, the good choice of β appears to be in the range between 0.01 and 0.1.

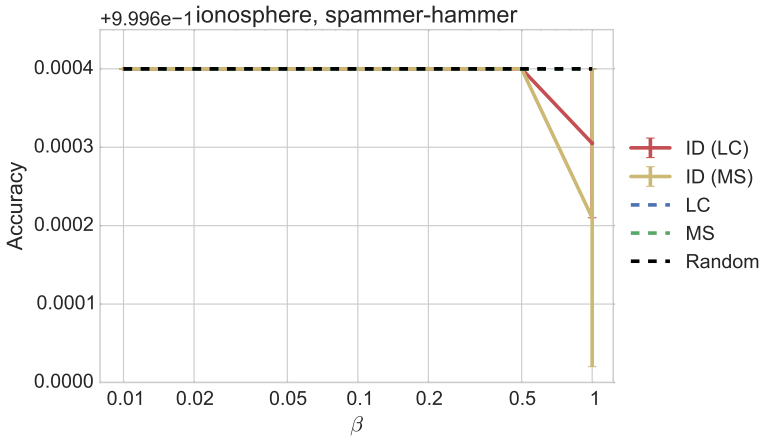
3.3.2 Real Data

Next, we compare different task selection strategies for BBTA on two real-world datasets: *Recognizing Textual Entailment* (RTE) (Snow et al., 2008) and *Gender Hobby* (GH) (Mo et al., 2013). The numbers of tasks in these two datasets are $N = 800, 204$ respectively. RTE is a homogeneous dataset, so the context number is $S = 1$; while GH is a typical heterogeneous dataset with $S = 2$ contexts. The labels of these two dataset are collected in Amazon Mechanical Turk, from $K = 164, 42$ workers respectively.

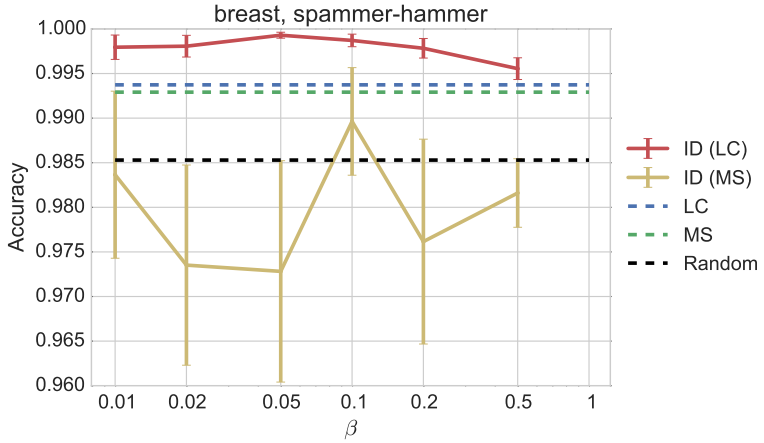
For more details of experimental setup, see Section 2.5 of Chapter 2.

Comparison Experiments

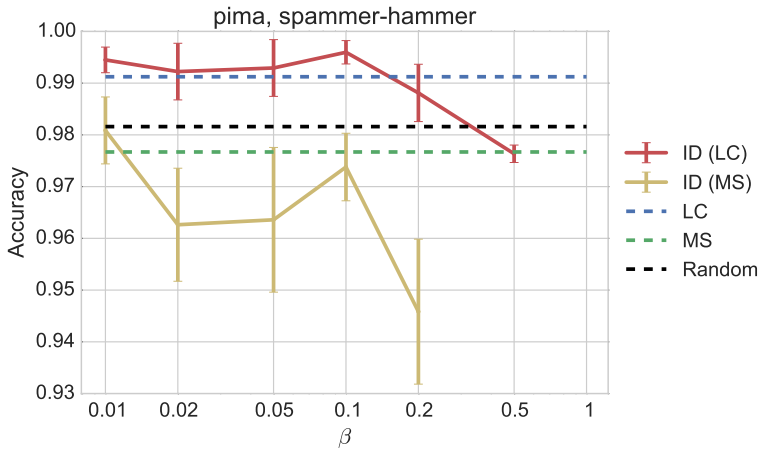
Figure 3.7(a) depicts the comparison results on the RTE data, showing that LC and MS are comparable, and both of them are better than the random strategy. Since this is a homogeneous dataset, there is only one task type and its proportion



(a) $N = 351, K = 30, S = 3$

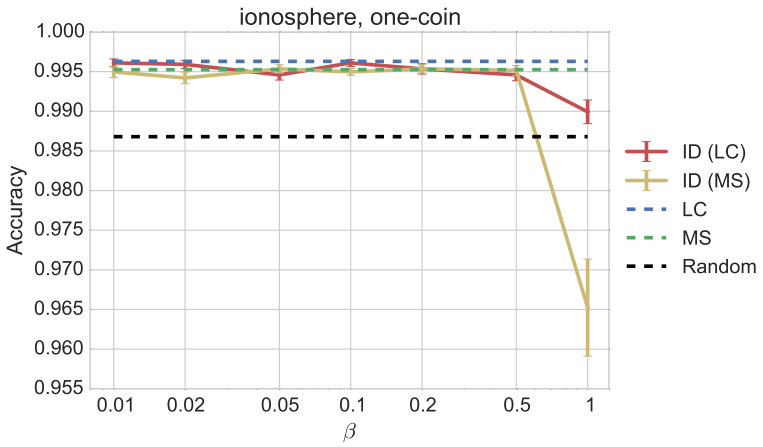


(b) $N = 569, K = 40, S = 4$

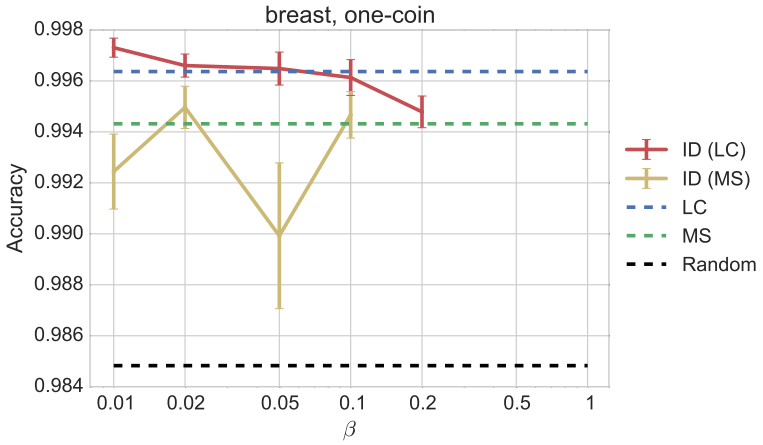


(c) $N = 768, K = 50, S = 5$

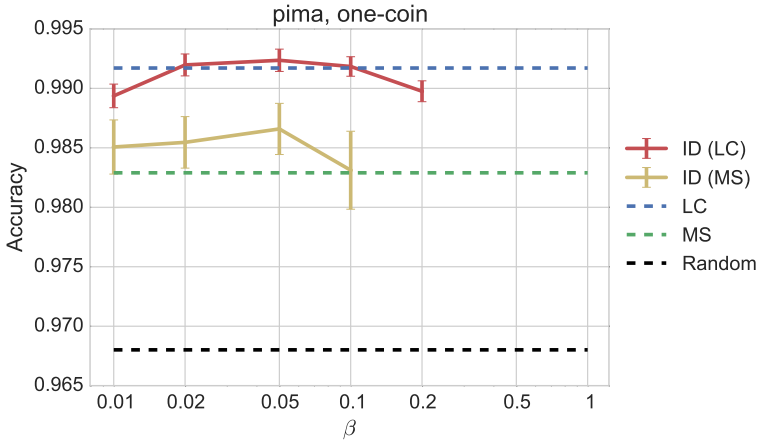
Figure 3.4: Results of varying β in ID strategy on three benchmark datasets with the spammer-hammer worker model.



(a) $N = 351, K = 30, S = 3$

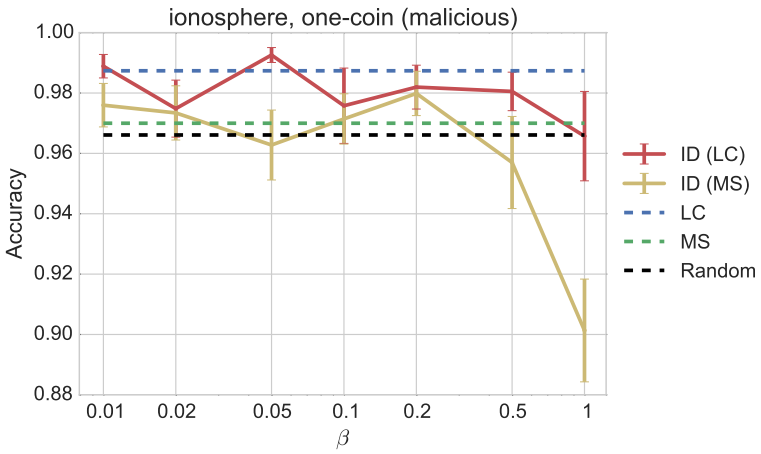


(b) $N = 569, K = 40, S = 4$

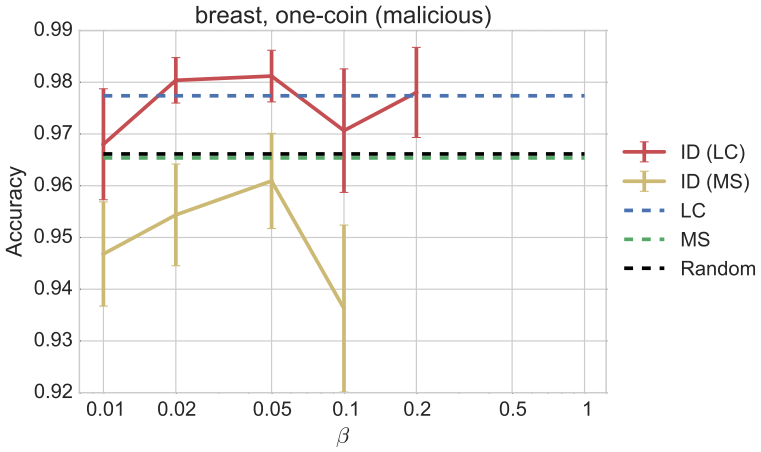


(c) $N = 768, K = 50, S = 5$

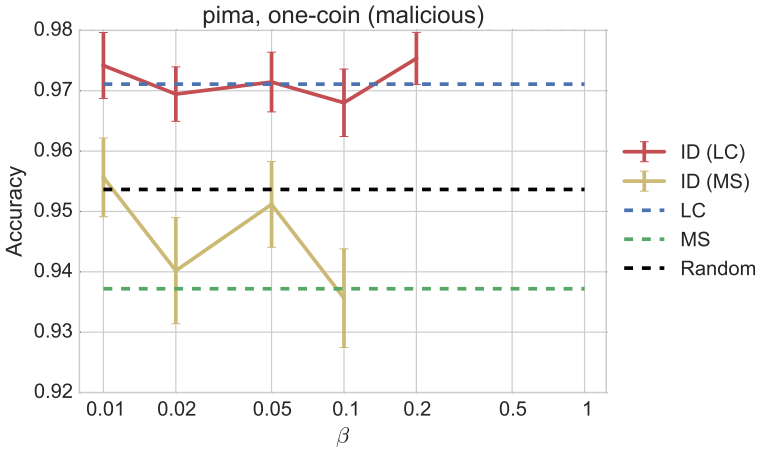
Figure 3.5: Results of varying β in ID strategy on three benchmark datasets with the one-coin worker model.



(a) $N = 351, K = 30, S = 3$

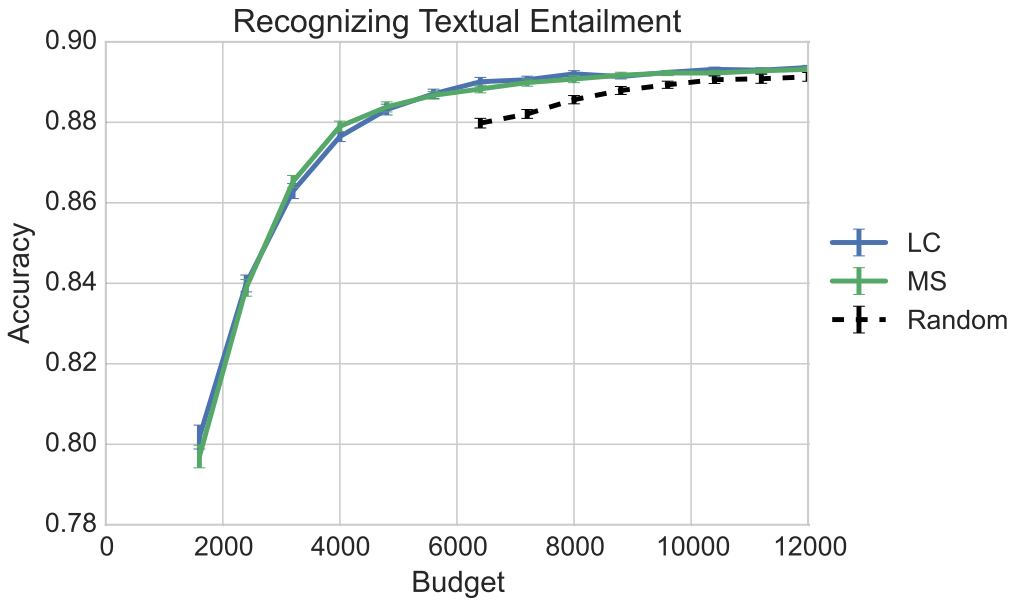


(b) $N = 569, K = 40, S = 4$

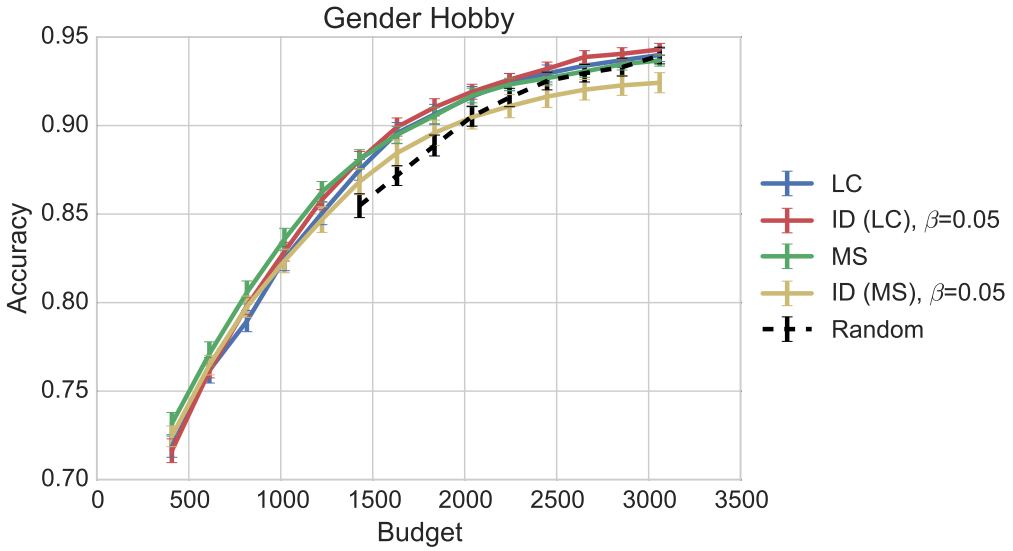


(c) $N = 768, K = 50, S = 5$

Figure 3.6: Results of varying β in ID strategy on three benchmark datasets with the one-coin worker model (malicious).



(a) $N = 800, K = 164, S = 1$



(b) $N = 204, K = 42, S = 2$

Figure 3.7: Results on two real-world datasets.

is 100%. Then ID (LC) and ID (MS) are equivalent to LC and MS respectively and we do not need to plot curves for ID (LC) and ID (MS).

Figure 3.7(b) plots the experimental results, showing that ID (LC) performs the best on this typical heterogeneous dataset, and LC follows. Note that the performance of ID (MS) is even worse than that of the random strategy on this dataset.

Chapter 4

Conclusions and Future Work

In this chapter, we summarize the major contributions of this thesis and introduce several future directions.

4.1 Conclusions

This thesis was devoted to developing statistical learning approaches to the task-worker assignment problem in crowdsourcing. The task-worker assignment problem consists of two important aspects: worker selection and task selection. Chapter 2 and Chapter 3 tackled these two aspects respectively.

In Chapter 2, we proposed a contextual bandit formulation to address the problem of worker selection in heterogeneous crowdsourcing. In the proposed method called *bandit-based task assignment* (BBTA), we first explored workers' reliability and then attempted to adaptively assign tasks to appropriate workers. We used the exponential weighting scheme to handle the exploration-exploitation trade-off in worker selection and utilized the weighted voting mechanism to aggregate the collected labels. Thanks to the contextual formulation, BBTA models the context-dependent reliability for workers, and thus is able to intelligently match workers to tasks they can do best. This is a significant advantage over the state-of-the-art online-style methods in crowdsourcing. We experimentally showed the usability of BBTA in heterogeneous crowdsourcing. We also theoretically investigated the regret bounds for BBTA. In the regret analysis, we showed the performance of our strategy converges to that of the optimal one as the budget goes to infinity.

In Chapter 3, we further investigated task selection strategies for task-worker assignment in heterogeneous crowdsourcing. While the existing method BBTA focused on the worker selection strategy in task-worker assignment, we further extended it by adopting different task selection strategies for picking tasks. The experimental results showed that the performance of BBTA can be further improved by using more appropriate strategies, such as *least confidence* (LC) and its *information density* (ID) variant. ID involves tuning the importance parameter β, which could be cumbersome in practice, while LC does not require tuning any parameter, and its performance is comparable to that of ID in most cases. Therefore, a practical choice is to adopt LC as the task selection strategy in BBTA.

4.2 Future Work

In this thesis, we have demonstrated that the proposed strategies of worker selection and task selection are effective for crowdsourcing. However, there is still a lot of room for further work in this field.

In the final section of this thesis, we introduce several promising future directions in crowdsourcing. We discuss the future work from two perspectives: within BBTA, and beyond BBTA.

4.2.1 Within BBTA

The proposed method bandit-based task assignment (BBTA) has been shown to be successful in crowdsourcing, together with the proposed strategies of task selection. There are still some possible extensions of BBTA, which we describe as follows.

Extensions to the Multi-Class Setting

It is shown that classification tasks are the most commonly posted in crowdsourcing marketplaces (Ipeirotis, 2010). In Chapter 2 and Chapter 3, we consider binary classification tasks for simplicity. However, multi-class classification tasks, where the label space contains more than 3 classes, are also common in real-world crowdsourcing.

Fortunately, it is straightforward to extend BBTA to the multi-class setting, by simply changing the current weighted voting mechanism to

$$\widehat{y}_i = \arg\max_{c \in \mathcal{C}} \sum_{j=1}^{K} w_j^s \cdot \mathbb{1}_{y_{i,j}=c},$$

where $\mathcal{C}$ is the multi-class label space.

For the strategies of task selection, least confidence (LC) turns to

$$i_t = \arg\min_{i \in \mathcal{I}_2} \overline{y}_i, \text{ where } \overline{y}_i = \max_{c \in \mathcal{C}} \overline{y}_i^c.$$

Recall that we have the positive score $\overline{y}_i^+$ and the negative score $\overline{y}_i^-$ in the binary setting, while we define the score $\overline{y}_i^c$ for each class $c \in \mathcal{C}$ in the multi-class setting. Then the confidence score $\overline{y}_i$ is the score of the most probable class label in $\mathcal{C}$. Similarly, the margin sampling (MS) strategy can also be adapted to the multi-class setting as

$$i_t = \arg\min_{i \in \mathcal{I}_2} \overline{y}_i, \text{ where } \overline{y}_i = \left|\overline{y}_i^{c_1} - \overline{y}_i^{c_2}\right|.$$

Here, c_1 and c_2 are the first and second most probable class labels respectively. Since the information density (ID) strategy is based on LC or MS, it can be directly used in the multi-class setting.

In the binary setting, LC is better than MS in general, as shown in the experiments in Section 3.3 of Chapter 3. It is very interesting to think about whether this still holds in the multi-class setting, where the label space increases. LC only considers the most likely class label, but neglects the information of the remaining label distribution; whereas MS was originally proposed for multi-class uncertainty sampling to alleviate the drawback of LC (Scheffer et al., 2001). However, it is unclear whether MS can perform better in our multi-class setting. Therefore, it would be a promising direction to investigate and find more problems in the multi-class setting.

Theoretical Challenges

The regret analysis in Section 2.3 of Chapter 2 has shown that BBTA performs as well as the optimal strategy of worker selection when the budget goes to infinity.

Broadly speaking, this important theoretical analysis helps us specify the step size parameter η in the exponential weighting scheme (Line 19 in Algorithm 1). On the other hand, how to choose the parameter N', which is related to the length of pure exploration, also requires the theoretical guidance. Although we conducted the experiments of varying N' for BBTA, and gave an empirical choice of N', it is still very interesting to consider how to theoretically analyze the pure exploration phase for choosing N'.

Another point in the current theoretical analysis is that, we consider the sequence of contexts is unpredictable external information. Because we assume the scheme that each worker adopts to assign labels as a black box, and then the labeled status of all tasks could abruptly change over time. From another point of view, the context and loss at each round could also be considered to be dependent on the history of worker selection. However, reasonably modeling this dependence relation is very challenging in the setting of heterogeneous crowdsourcing. The current method considered the sequence of contexts and losses as external feedback, and adopted a standard bandit formulation. If we could somehow appropriately capture the dependence relation mentioned above, it would be possible to further improve the current theoretical results. Moreover, we can also consider selecting a task-worker pair instead of treating them independently at each round. In this case, taking an action would not be selecting a worker any more, and a new formulation (e.g. Markov decision processes) rather than bandits could be further investigated.

4.2.2 Beyond BBTA

The proposed method BBTA and task selection strategies tackled the task-worker assignment problem in crowdsourcing. There are also other important and challenging problems in this area. In the following, we further introduce other promising directions.

Dynamic Procurement

Dynamic procurement (Slivkins and Vaughan, 2014) is another interesting problem in crowdsourcing. In the setting of dynamic procurement, the requester can

set dynamic prices for labels from workers, and the goal is to maximize the total number of completed tasks with the constrain on budget. This problem involves incentive mechanism design in crowdsourcing. Since the budget is limited, setting prices too high might result in inefficiently using the budget for the requester; while setting prices too low could discourage workers to complete the tasks. This is a trade-off between the efficiency of budget usage and the incentives of workers. There are some existing results for this problem (Badanidiyuru et al., 2012, 2013a,b), but it is unclear whether these results are optimal (Slivkins and Vaughan, 2014). Hence, dynamic procurement in crowdsourcing is promising and worth further investigating.

Incorporating Experts into Crowds

To further improve the performance of crowdsourcing systems, we can incorporate the supervision of the expert. The expert is usually considered to provide correct labels. Since expert labels are much more expensive, we only require a small amount of labels from the expert, and collect most of labels from low-paid workers. The usage of expert labels is for better learning workers' reliability and finally improving the overall performance of label aggregation. More expert labels could improve the performance more, but at the same time consume more budget. In such a scenario, we need to cope with the balance between the performance and the total budget. Another question is when to require labels from expert. Before collecting labels from worker? Or after? Or even adaptively switch label collection between the expert and workers? These questions make the problem of incorporating experts into crowds non-trivial. Therefore, it would be interesting and challenging to further tackle this promising problem.

Bibliography

S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(6):121–164, 2012.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002a.

P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002b.

A. Badanidiyuru, R. Kleinberg, and Y. Singer. Learning on a budget: Posted price mechanisms for online procurement. In *Proceedings of the 13th ACM Conference on Electronic Commerce (EC2012)*, pages 128–145. ACM, 2012.

A. Badanidiyuru, R. Kleinberg, and A. Slivkins. Bandits with knapsacks: Dynamic procurement for crowdsourcing. In *The 3rd Workshop on Social Computing and User Generated Content, co-located with ACM EC 2013*, 2013a.

A. Badanidiyuru, R. Kleinberg, and A. Slivkins. Bandits with knapsacks. In *Proceedings of the 2013 IEEE 54th Annual Symposium on Foundations of Computer Science (FOCS2013)*, pages 207–216. IEEE, 2013b.

L. Balzano, R. Nowak, and B. Recht. Online identification and tracking of subspaces from highly incomplete information. In *Communication, Control, and Computing (Allerton2010)*, pages 704–711. IEEE, 2010.

S. Bubeck. *Bandits Games and Clustering Foundations*. PhD thesis, Université Lille, 2010.

S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5 (1):1–122, 2012.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

X. Chen, Q. Lin, and D. Zhou. Optimistic knowledge gradient policy for optimal budget allocation in crowdsourcing. In S. Dasgupta and D. McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML2013)*, pages 64–72. Omnipress, 2013.

X. Chen, Q. Lin, and D. Zhou. Statistical decision making for optimal budget allocation in crowd labeling. *Journal of Machine Learning Research*, 16:1–46, 2015.

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 C (Applied Statistics)*, 28(1):20–28, 1979.

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*, 39(1):1–38, 1977.

J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR2009)*, pages 248–255. IEEE, 2009.

P. Donmez, J. G. Carbonell, and J. Schneider. Effciently learning the accuracy of labeling sources for selective sampling. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD2009)*, pages 259–268. ACM, 2009.

S. Ertekin, C. Rudin, and H. Hirsh. Approximating the crowd. *Data Mining and Knowledge Discovery*, 28(5-6):1189–1221, 2014.

T. Finin, W. Murnane, A. Karandikar, N. Keller, J. Martineau, and M. Dredze. Annotating named entities in twitter data with crowdsourcing. In *NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 80–88, 2010.

G. Goel, A. Nikzad, and A. Singla. Mechanism design for crowdsourcing markets with heterogeneous tasks. In *Proceedings of the Second AAAI Conference on Human Computation and Crowdsourcing (HCOMP2014)*. AAAI, 2014.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag New York, 2009.

E. Hazan. *Efficient Algorithms for Online Convex Optimization and Their Applications*. PhD thesis, Princeton University, 2006.

E. Hazan. The convex optimization approach to regret minimization. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*, pages 287–303. MIT Press, 2011.

C. Ho and J.W. Vaughan. Online task assignment in crowdsourcing markets. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI2012)*, pages 45–51, 2012.

C. Ho, S. Jabbari, and J. W. Vaughan. Adaptive task assignment for crowdsourced classification. In S. Dasgupta and D. McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML2013)*, pages 534–542. Omnipress, 2013.

J. Howe. *Crowdsourcing: Why the Power of the Crowd Is Driving the Future of Business*. Random House, 2008.

P. G. Ipeirotis. Analyzing the amazon mechanical turk marketplace. *XRDS*, 17 (2):16–21, 2010.

P. G. Ipeirotis, F. Provost, and J. Wang. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD Workshop on Human Computation (HCOMP2010)*, pages 64–67. ACM, 2010.

P. G. Ipeirotis, F. Provost, V. S. Sheng, and J. Wang. Repeated labeling using multiple noisy labelers. *Data Mining and Knowledge Discovery*, pages 1–40, March 2013.

M. I. Jordan and T. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.

L. P. Kaelbling. *Learning in Embedded Systems*. PhD thesis, Stanford University, 1990.

H. Kajino, Y. Tsuboi, and H. Kashima. A convex formulation for learning from crowds. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI2012)*, pages 73–79, 2012.

D. Karger, S. Oh, and D. Shah. Iterative learning for reliable crowdsourcing systems. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24 (NIPS2011)*, pages 1953–1961, 2011.

A. Kittur, E. H. Chi, and B. Suh. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI2008)*, pages 453–456. ACM, 2008.

J. Langford and T. Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20 (NIPS2007)*, pages 817–824, 2008.

E. Law and L. von Ahn. *Human Computation*. Morgan & Claypool Publishers, 2011.

D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR1994)*, pages 3–12. Springer-Verlag New York, Inc., 1994.

L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web (WWW2010)*, pages 661–670. ACM, 2010.

Q. Liu, J. Peng, and A. Ihler. Variational inference for crowdsourcing. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25 (NIPS2012)*, pages 701–709, 2012.

Q. Liu, A. Ihler, and M. Steyvers. Scoring workers in crowdsourcing: How many control questions are enough? In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26 (NIPS2013)*, pages 1914–1922, 2013.

K. Mo, E. Zhong, and Q. Yang. Cross-task crowdsourcing. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD2013)*, pages 677–685. ACM, 2013.

M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. MIT Press, 2012.

K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy. Learning from crowds. *Journal of Machine Learning Research*, 11:1297–1322, 2010.

A. Ritter, S. Clark, Mausam, and O. Etzioni. Named entity recognition in tweets: An experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP2011)*, pages 1524–1534, 2011.

H. Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.

T. Scheffer, C. Decomain, and S. Wrobel. Active hidden markov models for information extraction. In *Proceedings of the International Conference on Advances in Intelligent Data Analysis (CAIDA2001)*, pages 309–318, 2001.

B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

B. Settles and M. Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP2008)*, pages 1070–1079, 2008.

S. Shalev-Shwartz. *Online Learning: Theory, Algorithms, and Applications*. PhD thesis, The Hebrew University of Jerusalem, 2007.

S. Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2012.

V. S. Sheng, F. Provost, and P. G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD2008)*, pages 614–622. ACM, 2008.

A. Slivkins and J. W. Vaughan. Online decision making in crowdsourcing markets: Theoretical challenges. *SIGecom Exchanges*, 12(2):4–23, 2014.

P. Smyth, U. M. Fayyad, M. C. Burl, P. Perona, and P Baldi. Inferring ground truth from subjective labelling of venus images. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7 (NIPS1994)*, pages 1085–1092, 1995.

R. Snow, B. O. Connor, D. Jurafsky, and A. Y. Ng. Cheap and fast - but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP2008)*, pages 254–263. Association for Computational Linguistics, 2008.

A. Sorokin and D. Forsyth. Utility data annotation with amazon mechanical turk. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW2008)*, pages 1–8. IEEE, 2008.

G. Stoltz. *Incomplete Information and Internal Regret in Prediction of Individual Sequences*. PhD thesis, Université Paris-Sud, 2005.

J. Surowiecki. *The Wisdom of Crowds*. Anchor, 2005.

W. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.

P. Wais, S. Lingamneni, D. Cook, J. Fennell, B. Goldenberg, D. Lubarov, D. Marin, and H. Simons. Towards building a high-quality workforce with mechanical turk. In *NIPS Workshop on Computational Social Science and the Wisdom of Crowds*, 2010.

F. L. Wauthier and M. I. Jordan. Bayesian bias mitigation for crowdsourcing. In J. Shawe-taylor, R. S. Zemel, P. Bartlett, F. C. N. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24 (NIPS2011)*, pages 1800–1808, 2011.

P. Welinder and P. Perona. Online crowdsourcing: rating annotators and obtaining costeffective labels. In *Workshop on Advancing Computer Vision with Humans in the Loop at CVPR*, pages 25–32. IEEE, 2010.

P. Welinder, S. Branson, S. Belongie, and P. Perona. The multidimensional wisdom of crowds. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23 (NIPS2010)*, pages 2424–2432, 2010.

J. Whitehill, T. Wu, J. Bergsma, J. R. Movellan, and P. L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22 (NIPS2009)*, pages 2035–2043, 2009.

Y. Yan, R. Rosales, G. Fung, M. Schmidt, G. Hermosillo, L. Bogoni, L. Moy, and J. Dy. Modeling annotator expertise: Learning when everybody knows a bit of something. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS2010)*, pages 932–939. JMLR, 2010.

Y. Yan, R. Rosales, G. Fung, and J. Dy. Active learning from crowds. In L. Getoor and T. Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML2011)*, pages 1161–1168, 2011.

D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16 (NIPS2004)*, pages 321–328, 2004.

D. Zhou, S. Basu, Y. Mao, and J. Platt. Learning from the wisdom of crowds by minimax entropy. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25 (NIPS2012)*, pages 2204–2212, 2012.

D. Zhou, Q. Liu, J. Platt, and C. Meek. Aggregating ordinal labels from crowds by minimax conditional entropy. In E. P. Xing and T. Jebara, editors, *Proceedings of The 31st International Conference on Machine Learning (ICML2014)*, pages 262–270, 2014.