Title
Theoretical Analysis and Efficient Algorithms for Crowdsourcing

Author
Li, Hongwei

Publication Date
2015

Peer reviewed|Thesis/dissertation

Theoretical Analysis and Efficient Algorithms for Crowdsourcing

by

Hongwei Li

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Statistics

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Bin Yu, Chair
Professor Peter J. Bickel
Professor Adityanand Guntuboyina
Professor Bruno A. Olshausen

Spring 2015

Theoretical Analysis and Efficient Algorithms for Crowdsourcing

Abstract

Theoretical Analysis and Efficient Algorithms for Crowdsourcing

by

Hongwei Li

Doctor of Philosophy in Statistics

University of California, Berkeley

Professor Bin Yu, Chair

Crowdsourcing has become an effective and popular tool for human-powered computation to label large datasets. Since the workers can be unreliable, it is common in crowdsourcing to assign multiple workers to one task, and to aggregate the labels in order to obtain results of high quality. This thesis is dedicated to analyze the process of crowdsourced labeling and propose algorithms to improve the efficiency of this process.

In Chapter 1, we give an overview of crowdsourcing and its applications. Meanwhile, we address the challenges of labeling by crowdsourcing and present the organization of the thesis.

In Chapter 2, we provide finite-sample exponential bounds on the error rate (in probability and in expectation) of general aggregation rules under the Dawid-Skene crowdsourcing model. The bounds are derived for multi-class labeling, and can be used to analyze many aggregation methods, including majority voting, weighted majority voting and the oracle Maximum A Posteriori (MAP) rule. We show that the oracle MAP rule approximately optimizes the upper bound on the mean error rate of weighted majority voting in certain setting.

In Chapter 3, we propose an iterative weighted majority voting (IWMV) method that optimizes the error rate bound and approximates the oracle MAP rule. Its one step version has a provable theoretical guarantee of the error rate. The IWMV method is intuitive and computationally simple. Compared to the traditional EM approach, IWMV is more robust to the noise of estimating workers' reliabilities and model misspecification. Experimental results on both simulation and real data show that IWMV performs at least on par with the state-of-the-art methods, and it has low computation cost (around one hundred times faster than the state-of-the-art methods).

In Chapter 4, we study the problem of selecting subsets of workers from a given worker pool to maximize the accuracy under a budget constraint. One natural question is whether we should hire as many workers as the budget allows, or restrict on a small number of top-quality workers. By theoretically analyzing the error rate of a typical setting in crowdsourcing, we frame the worker selection problem into a combinatorial optimization problem and propose

an algorithm to solve it efficiently. Empirical results on both simulated and real-world datasets show that our algorithm is able to select a small number of high-quality workers, and performs as good as, sometimes even better than, the much larger crowds as the budget allows.

In practice, worker quality (for some tasks) can be associated with some explicit characteristics of the worker, such as education level, major and age; although it is also possible that in some other tasks none of the attributes matter. In Chapter 5, we propose a general crowd targeting framework that can automatically discover, for a given task, if any group of workers based on their attributes have higher quality on average, and target these groups, if they exist, for future work of the same task. This framework is complementary to traditional worker quality estimation approaches and is more budget efficient because we are able to target potentially good workers before they actually do the task. Experiments on real datasets show that the accuracy of final prediction can be improved significantly for the same budget or even less budget in some cases. Our framework is very practical and can be easily integrated in current crowdsourcing platforms.

To my wife Sealeen Fangfang Ren,
who always encourages me to pursue my dream.

To my parents,
who support me with their fully dedicated heart.

Acknowledgments

The process of pursuing a PhD is sure one of the important journeys in my life. During this long trip, so many people have given me their greatest help, support and encouragement. Here, I would like to express my sincere thanks to them.

The first and foremost person I would like to thank is my advisor, Professor Bin Yu. Throughout the entire period, Bin advised me not only on academic research but also on my personal growth. She posed interesting questions, provided useful suggestions and comments, helped me to revise papers and encouraged me to do independent research. Looking back from this point, there were so many good moments with her that I can remember as if they just happened yesterday. During my toughest time a couple of years ago, Bin spent tremendous number of hours on helping me with her best effort. When I looked for jobs, she supported me and spent her spare time to help me to make decisions. I felt very lucky to be her student and learn from her. In short, she is not only my PhD advisor, but also my life advisor.

I want to express my many thanks to Terry Speed. We started to meet and talk with each other in 2012. Ever since then, he spends time with me whenever he visits Berkeley. To me, he is a wise and gentle mentor who helps me in many aspect of my life. Similar to Bin, he cares about me, my wife and our lives in general. We always appreciate his kind support and advice.

I would like to thank Jas Sekhon. Bin introduced me to Jas in 2013 and we started to meet and discuss projects. It is very inspiring to talk to him and he also encouraged me a lot along the way. Later when I looked for jobs, Jas was very supportive and gave me much good advice.

I would also like to thank Michael I. Jordan and Howard D'abrera for writing recommendation letters for my job applications (thanks also to Bin, Terry and Jas for doing so). I taught statistical machine learning class (as the GSI of Stat 241A) with Michael in the spring semester of 2014. It was a very enjoyable teaching experience to work with him. During my study at Berkeley, I constantly participated in Michael's group meeting and learned many things from him. I taught (as GSI) twice with Howard and he is a very nice instructor to work with. Later, we became friends and talked to each other frequently in the department and he gave me lots of encouragement. I sincerely appreciate his support.

My two internship experiences (one in summer 2012 and another in summer 2013) helped me shape my dissertation research. I would like to thank my mentor Denny Dengyong Zhou (Microsoft Research, Redmond) and Bo Zhao (Microsoft Research, Silicon Valley) for their mentoring and generous support. Denny introduced crowdsourcing to me. Then I started to work on it and chose it as my dissertation research with Bin's help. Bo kindly supported me to choose whatever interesting problems to work on during my internship in MSR-SVC in 2013. My work with them contributes to the important parts of this thesis.

Meanwhile, I want to thank my thesis committee members: Peter Bickel, Aditya Guntuboyina, Bruno Olshausen and Bin Yu for their support and review for this thesis.

Life wouldn't be colorful without friends. I thank my office mate Riddhipratim Basu and Jorge Banuelos for their cheering and jokes in the office. I would like to thank Noureddine El Karoui, Yuval Benjemini, Julien Marial, Michael Klass, David Brillinger, Miles Lopes, Christine Ho, Po-Ling Loh, Tamara Broderick, Rachel Yingxiang Wang, Siqi Wu, Hanzhong Liu, Guancheng Li, Wayne Lee, Winston Lin, Jessica Jingyi Li, Adam Bloniarz, Steven Troxler, Ryan Lovett, Mary Melinn, La Shana Porlaris, Denise Yee, Luke Miratrix, Jinzhu Jia, Xiangyu Chang and Hai Zhang for their care and help. The list is long, but I only mention a few of them above.

Finally, I want to thank my family for their unconditional love and support. Thousands of words cannot express my gratitude to my great parents, Mantang Li and Xizhi Zhu. They have been the role models to me since I was a kid. Among all these years, they dedicated all their time and energy to raise the family and support me to go to school no matter how poor and how difficult the situation is. My father Mantang always teaches me to work hard and overcome whatever difficulties in life to fulfill my dream. He has been working more than 360 days per year without breaks so as to support my family over the past twenty years. My mother Xizhi is very industrious, loving and always optimistic. After I came to the US for PhD, she always cheered me up over the phone and shared with me funny jokes and exciting things happening around her. Although she didn't have chance to go to school when she was little, she is a super smart and perseverant person in my eyes. Both my father and mother are the best teachers and inspirations for me and my life. In particular, I want to thank my wife Sealeen (Fangfang) Ren. It is her who encourages me to pursue my dream and supports me without any reservation. I thank her for all her love, care and sacrifices all these years. I feel grateful and lucky to have her in my life and she is a great blessing to me.

All the people I mentioned above and all the friends I have met in life let me realize more and more of one thing: love and goodness are the two greatest pivots for this beautiful world!

Contents

List of Figures

List of Tables

Chapter 1

Overview

In modern big data era, the development of computer science and technology has advanced many areas in the direction of artificial intelligence and automatics. Examples include self-driving car, speech recognition and image understanding, to name a few. However, there are still many tasks that can be easily carried out by people but tend to be hard for computers, e.g., image annotation, visual design and video event classification. When these tasks are extensive, outsourcing them to experts or well-trained people may be too expensive. Crowdsourcing has recently emerged as a powerful alternative.

1.1 What is crowdsourcing?

Wikipedia[1] gives the following definition of Crowdsourcing:

> *"**Crowdsourcing** is the process of obtaining needed services, ideas, or content by soliciting contributions from a large group of people, and especially from an online community, rather than from traditional employees or suppliers."*

From the definition above, the core components of crowdsourcing are the usage of open call format and a large network of potential laborers. In fact, Wikipedia itself is a product of crowdsourcing: Instead of creating an encyclopedia by hiring writers and editors, Wikipedia gives the people on the Internet the ability to create the information on their own with moderators. This leads to the most comprehensive encyclopedia that the world has ever seen.

The term "crowdsourcing" is coined by Jeff Howe in his Wired magazine article "The Rise of Crowdsourcing" (Howe, 2006). He defined crowdsourcing as "an idea of outsourcing a task that is traditionally performed by an employee to a large group of people in the form of an open call". Howe's definition pins down the essence of crowdsourcing to be distributive and related to "open call". For convenience of discussion, the entity that distributes the

[1] https://en.wikipedia.org/wiki/Crowdsourcing

tasks is referred to as *task owner* or *requester*. People who provide the answers or finish the tasks are referred to *crowds* or *crowd workers*.

The traditional outsourcing strategy that hires professionals or well-trained people is reliable but expensive. When the scale of the task is huge, *e.g.*, labeling a large volume of images, outsourcing might be unaffordable and not scalable. On the contrary, crowdsourcing can provide a scalable workforce in a budget-efficient way. This is why crowdsourcing has recently been evolving as a distributed problem solving solution in general and as a business production model (Yuen et al., 2011).

A normal process of crowdsourcing is as follows: after crowdsourcing the tasks, the task owner collects the submitted answers from the crowds. Usually, the crowd workers are compensated monetarily, with prizes or recognition. In some cases, they are just rewarded with kudos and self-satisfaction.

The downside of crowdsourcing is that the performance of all individual crowd workers can be unreliable. To overcome this, task owner usually hires multiple workers to work on each task. By using certain aggregation rule, the noisy inputs can be integrated into a result of high quality. The aggregation of crowdsourced results is the strategy of using the wisdom of crowd to solve problems.

A famous historical example of exploring the wisdom of crowd dates back to Sir Francis Galton, a British statistician. In the summer of 1906, Galton was at a livestock fair and proposed a weight-judging competition. An ox was selected and placed on display, and the people in the crowd were invited to guess the weight of the ox after it was slaughtered and dressed. There were 787 people in total who participated in this competition. After collecting all the guesses, Galton computed the median of the data. Compare to the true weight of the ox, which was 1,198 pounds, the median was 1,207 pounds (Galton, 1907b), which was within 0.8% of the true weight. Soon afterwards, Galton acknowledged that the mean was 1,197 pounds, which was even more accurate (Galton, 1907a). This vivid example shows a simple but powerful fact: under certain circumstances, crowds are outstandingly intelligent when their opinions are appropriately aggregated (Surowiecki, 2005).

With the rise of crowdsourcing, there are more and more examples of it emerge and demonstrate its power. In the next section, we will present some selected examples to show how crowdsourcing is used in our society to advance the development of science and technology.

1.2 Examples of crowdsourcing

The most famous crowdsourcing platform is Amazon Mechanical Turk (or briefly, MTurk)[2]. It enables requesters to use human intelligence to perform tasks that are challenging for computers. The requesters can first decompose their demands into microtasks called *Human Intelligence Tasks* (HITs), and then post these HITs to the online marketplace on MTurk.

[2] http://www.mturk.com

For example, some typical microtasks are identifying objects in images or videos, performing data de-duplication, transcribing recording or researching data details. The large and global population of crowd workers will finish these microtasks and get rewards, which is usually around several cents per task. MTurk enables individuals and businesses to achieve their goals in a time-efficient and budget-efficient manner. Ever since its distribution on November 2005, MTurk kept growing. In January 2011, there were reportedly over 500,000 workers from 190 countries[3].

MTurk is a typical win-win business model. From the perspective of requesters, they can get their tasks done in a relatively short time and save money. From the perspective of crowd workers, they can earn money with flexible work time and location. MTurk provides many job opportunities to hundreds of thousands of people around the world. At the same time, Amazon takes a small proportion of the rewards paid by requesters to maintain its crowdsourcing platform. The three parties – requesters, crowd workers and Amazon – combine together to form an ecosystem of crowdsourcing marketplace.

Of course MTurk is not the only crowdsourcing marketplace in the world, there are many other companies that focus on crowdsourcing business as well, such as clickworker[4], CrowdFlower[5] and Taskcn[6]. Interested readers can check a list of crowdsourcing sites in the directory of crowdsourcing sites at http://www.crowdsourcing.org/directory.

In addition to the crowdsourcing marketplaces and platforms mentioned above, there is another form of crowdsourcing system – *information sharing system*, which collects information from the crowd and share the information among the crowd. A typical example is Wikipedia[7] that mentioned early. It is a free-access and free content Internet-based encyclopedia, launched in 2001. Anyone who access the site can edit any of its articles as long as they follow the rules of Wikipedia. According to Alexa website ranking, Wikipedia is the seventh-most popular website and the most general reference work on the Internet[8]. People around the world contributed over 4.7 million articles as of 2014. Without any monetary rewards, these contributors participated for intellectual satisfaction and pure altruistic purpose. Five years after Wikipedia's launch, a similar knowledge sharing system in Chinese – *Baidu Baike*[9] was launched. As of January 2015, it had over 5 million registered users who had contributed over 10 million Chinese articles. These encyclopedia crowdsourcing-based systems benefit people all over the world every day.

There are also many other information sharing systems launched on the Internet such

[3] According to http://en.wikipedia.org/wiki/Amazon_Mechanical_Turk
[4] http://www.clickworker.com/
[5] http://www.crowdflower.com/
[6] http://www.taskcn.com/
[7] http://www.wikipedia.org/
[8] http://www.alexa.com/siteinfo/wikipedia.org
[9] http://baike.baidu.com

as the Q&A sites *Yahoo! Answers*[10], *Baidu Zhidao* [11], *Quora*[12] and *Zhihu*[13] among many others. Registered users can post any question they want to ask on these platforms, and other people who know the answers can provide solutions. Since some answers are noisy, most of such systems adopt the user voting strategy to rank the quality of answers, and show the answers in the order of their voted scores. Other people who are interested in the same questions can browse the answers on the Internet. Both the requesters who post questions and the answer providers are not rewarded with money but with recognition or virtual scores.

As an interesting paradigm of crowdsourcing, the *Game with a purpose*, pioneered by Von Ahn (2006), utilizes people's desire to be entertained to solve problems efficiently by online game players. For example, the ESP game (Von Ahn and Dabbish, 2004) is an online image labeling game that collects labels for images on web. Another similar game created by Russell et al. (2008) is named *LabelMe*[14]. Users of LabelMe play the game of labeling objects in the images to gain scores and compete for fun. These game-based tools produced labeled image dataset that are important to research in computer vision community, and they are crucial to the development of image searching systems.

Even with advanced technologies, creative work is mostly done by humans and rarely by machines. Therefore, some business models harness the power of crowdsourcing to perform creative work to reduce production costs. *Threadless*[15] is a good example and it is a platform for collecting graphic T-shirt designs submitted by users. The collected designs are then voted by the community. The designer of the top voted design will get paid with good amount of money (usually thousands of dollars). The company picks up the most popular designs, screens out copyright violations and obscenities, and then produces T-shirts with these designs out for sale. This practice not only ensures that the company has enough original designs to produce but also maintains enough attraction to customers in the market.

Another example of using crowdsourcing in a creative way is reCAPTCHA (Von Ahn et al., 2008). It is an upgraded version of CAPTCHA (Von Ahn et al., 2003), which is an acronym for "Completely Automated Public Turing test to tell Computers and Humans Apart". Essentially, reCAPTCHA is a type of challenge-response test used in computing to identify whether a user is a human or a computer program. At the same time, it solves some hard problems that challenge optical character recognition (OCR) community for a long time, such as recognizing highly distorted text images. The idea of reCAPTCHA is to present two words in distorted images to users, and asks the users to type the words to demonstrate that they are not automatic computer program. One of the two shown words is known and used as control example (identifying if this user is computer program or not), another word is supplied by the image that OCR software has been unable to recognize.

[10] http://answers.yahoo.com/
[11] http://zhidao.baidu.com
[12] http://www.quora.com
[13] http://www.zhihu.com
[14] http://labelme.csail.mit.edu
[15] https://www.threadless.com/

The results are sent to digitization projects such as Google Books. The reCAPTCHA not only contributes to identifying users but also uses the wisdom of users (crowds) in OCR for digitizing scanned books. Due to its importance in web services, it is acquired by Google Inc. in 2009.

Crowdsourcing is also applied in scientific research. An example is Foldit[16], which is an online video game on protein folding (Cooper et al., 2010). Its objective is to use the wisdom of crowd to fold the structure of selected proteins as well as possible. The solutions submitted by players are scored by the game. Then the top scored solutions are analyzed by researchers to determine whether or not there is a native structural configuration that can be applied to the relevant proteins. Scientists can use good solutions to solve real world problems and create biological innovations. In computer vision community, crowdsourcing is used to label images for creating benchmark dataset such as ImageNet[17]. ImageNet contains 14 million annotated images from Amazon Mechanical Turk, and now is used widely to train and test computer vision algorithms and systems. In natural language processing (NLP), many datasets (Snow et al., 2008) are collected by crowdsourcing to serve the NLP research community.

In the future, crowdsourcing will continue to incubate more innovations and provide power to both academic and industrial communities.

1.3 Challenges of labeling by crowdsourcing

Among many applications of crowdsourcing, labeling (or multiple choice questions) is a common and important application. Nowadays there are more and more data available on the web, but labeled data are far less than unlabeled data. It usually requires huge volume of well annotated data to train machine learning systems to build a predictive model. With crowdsourcing services such as Amazon Mechanical Turk, it becomes easy and inexpensive to obtain labeled data in a short time (Sheng et al., 2008, Snow et al., 2008, Raykar and Yu, 2012).

The success of the ImageNet dataset has confirmed that crowdsourcing can be an effective approach for labeling tasks. For illustration, image labeling tasks can be done by crowdsourcing as follows: a crowd worker is presented with an image and is asked to provide a label for this image. The label should indicate which class the presented image belongs to according to the worker's judgement, such as an image is about a cat or a dog, or a face image is about a male or a female. Therefore, the labeling problem is similar to multiple choice questions. Usually, the candidate labels form a pool of choices that the workers can select.

One major concern in crowdsourcing is quality control over crowd workers and their inputs. There are many practical questions to be addressed when the practitioners apply crowdsourcing. For example, how many workers one should hire to ensure the quality of

[16] http://fold.it/portal

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

the final results? What kind of workers one should hire to save budget and improve result quality? To study quality control in crowdsourcing, labeling problem is an ideal example to explore. The process of crowdsourced labeling is relatively straightforward to model. Since labeling is similar to classification, there are existing models that can be directly applied, such as the famous Dawid-Skene model (Dawid and Skene, 1979). Moreover, labeling is intuitive and it is easy to measure the quality of final results and performance of workers. Many efforts have been devoted to analyzing crowdsourced labeling, such as (Snow et al., 2008, Karger et al., 2011a, Raykar et al., 2010, Liu et al., 2012) and references therein.

In this thesis, we focus on both theoretical analysis and algorithms design for crowdsourced labeling. First, by deriving the error rate bounds under the general crowdsourcing models, we study how the underlying factors, such as aggregation rules, number of workers, number of tasks, worker reliabilities and task assignments, affect the quality of the final results. Second, based on the error rate bounds, we propose iterative weighted majority voting algorithm to effectively infer the ground truth labels. Third, we study the problem of selecting good workers with some exam questions. We formulate this problem into a combinatorial optimization problem and propose an efficient algorithm to obtain its global optimal solution. In the end, we also study the problem of identifying good worker group based on their characteristics and develop a crowd-targeting framework to discover good groups of workers for improving data quality and saving budget.

1.4 Organization of this thesis

The rest of the thesis is organized as follows.

In Chapter 2, we present error rate analysis of crowdsourcing models. For the multiclass labeling problem, we derive error rate bounds (both in high probability and in expectation) of the general prediction rule under the Dawid-Skene model. Then we illustrate its application to some scenarios that of common interest in the community. Simulations are conducted to demonstrate how both the error rate and its bounds are affected by different factors in the crowdsourcing system.

Based on the bounds derived in Chapter 2, we design in Chapter 3 a method called iterative weighted majority voting (IWMV) algorithm, to infer the ground truth labels effectively. Meanwhile, EM algorithms (Dempster et al., 1977) under the different crowdsourcing models are derived in this chapter. Furthermore, we provide theoretical guarantee for one-step IWMV. Experiments on both simulated and real data are conducted to compare the effectiveness of the EM algorithm, IWMV and other state-of-the-art methods.

Chapter 4 is motivated by the worker selection problem: given a set of workers and some initial estimates of their reliabilities, how we can adaptively choose a subset of workers within a budget constraint to perform the tasks to save budget and maintain high performance. We formulate this problem into a combinatorial optimization problem based on an error rate bound obtained with a similar technique as in Chapter 2. Then we propose a straightforward

solution to this optimization problem and experimentally test its effectiveness in the real crowdsourcing applications.

In Chapter 5, we develop a general framework, called Crowd Targeting, to discover and target at a good group of workers based on their worker characteristics. This is inspired by "the wisdom of minority": we hypotheses that given a task, there are certain subgroups in the crowd population who are more suitable on this task than the general population. We propose a general crowd targeting framework that can automatically discover, for a given task, if any groups of workers based on their attributes have higher quality on average, and target on these groups, if they exist, for future work of the same task. This crowd targeting framework is complementary to traditional worker quality estimation approaches. One additional advantage of our framework is that it is more budget efficient because we are able to target on potentially good workers before they actually do the task. Experiments on real datasets show that the accuracy of final prediction can be improved significantly for the same budget or even less budget in some cases.

Note that all the long proofs in each chapter are deferred to the appendix.

Chapter 2

Error rate analysis of crowdsourcing models

2.1 Introduction

As being mentioned in the previous chapter, the idea of crowdsourcing is to outsource tasks to a distributed group of people (called workers) who might be inexperienced in these tasks. The downside of it is that the input from crowds might be very noisy due to various reasons, such as low pay or lack of experience etc. However, if we can appropriately aggregate the outputs from a crowd, the yielded results could be as good as the ones by experts (Smyth et al., 1995, Snow et al., 2008, Whitehill et al., 2009, Raykar et al., 2010, Welinder et al., 2010, Yan et al., 2010, Liu et al., 2012, Zhou et al., 2012).

The flaws of crowdsourcing are apparent. Each worker is paid purely based on how many tasks that he/she has completed (for example, one cent for labeling one image). No ground truth is available to evaluate how well he/she has performed on the tasks. So some workers may randomly submit answers independent of the questions when the tasks assigned to them are beyond their expertise. Moreover, workers are usually not persistent. Some workers may complete many tasks, while the others may finish only very few tasks.

In spite of these drawbacks, is it still possible to get reliable answers in a crowdsourcing system? The answer is yes. In fact, majority voting (MV) has been able to generate fairly reasonable results (Snow et al., 2008). However, majority voting treats each worker's result as equal in quality. It does not distinguish a spammer from a diligent worker. Thus majority voting can be significantly improved upon (Karger et al., 2011a).

The first improvement over majority voting dates back at least to (Dawid and Skene, 1979). They assumed that each worker is associated with an unknown confusion matrix, whose rows are discrete conditional distributions of input from workers given ground truth. Each off-diagonal element represents misclassification rate from one class to the other, while the diagonal elements represent the accuracy in each class. Based on the observed labels by the workers, the maximum likelihood principle is applied to jointly estimate unobserved

true labels and worker confusion matrices. Although the likelihood function is non-convex, a local optimum can be obtained by using the Expectation-Maximization (EM) algorithm, which can be initialized by majority voting.

Dawid and Skene's model (Dawid and Skene, 1979) can be extended by assuming true labels are generated from a logistic model (Raykar et al., 2010), or putting a prior over worker confusion matrices (Liu et al., 2012), or taking the task difficulties into account (Bachrach et al., 2012). One may simplify the assumption made by (Dawid and Skene, 1979) to consider a confusion matrix with only a single parameter (Karger et al., 2011a, Liu et al., 2012), which we call the Homogenous Dawid-Skene model (Section 2.2).

Recently, significant progress has been made for inferring the true labels of the items. (Raykar et al., 2010) presented a maximum likelihood estimator (via EM algorithm) that infers worker reliabilities and true labels. (Welinder et al., 2010) endowed each item (i.e., image in their work) with features, which could represent concepts or topics, and workers have different areas of expertise of matching these topics. (Liu et al., 2012) transformed label inference in crowdsourcing into a standard inference problem in graphical models, and applied approximate variational methods. (Zhou et al., 2012) inferred the true labels by applying a minimax entropy principle to the distribution which jointly model the workers, items and labels. Some work also considers the problem of adaptively assigning the tasks to workers for budget efficiency (Ho et al., 2013, Chen et al., 2013).

All the previous work we mentioned above focused on applying or extending Dawid-Skene model, and inferring the true labels based on that. However, to understand the behavior and consequences of the crowdsourcing system, it is of great intention to investigate the error rate of various aggregation rules. To theoretically analyze specific algorithm, Karger et al. (2011a) provided asymptotic error bounds for their iterative algorithm and also majority voting. It seems difficult to generalize their results to other aggregation rules in crowdsourcing or apply to finite sample scenario. Very recently, Gao and Zhou (2014) studied the minimax convergence rate of the global maximizer of a lower bound of the marginal-likelihood function under a simplified Dawid-Skene model (i.e., one coin model in binary labeling). Their results are on clustering error rate, which is different from the ordinary error rate, i.e., proportion of mistakes in final labeling. They focused on the mathematical properties of the global optimizer of a specific function for sufficiently large number of workers and items, and not on the behavior of rules/algorithms which find the optimizer or aggregate the results.

In this chapter, we focus on providing finite sample bounds on the error rate of some general aggregation rules under crowdsourcing models of which the effectiveness on real data has been evaluated in (Dawid and Skene, 1979, Raykar et al., 2010, Liu et al., 2012, Zhou et al., 2012), and motivating efficient algorithms.

To the best of our knowledge, this is the first work which focuses on the finite sample error rate analysis on general aggregation rules under the practical Dawid-Skene model for crowdsourcing. The results we obtained can be used for analyzing error rate and sample complexity of algorithms. It is also worth mentioning that most of the previous work done only focused on binary crowdsourcing labeling, while our results are based on multi-class labeling, which naturally apply to the binary case. Meanwhile, we did not make any as-

sumptions on the number of workers and items in the crowdsourcing, thus the results can be directly applied to the setting of real crowdsourcing data.

2.2 Background and formulation

As an example of crowdsourcing, we assume that a set of workers are assigned to perform labeling tasks, such as judging whether an image of an animal is that of a cat, a dog or a sheep, or evaluating if a video event is abnormal or not.

Throughout this chapter, we assume there are M workers and N items for a labeling task with L label classes. We denote the set of workers $[M] = \{1, 2, \cdots, M\}$, the set of items $[N] = \{1, 2, \cdots, N\}$, and the set of labels $[L] = \{1, 2, \cdots, L\}$ (called *label set*). The extended label set is defined as $\overline{[L]} = [L] \cup \{0\} = \{0, 1, 2, \cdots, L\}$, where 0 represents the label is missing. However, in the case of $L = 2$, we follow the common convention by denoting label set as $\{-1, +1\}$ and extended label set as $\{0, -1, +1\}$.

In what follows, we use y_j as the true label for the j-th item, and $\hat{y}_j$ as the predicted label for the j-th item by an algorithm.[1] Let $\pi_k = \mathbb{P}(y_j = k)$ denotes the prevalence of label "k" in the true labels of the items for any $j \in [N]$ and $k \in [L]$.

The observed data matrix is denoted by $Z \in \overline{[L]}^{M \times N}$, where Z_{ij} is the label given by the i-th worker to the j-th item, and it will be 0 if the corresponding label is missing (the i-th worker did not label the jth item). We introduce the indicator matrix $T = (T_{ij})_{M \times N}$, where $T_{ij} = 1$ indicates that entry (i, j) is observed, and $T_{ij} = 0$ indicates entry (i, j) is unobserved. Note that T and Z are observed together.

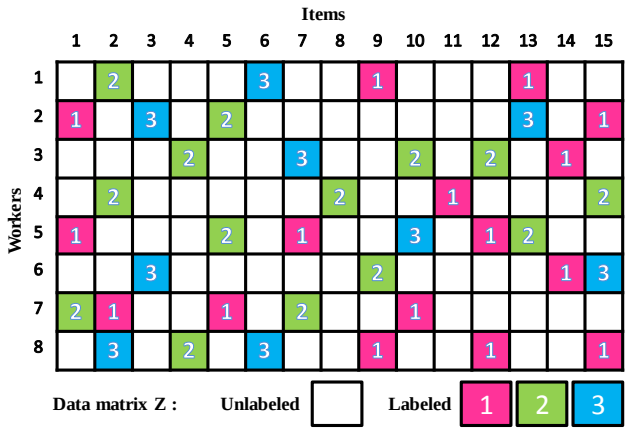


Figure 2.1: Illustration of the input data matrix. Entry (i, j) is the label of j-th item given by i-th worker. The set of labels is $[L] = \{1, 2, 3\}$.

The process of matching workers with tasks (i.e., labeling items) can be modeled by a probability matrix $Q = (q_{ij})_{M \times N}$, where $q_{ij} = \mathbb{P}(T_{ij} = 1)$ is the probability that the

[1] In this chapter, any parameter with a hat $\hat{}$ is an estimate for this parameter.

j-th item was assigned to the ith worker (i.e., gets labeled). We call Q as *assignment probability matrix*. Unlike the fixed assignment configuration in (Karger et al., 2011a), the assignment probability matrix is more flexible, and it does not require each worker label the same number of items, nor each item gets labeled by same number of workers. Hence, the assignment probability matrix covers the most general form of assigning items to workers, and there are special cases commonly adopted in literature, such as a worker has the same chance to label all items (Karger et al., 2011a, Liu et al., 2012). More specifically, when $q_{ij} = q_i \in (0,1], \forall i \in [M], j \in [N]$, we call it the *assignment probability vector* $\vec{q} = (q_1, \cdots, q_M)$. If $q_{ij} = q \in (0,1], \forall i \in [M], j \in [N]$, then we call it *constant assignment probability* q. The three assignment configurations above are referred to as the *task assignment* [2] *is based on probability matrix* Q*, probability vector* $\vec{q}$ *and constant probability* q , respectively.

Generally, we use π, p, q as probabilities, and they might have indices according to the context. We denote A and C as shorthand notations which depend on other given variables, and a as either a general constant or a vector depending on context. η denotes likelihood probabilities in the context. Θ denotes a set of parameters. ϵ and δ are constants in $(0,1)$, where ϵ is used for bounding the error rate, and δ is used for denoting a positive probability. $H_e(\epsilon)$ denotes the natural entropy of Bernoulli random variable with parameter ϵ, i.e., $H_e(\epsilon) = -\epsilon \ln \epsilon - (1-\epsilon)\ln(1-\epsilon)$. Operators $\wedge$ and $\vee$ denote the min operator and the max operator between two numbers, respectively. Meanwhile, throughout the chapter, we will locally define each notation in the context before using them.

2.2.1 Dawid-Skene models

We discuss three models covering all the cases that are widely used for modeling the quality of the workers (Dawid and Skene, 1979, Raykar et al., 2010, Karger et al., 2011a, Liu et al., 2012, Zhou et al., 2012). The first one, which is also the most general one, was originally proposed by (Dawid and Skene, 1979):

General Dawid-Skene model. In this model, the reliability of worker i is modeled as a confusion matrix $P^{(i)} = \left(p_{kl}^{(i)}\right)_{L\times L} \in [0,1]^{L\times L}$, which is in a matrix form and represents a conditional probability table such that

$$p_{kl}^{(i)} \doteq \mathbb{P}\left(Z_{ij} = l | y_j = k, T_{ij} = 1\right), \quad \forall k, l \in [L], \forall i \in [M]. \tag{2.1}$$

[2] The term task assignment seems to imply that workers are passive of labeling items — they will surely label an item whenever they are assigned to. This might not match the reality in the crowdsourcing platform. When a task owner distributed tasks to the crowd, it is likely that most of the workers will label a set of items and they can stop whenever they want to (Snow et al., 2008), unless they are required (by the owner) to complete a specific set of tasks for getting paid. Thus the process of matching tasks with workers might be determined by either workers (subjectively) or task owners (by enforcement), which depends on how the owners design and distribute tasks. If the workers have choice to select which item to label, it might be more proper to call the task-worker matching as *task selection*, instead of *task assignment*. However, both of the cases can be modeled by the probability matrix Q (or probability vector $\vec{q}$, or constant probability q). In what follows, we use the term *task assignment* to represent the task-worker matching without introducing ambiguity.

Note that $p_{kk}^{(i)}$ denotes the accuracy of worker i on labeling an item with true label k correctly, and $p_{kl}^{(i)}, k \neq l$ represents the error probability of labeling an item with true label k as l mistakenly. The number of free parameters of modeling the reliability of a worker are $L(L-1)$ under the General Dawid-Skene model.

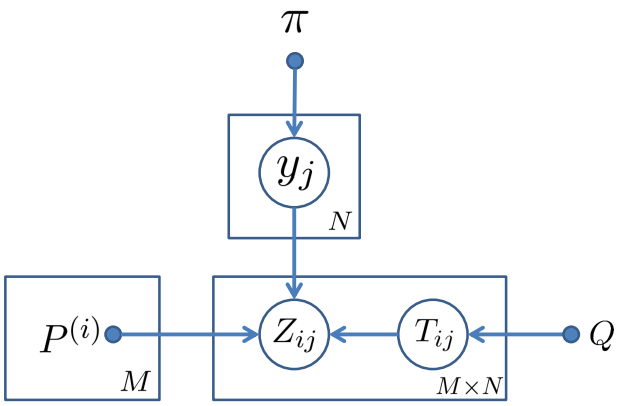


Figure 2.2: The graphical model of the General Dawid-Skene model. Note that $P^{(i)}$ is the confusion matrix of worker i and if we change $P^{(i)}$ to w_i, the graph will become the graphical model for the Homogenous Dawid-Skene model.

Since the General Dawid-Skene model has $L(L-1)$ degree of freedom to model each worker, it is flexible, but often leads to overfitting on small datasets. As a further regularization of the worker models, we consider another two models which are special cases of General Dawid-Skene model via imposing constraints on the worker confusion matrices.

- *Class-Conditional Dawid-Skene model.* In this model, the error probabilities of labeling an item with true label k as label l mistakenly for each worker are the same across different l. Formally, we have

$$\begin{cases} p_{kk}^{(i)} = \mathbb{P}\left(Z_{ij} = k | y_j = k, T_{ij} = 1\right), & \forall k \in [L], \forall i \in [M], \\ p_{kl}^{(i)} = \frac{1-p_{kk}^{(i)}}{L-1}, & \forall k, l \in [L], l \neq k, \forall i \in [M]. \end{cases} \tag{2.2}$$

  This model simplifies the error probabilities of a worker to be the same given the true label of an item. Thus, the off-diagonal elements of each row of the confusion matrix $P^{(i)}$ will be the same. The number of free parameters to model the reliability of each worker under the Class-Conditional Dawid-Skene model is L.

- *Homogenous Dawid-Skene model.* Each worker is assumed to have the same accuracy on each class of items, and have the same error probabilities as well. Formally, worker i labels an item correctly with a fixed probability w_i and mistakenly with another fixed

probability $\frac{1-w_i}{L-1}$, i.e.,

$$\begin{cases} p_{kk}^{(i)} = w_i, & \forall k \in [L], \forall i \in [M], \\ p_{kl}^{(i)} = \frac{1-w_i}{L-1}, & \forall k, l \in [L], k \neq l, \forall i \in [M]. \end{cases} \tag{2.3}$$

In this case, the worker labels an item with the same accuracy, independent of which label this item actually is. The number of parameters of modeling the reliability of each worker is 1 under the Homogenous Dawid-Skene model.

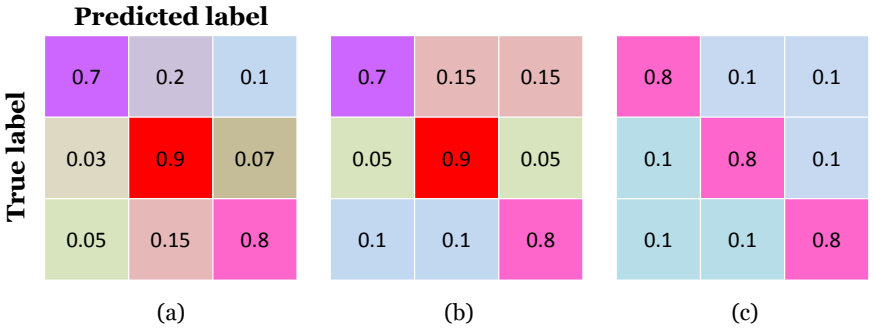


Figure 2.3: Toy examples of confusion matrices of different worker reliability models. We assume $L = 3$ thus confusion matrices $\in [0, 1]^{3\times3}$. (a) General Dawid-Skene model. (b) Class-Conditional Dawid-Skene model. (c) Homogenous Dawid-Skene model. Vertical axis is actual classes, and horizontal axis is predicted classes. Different color corresponds to different conditional probability, and the diagonal elements are the accuracy of labeling the corresponding class of items correctly.

The parameter set under all the three models can be denoted as $\Theta = \left\{ \left\{ P^{(i)} \right\}_{i=1}^{M}, Q, \pi \right\}$. Specifically, the parameter set of the Homogenous Dawid-Skene model can be denoted as $\Theta = \left\{ \{w_i\}_{i=1}^{M}, Q, \pi \right\}$.

It is worth mentioning that when $L = 2$, the Class-Conditional Dawid-Skene model and the General Dawid-Skene model are the same, which are referred to as the two-coin model, and the Homogenous Dawid-Skene model is referred to as the one-coin model in the literature (Raykar et al., 2010, Liu et al., 2012). In signal processing, the Homogenous Dawid-Skene model is equivalent to the random classification noise model (Angluin and Laird, 1988), and the Class-Conditional Dawid-Skene model is also referred to as the class-conditional noise model (Natarajan et al., 2013). We do not adopt these terms because the error comes from the limitations of workers' ability to label items correctly, not from noise which is in the context of signal processing.

Binary labeling is the special case ($L = 2$), and it is the major focus of the previous research in crowdsourcing (Raykar et al., 2010, Karger et al., 2011a, Liu et al., 2012). As

a convention, we assume the set of labels is $\{\pm 1\}$ instead of $\{1, 2\}$ when $L = 2$. For notation convenience, we define the worker confusion matrix in a different way as follows: for $i = 1, 2, \cdots, M$

$$\begin{cases} p_+^{(i)} = \mathbb{P}(Z_{ij} = 1 | y_j = 1, T_{ij} = 1), \\ p_-^{(i)} = \mathbb{P}(Z_{ij} = -1 | y_j = -1, T_{ij} = 1). \end{cases} \tag{2.4}$$

Then the parameter set will be $\Theta = \left\{ \left\{ p_+^{(i)}, p_-^{(i)} \right\}_{i=1}^{M}, Q, \pi \right\}$ under this model. When we present the result of binary labeling, we will use $p_+^{(i)}$ and $p_-^{(i)}$ without introducing any ambiguity.

Under the models above, the posterior probability of the true label of item j to be k is defined as:

$$\rho_k^{(j)} = \mathbb{P}(y_j = k | Z, T, \Theta), \quad \forall j \in [N]. \tag{2.5}$$

For binary labeling, the posterior probability of the true label of item j to be k is defined as:

$$\rho_+^{(j)} = \mathbb{P}(y_j = 1 | Z, T, \Theta), \quad \forall j \in [N]. \tag{2.6}$$

2.2.2 Aggregation rules

After collecting the crowdsourced labels, the task owner could use an arbitrary rule to aggregate the multiple noisy labels of an item to a "refined" label for that item. The quality of final predicted labels depends not only on the input from the workers but also on the aggregation rule. It is hence of great importance to design a good aggregation rule.

A natural aggregation rule is *majority voting* (Snow et al., 2008, Karger et al., 2011a). For multiple labeling, majority voting can be written formally as

$$\hat{y}_j = \underset{k \in [L]}{\operatorname{argmax}} \sum_{i=1}^{M} \mathrm{I}\left(Z_{ij} = k\right), \tag{2.7}$$

which gives the jth item the majority label among all workers.

Since workers have different reliabilities, it is inefficient to treat the labels from different workers with the same weight as in majority voting (Karger et al., 2011a). A natural extension of majority voting is *weighted majority voting* (WMV), which weighs the labels differently. Formally, weighted majority voting rule can be written as

$$\hat{y}_j = \underset{k \in [L]}{\operatorname{argmax}} \sum_{i=1}^{M} \nu_i \mathrm{I}\left(Z_{ij} = k\right), \tag{2.8}$$

where $\nu_i \in \mathbb{R}$ is the weight associated with the ith worker.

The idea behind weighted majority voting can be generalized in this way: given an item and the labels input from workers, the item can be potentially predicted to be any label class in $[L]$. Suppose each class has a score which is computed based on the worker inputs, and we call it the *aggregated score* for potential class [3] , then the aggregated label can be chosen as the class that obtains the highest score.

Based on the ideas above, we consider a general form of aggregation rule which is based on maximizing aggregated scores. The aggregated score of each label class can be decomposed into the sum of bounded prediction functions associated with workers, plus a shift constant. We refer this type of aggregation rule to *decomposable aggregation rule*, which has the form

$$\hat{y}_j = \underset{k\in[L]}{\operatorname{argmax}}\, s_k^{(j)} \quad \text{and} \quad s_k^{(j)} \doteq \sum_{i=1}^{M} f_i(k, Z_{ij}) \;+\; a_k, \tag{2.9}$$

where $s_k^{(j)}$ is the *aggregated score for potential class k on item j*, $a_k \in \mathbb{R}$ and $f_i : [L] \times \overline{[L]} \to \mathbb{R}$ $\forall i \in M$ is finite, i.e., $|f_i(k,h)| < \infty$, $\forall k \in [L], h \in \overline{[L]}$. Given k and h, $f_i(k,h)$ is a constant. Intuitively, $f_i(k,h)$ is the score gained for the kth potential label class when the ith worker labels an item with label h. It is reasonable to assume that $Z_{ij} = 0$ contributes no information to predict the true label of jth item, thus, we further assume that $f_i(k,0) = \text{constant}$, $\forall i \in [M], k \in [L]$. Without ambiguity, we will refer to $\{f_1, f_2, \cdots, f_M\}$ as *score functions* in what follows. Note that score functions are usually designed by the task owners when they aggregate the noisy inputs into final predicted results.

For illustration, majority voting is a special case of decomposable aggregation rule with $a_k = 0$, $f_i(k, Z_{ij}) = 1$ if $Z_{ij} = k$, and 0 if $Z_{ij} \neq k$. For weighted majority voting, $f_i(k, Z_{ij}) = \nu_i$ if $Z_{ij} = k$, and 0 otherwise. Later, we will see more aggregation rules which can be expressed or approximated in this form (Section 2.4.2).

2.2.3 Performance metric

Given an estimation or an aggregation rule, suppose that its predicted label for item j is $\hat{y}_j$, then our objective is to minimize the error rate

$$\text{ER} \;=\; \frac{1}{N}\sum_{j=1}^{N} \mathrm{I}\left(\hat{y}_j \neq y_j\right). \tag{2.10}$$

Since the error rate is random, we are also interested in its expected value (i.e., the *mean error rate*). Formally, the mean error rate is:

$$\mathbb{E}[\text{ER}] \;=\; \frac{1}{N}\sum_{j=1}^{N} \mathbb{P}(\hat{y}_j \neq y_j). \tag{2.11}$$

[3] Since we do not know the true label of an item, we use the term *potential class* based on the fact that all the label class can potentially be the true label of this item.

The rest of the chapter is organized as follows. In Section 2.3, we present finite-sample bounds on the error rate of the decomposable aggregation rule both in probability and in expectation under the General Dawid-Skene model. Section 2.4 contains the error rate bounds of some special cases, which the crowdsourcing community is widely concerned with. Experimental results on simulated dataset are presented in section 2.5. Note that the proofs are deferred to the supplementary materials.

2.3 Error rate bounds

In this section, finite sample bounds are provided for error rates (in high probability and in expectation) of the decomposable aggregation rule under the Dawid-Skene model.

Our main results will be focused on the setting which is as general as possible. We define the *general setting* as follows:

- *Worker modeling.* We focus on the General Dawid-Skene model, and then the results can be specialized for the Class-Conditional Dawid-Skene model and the Homogenous Dawid-Skene model straightforwardly.
- *Task assignment.* We consider the data matrix that is observed based on assignment probability matrix $Q = (q_{ij})_{M\times N}$,where $q_{ij} = \mathbb{P}(T_{ij} = 1)$. The results can be easily simplified to the scenarios where task assignment is based on probability vector $\vec{q}$ or constant probability q according to the practical assignment process.
- *Aggregation rule.* Our main results will be presented based on the decomposable aggregation rule (2.9).

2.3.1 Some quantities of interest

One important question we want to address is that how the error rate is bounded with high probability, and what quantities have impact on the bounds. Before deriving the error rate bound, we introduce some quantities of interest under the *general setting* in this section. We shall bear in mind that all the quantities defined here serve the purpose of defining two measures t_1 and t_2, which play a central role in bounding the error rate.

The first quantity Γ is associated with the score functions $\{f_1, f_2, \cdots, f_M\}$:

$$\Gamma \doteq \sqrt{\sum_{i=1}^{M}\left(\max_{k,l,h\in[L],k\neq l}|f_i(k,h)-f_i(l,h)|\right)^2}. \tag{2.12}$$

It measures the overall variation of the f_i's on their first argument (i.e., when the potential label class changes). Take weigthed majority voting (2.8) as an example, $f_i(k,h) = \nu_i \mathrm{I}(h = k)$, then $\Gamma = \sqrt{\sum_{i=1}^{M}\nu_i^2} = ||\nu||_2$. For majority voting, $\Gamma = \sqrt{\sum_{i=1}^{M}1} = \sqrt{M}$. Note that Γ is

invariant to a translation of score functions, and is linear to scale of score functions. That is to say, if we design new score functions as $f_i' = mf_i + b$ for constant m and b, and for all $i \in [M]$, then the corresponding quantity $\Gamma' = m\Gamma$. Later on, we will see that it plays a role in normalization.

Another quantity $\Lambda_{kl}^{(j)}$ is defined as the *expected gap of the aggregated scores* between two potential label classes k and l, when the true label is k. Formally,

$$\Lambda_{kl}^{(j)} \doteq \mathbb{E}\left[s_k^{(j)} - s_l^{(j)} | y_j = k\right] = \sum_{i=1}^{M}\sum_{h=1}^{L} q_{ij}\left(f_i(k,h) - f_i(l,h)\right) p_{kh}^{(i)} \ + \ (a_k - a_l)\,. \tag{2.13}$$

The larger this quantity is, the easier the aggregation rule identifies the true label as k instead of l (i.e., correctly predicted the label). Like Γ, $\Lambda_{kl}^{(j)}$ is also invariant to translation of score functions, and linear to scale of score functions. Take weighted majority voting (2.8) for illustration, the gap of aggregated scores under the Homogenous Dawid-Skene model is $\Lambda_{kl}^{(j)} = \sum_{i=1}^{M}\sum_{h=1}^{L}\left(q_{ij}\nu_i w_i - q_{ij}\nu_i\frac{1-w_i}{L-1}\right) = \frac{1}{L-1}\sum_{i=1}^{M} q_{ij}\nu_i(Lw_i - 1)$, because $f_i(k,h) = \nu_i \mathrm{I}\left(h = k\right)$ and $p_{kh}^{(i)} = w_i$ if $h = k$, otherwise $\frac{1-w_i}{L-1}$.

The following two quantities serve as the lower bound and the upper bound of the normalized gap of aggregated scores for the jth item (i.e., the ratio of $\Lambda_{kl}^{(j)}$ and Γ).

$$\tau_{j,\min} \doteq \min_{k,l\in[L],k\neq l} \frac{\Lambda_{kl}^{(j)}}{\Gamma} \qquad \text{and} \qquad \tau_{j,\max} \doteq \max_{k,l\in[L],k\neq l} \frac{\Lambda_{kl}^{(j)}}{\Gamma}, \tag{2.14}$$

Both $\tau_{j,\min}$ and $\tau_{j,\max}$ are invariant to translation and scale of the score functions.

Now, we introduce the two most important quantities of interest — t_1 and t_2, which are respectively the lower bound and the upper bound of the normalized gap of aggregated scores across all items. In our main results, t_1 is used to provide a sufficient condition for an upper bound on the error rate of crowdsourced labeling under the general setting. Meanwhile, t_2 is used to provide a sufficient condition for a lower bound of the error rate.

$$t_1 \doteq \min_{j\in[N]} \tau_{j,\min} \qquad \text{and} \qquad t_2 \doteq \max_{j\in[N]} \tau_{j,\max}, \tag{2.15}$$

Both t_1 and t_2 are invariant to translation and scale of the score functions $\{f_1, f_2, \cdots, f_M\}$. t_1 and t_2 are related to how good a group of workers are, how the tasks are assigned and how well the aggregation rule is with respect to this type of labeling tasks.

Besides the quantities of interest above, we further introduce two notations which can capture the fluctuation of the score functions and the gap of aggregated scores. These two quantities will be used to bound the mean error rate of crowdsourced labeling.

A quantity c measures the maximum change amongst all the score functions when the potential label class changes from one label to another, and it is defined as

$$c \ = \ \frac{1}{\Gamma} \cdot \max_{i\in[M],k,l,h\in[L],k\neq l} |f_i(k,h) - f_i(l,h)|. \tag{2.16}$$

For weighted majority voting (2.8), $c = \frac{\max_{i\in[M]}|\nu_i|}{||\nu||_2} = \frac{||\nu||_\infty}{||\nu||_2}$. And for majority voting (2.7), $c = \frac{1}{\sqrt{M}}$. Another quantity σ^2 relate to the variation of the gap of aggregated scores, and

$$\sigma^2 \quad = \quad \frac{1}{\Gamma^2} \cdot \max_{j\in[N],k,l\in[L],k\neq l} \sum_{i=1}^{M}\sum_{h=1}^{L} q_{ij}\left(f_i(k,h) - f_i(l,h)\right)^2 p_{kh}^{(i)} \quad . \tag{2.17}$$

Note that $0 < c \leq 1$, and $0 < \sigma^2 \leq \max_{i\in[M],j\in[N]} q_{ij}$. Both c and σ^2 are translation and scale invariant of the score functions $\{f_1, f_2, \cdots, f_M\}$.

In the next section, with t_1 and t_2, we will derive the bounds on the error rate of crowdsourced labeling with high probability, and together with c and σ^2, we will derive the bounds on the mean error rate under the general setting.

2.3.2 Main results

In this section, we start with a main theorem to provide finite-sample error rate bounds for decomposable aggregation rules under the General Dawid-Skene model (2.1).

To lighten the notation, we define two functions as follows:

$$\phi(x) = e^{-\frac{x^2}{2}} \qquad x \in \mathbb{R}, \tag{2.18}$$

$$\mathrm{D}(x||y) = x\ln\frac{x}{y} + (1-x)\ln\frac{1-x}{1-y} \qquad \forall x, y \in (0,1). \tag{2.19}$$

$\phi(\cdot)$ is the unnormalized standard Gaussian density function. $\mathrm{D}(x||y)$ is the Kullback-Leibler divergence of two Bernoulli distributions with parameters x and y respectively.

The following theorem provides sufficient conditions and the corresponding high probability bounds on the error rate under the *general setting* as described in Section 2.3.

Theorem 2.1. ***(Bounding error rate with high probability)*** *Under the General Dawid-Skene model as in (2.1), with the prediction function for each item as in (2.9), and suppose the task assignment is based on probability matrix $Q = (q_{ij})_{M\times N}$ where q_{ij} is the probability that the worker i labels item j.*

For $\forall\epsilon \in (0,1)$, with notations defined from (2.12) to (2.19), we have:

(1) If $t_1 \geq \sqrt{2\ln\frac{L-1}{\epsilon}}$, *then*

$$\frac{1}{N}\sum_{j=1}^{N} I(\hat{y}_j \neq y_j) \leq \epsilon \text{ with probability at least } 1 - e^{-ND(\epsilon||(L-1)\phi(t_1))}. \tag{2.20}$$

(2) If $t_2 \leq -\sqrt{2\ln\frac{1}{1-\epsilon}}$, *then*

$$\frac{1}{N}\sum_{j=1}^{N} I(\hat{y}_j \neq y_j) \geq \epsilon \text{ with probability at least } 1 - e^{-ND(\epsilon||1-\phi(t_2))}. \tag{2.21}$$

Remark: The high probability bounds on error rate require conditions on t_1 and t_2, which are related to the normalized gap of aggregated scores (Section 2.3.1). Basically, if the scores of predicting an item as its true label (predicted correctly) are larger than the scores of predicting as a wrong label, then it is more likely that the error rate will be small, thus it is bounded from above with high probability. The interpretation of the lower bound and its condition is similar to the upper bound.

To ensure the probability of bounding the error rate to be at least $1-\delta$, we have to solve the equation $D(\epsilon||(L-1)\phi(t_1)) = \frac{1}{N}\ln\frac{1}{\delta}$, which cannot be solved analytically. Thus, we need to figure out the minimum t_1 for bounding the error rate with probability at least $1-\delta$. The following theorem serves this purpose by slightly relaxing the conditions on t_1 and t_2 in Theorem 2.1.

Before presenting the next theorem, we define a notation C , which depends on parameters $\epsilon, \delta \in (0,1)$, for lightening notations in the theorem.

$$C(\epsilon,\delta) \quad = \quad 1+\exp\left(\frac{1}{\epsilon}\left[H_e(\epsilon)+\frac{1}{N}\ln\frac{1}{\delta}\right]\right), \tag{2.22}$$

where $H_e(\epsilon) = -\epsilon\ln\epsilon-(1-\epsilon)\ln(1-\epsilon)$, which is the natural entropy of a Bernoulli random variable with parameter ϵ.

Theorem 2.2. *With the same notation as in Theorem 2.1, for $\forall\epsilon,\delta\in(0,1)$, we have:*
(1) if $t_1 \geq \sqrt{2\ln[(L-1)C(\epsilon,\delta)]}$, then $\frac{1}{N}\sum_{j=1}^{N} I(\hat{y}_j \neq y_j) \leq \epsilon$ with probability at least $1-\delta$,
(2) if $t_2 \leq -\sqrt{2\ln G}$, then $\frac{1}{N}\sum_{j=1}^{N} I(\hat{y}_j \neq y_j) \geq \epsilon$ with probability at least $1-\delta$.

Remark: For any scenarios that can be formulated into the *general setting* as in 2.3, with t_1 and t_2 computed as in Section 2.3.1, both Theorem 2.1 and Theorem 2.2 can be applied. Therefore, in the rest of the chapter, for any special case (Section 2.4) of the general setting , we will only present one of Theorem 2.1 and Theorem 2.2, and omit the other for clarity.

In practice, the mean error rate might be a better measure of performance because of its non-random nature. The method of evaluating the accuracy of a certain algorithm is often conducted by taking an empirical average of its performance in each trial, which is a consistent estimator of the mean error rate. Thus it will be of general interest to bound the mean error rate.

For the next theorem, we present the mean error rate bound under the general setting used in Theorem 2.1.

Theorem 2.3. ***(Bounding the mean error rate.)*** *Under the same setting as in Theorem 2.1, with c and σ^2 defined as in (2.16) and (2.17) respectively,*
(1) if $t_1 \geq 0$, then $\frac{1}{N}\sum_{j=1}^{N}\mathbb{P}(\hat{y}_j \neq y_j) \leq (L-1)\cdot\min\left\{\exp\left(-\frac{t_1^2}{2}\right),\ \exp\left(-\frac{t_1^2}{2(\sigma^2+ct_1/3)}\right)\right\}$,
(2) if $t_2 \leq 0$, then $\frac{1}{N}\sum_{j=1}^{N}\mathbb{P}(\hat{y}_j \neq y_j) \geq 1-\min\left\{\exp\left(-\frac{t_2^2}{2}\right),\ \exp\left(-\frac{t_2^2}{2(\sigma^2-ct_2/3)}\right)\right\}$.

Remark: The results above are composed of two exponential bounds, and neither is generally dominant over the other. Thus each component inside the min operator can be served as an individual bound for the mean error rate. Take the upper bound in (1) as an example, when t_1 is small (recall that both σ^2 and c are bounded above by 1), the second component will be tighter than the first one. Thus the error rate bound behaves like e^{-t_1}. Otherwise, the first component will be tighter, and the mean error rate behaves like $e^{-t_1^2}$.

All the proofs of the results in this section are deferred to Appendix A. In the following section, we demonstrate how the main results here can be applied to various settings of labeling by crowdsourcing, and provide theoretical bounds for the error rate correspondingly.

2.4 Apply error rate bounds to some typical scenarios

In this section, we apply the general results in Section 2.3 to some common settings such as binary labeling, and aggregation rules such as majority voting and weighted majority voting. Specifically, we consider the following scenarios:

- Worker modeling: Recall that the General Dawid-Skene model is the same as the Class-Conditional Dawid-Skene model when $L = 2$. We will cover the binary case with a certain aggregation rule under the General Dawid-Skene model. For $L > 2$, we focus on multiclass labeling under the Homogenous Dawid-Skene model.
- Task assignment: We consider the scenario that the task assignment is based on probability vector $\vec{q} = (q_1, q_2, \cdots, q_M)$ or a constant probability q.
- Aggregation rule: We focus on weighted majority voting (WMV) rule and majority voting (MV) since they are intuitive and of common interest. In the case of binary labeling, we consider a general hyperplane rule. Later, we present results for Maximum A Posteriori rule.

2.4.1 Error rate bounds for hyperplane rule, WMV and MV

It turns out that in practice, many prediction methods for binary labeling ($y_j \in \{\pm 1\}$) can be formulated as a sign function of a hyperplane, such as majority voting, weighted majority voting and the oracle MAP rule (Section 2.4.2). In this section, we are going to apply our results in Section 2.3 to discuss the error rate of the aggregation rule, whose decision boundary is a hyperplane in a high dimensional space that the label vector of each item (i.e., all the labels from the workers for this item) lies in. Formally, the aggregation rule is

$$\hat{y}_j = \text{sign}\left(\sum_{i=1}^{M} \nu_i Z_{ij} + a\right) \tag{2.23}$$

This is called the *general hyperplane rule* with unnormalized weights $\nu = (\nu_1, \cdots, \nu_M)$ and shift constant a. For binary labeling, majority voting is a special case with $\nu_i = 1, \forall i \in [M]$ and $a = 0$.

Theorem 2.1, Theorem 2.2 and Theorem 2.3 can be directly applied to derive the error rate bounds of general hyperplane rule in the following corollary.

When task assignment is based on probability vector $\vec{q}$, the corresponding quantities of interest for the hyperplane rule (as in Section 2.3.1) are defined as follows:

$$t_1 = \frac{1}{||\nu||_2}\left[\left(\sum_{i=1}^{M} q_i\nu_i(2p_+^{(i)} - 1) + a\right) \wedge \left(\sum_{i=1}^{M} q_i\nu_i(2p_-^{(i)} - 1) - a\right)\right] \tag{2.24}$$

$$t_2 = \frac{1}{||\nu||_2}\left[\left(\sum_{i=1}^{M} q_i\nu_i(2p_+^{(i)} - 1) + a\right) \vee \left(\sum_{i=1}^{M} q_i\nu_i(2p_-^{(i)} - 1) - a\right)\right] \tag{2.25}$$

$$c = \frac{||\nu||_\infty}{||\nu||_2} \quad \text{and} \quad \sigma^2 = \frac{1}{||\nu||_2^2}\sum_{i=1}^{M} q_i\nu_i^2, \tag{2.26}$$

where $\wedge$ and $\vee$ are *min* and *max* operators respectively, $||\nu||_\infty \doteq \max_{i\in[M]} |\nu_i|$, and functions $\phi(x)$ and $\mathrm{D}(x||y)$ are defined in (2.18) and (2.19).

Corollary 2.4. (Hyperplane rule in binary labeling) *Consider binary labeling under the General Dawid-Skene model, i.e., $y_j \in \{\pm 1\}$, $p_+^{(i)}$ and $p_-^{(i)}$ are defined as in (2.4). Suppose the task assignment is based on probability vector $\vec{q} = (q_1, q_2, \cdots, q_M)$ where q_i is the probability that worker i labels an item, aggregation rule is general hyperplane rule as in (2.23), and the quantities of interest are defined in (2.24)-(2.26), for $\forall \epsilon \in (0, 1)$:*

(1) If $t_1 \geq 0$, *then* $\frac{1}{N}\sum_{j=1}^{N} \mathbb{P}\left(\hat{y}_j \neq y_j\right) \leq \min\left\{\exp\left(-\frac{t_1^2}{2}\right), \exp\left(-\frac{t_1^2}{2(\sigma^2 + ct_1/3)}\right)\right\}$.

Furthermore, if $t_1 \geq \sqrt{2\ln\frac{1}{\epsilon}}$, *then* $\mathbb{P}\left(\frac{1}{N}\sum_{j=1}^{N} I(\hat{y}_j \neq y_j) \leq \epsilon\right) \geq 1 - e^{-ND(\epsilon||\phi(t_1))}$.

(2) If $t_2 \leq 0$, *then* $\frac{1}{N}\sum_{j=1}^{N} \mathbb{P}\left(\hat{y}_j \neq y_j\right) \geq 1 - \min\left\{\exp\left(-\frac{t_2^2}{2}\right), \exp\left(-\frac{t_2^2}{2(\sigma^2 - ct_2/3)}\right)\right\}$.

Furthermore, if $t_2 \leq -\sqrt{2\ln\frac{1}{1-\epsilon}}$, *then* $\mathbb{P}\left(\frac{1}{N}\sum_{j=1}^{N} I(\hat{y}_j \neq y_j) \geq \epsilon\right) \geq 1 - e^{-ND(\epsilon||1-\phi(t_2))}$.

Proof. The aggregation rule can be expressed as $\hat{y}_j = \mathrm{argmax}_{k\in\{\pm1\}} \sum_{i=1}^{M} \nu_i \mathrm{I}\left(Z_{ij} = k\right) + a_k$, where $a_+ = a$ and $a_- = 0$. Directly apply Theorem 2.1 and Theorem 2.3, with $f_i(k, k) = \nu_i$ for $k \in \{\pm 1\}$, $f_i(k, l) = 0$ for $k \neq l$, and denote $p_{kl}^{(i)}$ in terms of $p_+^{(i)}$ and $p_-^{(i)}$, $q_{ij} = q_i, \forall j \in [N]$, we can obtain the desired result immediately. □

Remark: From this corollary, we know that if t_1 is big enough, then the error rate will be upper bounded by ϵ with high probability. This means when the workers' reliabilities are generally good, it is very likely that we will have high quality aggregated results. Note that usually we can freely choose q, ν and a, so the most important factors are worker reliabilities on positive samples ($p_+^{(i)}$) and negative samples ($p_-^{(i)}$).

If one needs to know, given ϵ and δ, what are the results under the setting above corresponding to Theorem 2.2, we can simply compute t_1 and t_2 as listed in Corollary 2.4, then the conditions and bounds in Theorem 2.2 hold.

As mentioned in the beginning of this section, weighted majority voting (WMV) is an important special case covered by our results in Section 2.3. For the next several results, we focus on the mean error rate bound of WMV and MV under Homogenous Dawid-Skene model.

For simplicity, we consider the case where task assignment is based on constant probability q. Therefore, in the following corollary, the label of any entry (i, j) is assumed to be revealed with constant probability q, and weighted majority voting (2.8) is applied to obtain the aggregated labels in the end.

Corollary 2.5. (Weighted majority voting under Homogenous Dawid-Skene model) *For weighted majority voting, whose prediction rule is* $\hat{y}_j = \text{argmax}_{k\in[L]} \sum_{i=1}^{M} \nu_i I(Z_{ij} = k)$ *with* $\nu = (\nu_1, \nu_2, \cdots, \nu_M)$. *Assume the task assignment is based on constant probability* $q \in (0, 1]$, *and assume the worker reliability model is the Homogenous Dawid-Skene model with parameter* $\{w_i\}_{i=1}^{M}$, *then with*

$$t = \frac{q}{(L-1)||\nu||_2} \sum_{i=1}^{M} \nu_i (Lw_i - 1), \quad c = \frac{||\nu||_\infty}{||\nu||_2} \quad \text{and} \quad \sigma^2 = q \tag{2.27}$$

(1) if $t \geq 0$, *then* $\frac{1}{N}\sum_{j=1}^{N} \mathbb{P}(\hat{y}_j \neq y_j) \leq (L-1) \min\left\{\exp\left(-\frac{t^2}{2}\right), \exp\left(-\frac{t^2}{2(\sigma^2 + ct/3)}\right)\right\}$

(2) if $t \leq 0$, *then* $\frac{1}{N}\sum_{j=1}^{N} \mathbb{P}(\hat{y}_j \neq y_j) \geq 1 - \min\left\{\exp\left(-\frac{t^2}{2}\right), \exp\left(-\frac{t^2}{2(\sigma^2 - ct/3)}\right)\right\}$

Proof. Under this setting, we have $p_{kk}^{(i)} = w_i$ and $p_{kl}^{(i)} = \frac{1-w_i}{L-1}$ when $k \neq l$. Meanwhile, $f_i(k, k) = \nu_i$ for any $k \in [L]$ and $f_i(k, l) = 0$ for $k, l \in [L], k \neq l$. Then $\Gamma = ||\nu||_2$. By plugging these above into the definitions of t_1, t_2, c and σ^2 in Section 2.3, we can then obtain the results. □

Remark: Note that $t_1 = t_2 = t$ under the setting of this corollary. By replacing t_1 and t_2 with t, the high probability bound on the error rate as in Theorem 2.1 and Theorem 2.2 will hold as well. It is worth mentioning that the measure t is of critical importance for the quality of the final aggregated results. It depends not only on the workers' reliability, but also on the weight vector ν, which can be chosen by us. This leaves room for us to study the best possible weight based on the bound we derived here. In Section 3.5.1, we will investigate the optimal weight in detail.

As a further special case of weighted majority voting and a commonly used prediction rule in crowdsourcing, the majority voting (MV) rule uses the same weight for each worker. It can be formally expressed as $\hat{y}_j = \text{argmax}_{k\in[L]} \sum_{i=1}^{M} \mathrm{I}(Z_{ij} = k)$. We can then directly obtain the error rate bounds of MV under the Homogenous Dawid-Skene model.

Corollary 2.6. (Majority voting under Homogenous Dawid-Skene model) *For majority voting with task assignment being based on a constant probability* $q \in (0,1]$, *and* $\bar{w} = \frac{1}{M}\sum_{i=1}^{M} w_i$, *if* $\bar{w} > \frac{1}{L}$, *then*

$$\frac{1}{N}\sum_{j=1}^{N}\mathbb{P}\left(\hat{y}_j \neq y_j\right) \leq (L-1)\cdot \exp\left\{-\frac{1}{2}\left(\frac{L}{L-1}\right)^2 Mq^2\left(\bar{w}-\frac{1}{L}\right)^2\right\}. \tag{2.28}$$

Meanwhile, we have

$$\frac{1}{N}\sum_{j=1}^{N}\mathbb{P}\left(\hat{y}_j \neq y_j\right) \leq (L-1)\cdot \exp\left\{-\frac{\frac{1}{2}\left(\frac{L}{L-1}\right)^2 Mq\left(\bar{w}-\frac{1}{L}\right)^2}{1+\frac{L}{3(L-1)}(\bar{w}-\frac{1}{L})}\right\}. \tag{2.29}$$

When $q < \frac{3}{4}$, *the second upper bound (2.29) is tighter than the first one (2.28).*

Proof. For obtaining the upper bounds, we directly apply Corollary 2.5 by letting $\nu_i = 1$, then $||\nu||_2 = \sqrt{M}$ and note that $\sum_{i=1}^{M}(Lw_i - 1) = LM(\bar{w} - \frac{1}{L})$. By direct simplification, we get the desired result. The only difference in the two bounds is that the exponent in the second one is equal to the first one dividing by the factor $\alpha = q(1 + \frac{L}{3(L-1)}(\bar{w} - \frac{1}{L}))$. Since $\bar{w} \leq 1$, then $\frac{L}{3(L-1)}(\bar{w} - \frac{1}{L}) \leq \frac{1}{3}$, and $\alpha < 1$ when $q < \frac{3}{4}$. Therefore, $q < \frac{3}{4}$ implies that the second bound is tighter than the first one. □

Remark: 1. In real crowdsourcing applications, it is common that q will be small enough due to the reasonable size of the available crowd (Snow et al., 2008). Thus the second bound will likely be tighter than the first one in practice. 2. The lower bound and its conditions can be easily derived similarly, so we omit them here since we are more interested in the "possibilities" of controlling the error rate to be small.

Due to the importance of majority voting in the crowdsourcing community, we are also interested in asymptotic properties of the bound. The following corollary discusses the case when $M \to \infty$, that is, when the number of workers who label items tends to infinity.

Corollary 2.7. (Majority voting in the asymptotic scenario) *For the Homogenous Dawid-Skene model with task assignment based on constant probability* $q \in (0,1]$, *and let* $\hat{y}_j$ *be the predicted label for the* j*th item via majority voting rule (2.7), for any* $N \geq 1$,

(1) if $\lim_{M\to\infty}\bar{w} > \frac{1}{L}$, *then* $\frac{1}{N}\sum_{j=1}^{N} I(\hat{y}_j \neq y_j) \to 0$ *in probability as* $M \to \infty$;

(2) if $\lim_{M\to\infty}\bar{w} < \frac{1}{L}$, *then* $\frac{1}{N}\sum_{j=1}^{N} I(\hat{y}_j \neq y_j) \to 1$ *in probability as* $M \to \infty$;

(3) if $\lim_{M\to\infty}\bar{w} > \frac{1}{L}$, *then* $\hat{y}_j \to y_j$ *in probablitliy as* $M \to \infty$, $\forall j \in [N]$, *i.e., majority voting is consistent.*

Proof. The setting of this corollary is the same as in Corollary 2.6, and $t_1 = \frac{Lq}{L-1}\sqrt{M}(\bar{w} - \frac{1}{L})$. When $\lim_{M\to\infty}\bar{w} > \frac{1}{L}$, there exists $\eta > 0$ such that $\eta = \lim_{M\to\infty}\bar{w} - \frac{1}{L}$, which implies $\lim_{M\to\infty} t_1 = +\infty$, which further implies $\frac{1}{N}\sum_{j=1}^{N}\mathbb{P}\left(\hat{y}_j \neq y_j\right) = 0$ for any $N \geq 1$ by Corollary 2.6. This is to say that $\lim_{M\to\infty}\mathbb{P}(\hat{y}_j \neq y_j) = \lim_{M\to\infty}\mathbb{P}(\mathrm{I}\left(\hat{y}_j \neq y_j\right) = 1) = 0$, which

further implies that $\lim_{M\to\infty}\mathbb{P}(\mathrm{I}\left(\hat{y}_j \neq y_j\right) = 0) = 1$. Note that for arbitrary $\epsilon > 0$, we have $\mathbb{P}(\mathrm{I}\left(\hat{y}_j \neq y_j\right) < \epsilon) \geq \mathbb{P}(\mathrm{I}\left(\hat{y}_j \neq y_j\right) = 0)$, then $\lim_{M\to\infty}\mathbb{P}(\mathrm{I}\left(\hat{y}_j \neq y_j\right) \geq \epsilon) = 0$, i.e., $\mathrm{I}\left(\hat{y}_j \neq y_j\right) \to 0$, $\forall j \in [N]$ in probability when $M \to \infty$. By the properties of convergence in probability, $\frac{1}{N}\sum_{j=1}^{N}\mathrm{I}\left(\hat{y}_j \neq y_j\right) \to 0$ in probability when $M \to \infty$. With the same argument, one can prove (2) as well. To prove (3), note that $\mathbb{P}(\hat{y}_j - y_j = 0) = \mathbb{P}(\mathrm{I}\left(\hat{y}_j \neq y_j\right) = 0) \to 0$ as $M \to \infty$, which implies $\hat{y}_j \to y_j$ in probability as $M \to \infty$. □

Remark: This corollary tells us that, if the average quality of the workers (i.e. the accuracy of labeling items) in the worker population, is better than random guess, then all the items can be labeled correctly with arbitrarily high probability when there are enough workers available. The consistency property of majority voting for finite number of items ensures us that as long as there are enough reliable workers (better than random guess) available, we can achieve arbitrary accuracy even by a simple aggregating approach — majority voting.

The examples we covered in this section do not require the estimation of parameters such as worker reliabilities. In the next section, we discuss the maximizing likelihood methods for inferring the parameters in crowdsourcing models, such as the celebrated EM (Expectation-Maximization) algorithm. Then we illustrate how our main results can be applied to analyze an underlining method that the ML methods approximate.

2.4.2 Error rate bounds for the Maximum A Posteriori rule

If we know the label posterior distribution of each item defined in (2.5), then the Bayes classifier is

$$\hat{y}_j = \underset{k\in[L]}{\operatorname{argmax}}\,\rho_k^{(j)}, \tag{2.30}$$

which is well known to be the optimal classifier (Duda et al., 2012).

In reality, we do not know the true parameters of the model, thus the true posterior remains unknown. One natural way is to estimate the parameters of the model by Maximum Likelihood methods, further estimate the posterior of each label class for each item, and then build a classifier based on that. This is usually called the *Maximum A Posteriori* (MAP) approach (Duda et al., 2012). After the EM algorithm estimating parameters, the Maximum A Posteriori (MAP) rule, which predicts the label of an item as the one that has the largest estimated posterior, can be applied. The prediction function of such a rule is

$$\hat{y}_j = \underset{k\in[L]}{\operatorname{argmax}}\,\hat{\rho}_k^{(j)}, \tag{2.31}$$

where $\hat{\rho}_k^{(j)}$ is the estimated posterior. If the MAP rule is applied after the parameters are learned by the EM algorithm, then we call this method the *EM-MAP rule*, and sometimes simply refer it to *EM* without introducing any ambiguity in the context.

However, the EM algorithm cannot guarantee convergence to the global optimum. Thus, the estimated parameters might be biased from the true parameters and the estimated posterior might be far away from the true one if it starts from a "bad" initialization. Moreover, it is generally hard to study the solution of the EM algorithm, and thus it is relatively difficult for us to obtain the error rate for the EM-MAP rule.

We consider the *oracle MAP rule*, which assumes there is an oracle who knows the true parameters and uses the true posterior to predict labels. Hence the oracle MAP rule is the Bayes classifier (2.30), and recall that its prediction function is $\hat{y}_j = \text{argmax}_{k\in[L]}\, \rho_k^{(j)}$, where $\rho_k^{(j)}$ is the true posterior of $y_j = k$. Based on our empirical observations, the EM-MAP rule approximates the oracle MAP rule well in performance when most of the workers are good (better than random guess). In the next, we provide an error rate bound for the oracle MAP rule, which hopefully will help us understand the EM-MAP rule better.

The following result is about the error rate bounds on the oracle MAP rule, and it can be straightforwardly derived from the main results in Section 2.3 since the oracle MAP rule is a decomposable rule as in (2.9).

Corollary 2.8. (Error rate bounds of the oracle MAP rule under the General Dawid-Skene model) *Suppose there is an oracle that knows the true parameters* $\Theta = \left\{ \left\{ P^{(i)} \right\}_{i\in[M]}, Q, \pi \right\}$ *where* $P^{(i)} = \left(p_{kh}^{(i)} \right)_{k,h\in[L]} \in (0,1]^{L\times L}$. *The prediction function of the oracle MAP rule is* $\hat{y}_j = \text{argmax}_{k\in[L]}\, \rho_k^{(j)}$, *where* $\rho_k^{(j)}$ *is the true posterior. All the error rate bounds in Theorem 2.1, 2.2 and 2.3 hold for the oracle MAP rule with* $f_i(k,h) \doteq \log p_{kh}^{(i)},\ \forall k,h \in [L]$.

The results in the section above help us understand more about the practice of inferring the ground truth labels via maximum likelihood methods. The prominent EM-MAP rule approximates the oracle MAP rule by estimating the parameters of crowdsourcing model and thus estimates the posterior distribution (2.5), then applies the MAP rule to predict the labels of items. A further study on the error rate bounds of the oracle MAP rule might be good for designing better algorithms with performance on par with the EM-MAP rule. This is the focus of the next section.

2.5 Simulation

In this section, we compare the theoretical error rate bound with the error rate of the oracle MAP rule via simulation. All results are averaged over 100 random trials. All our experiments are implemented in Matlab 2012a, and run on a PC with Windows 7 operation system, Intel Core i7-3740QM (2.70GHz) CPU and 8GB memory.

The error rate of a crowdsourcing system is affected by variations of different parameters such as number of workers M, number of items N and worker reliabilities $\{w_i\}_{i\in[M]}$ etc. To study how the error rate bound reflects the change of error rate when a parameter of the system changes, we conduct numerical experiments on simulated data for comparing the

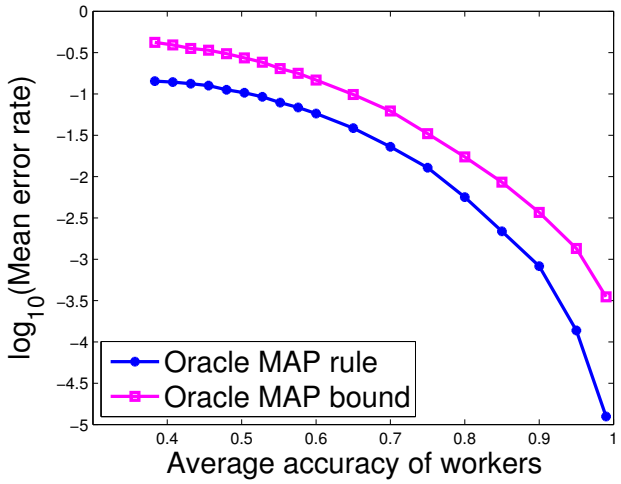


Figure 2.4: Comparing the oracle MAP rule with its theoretical error rate bound by simulation. These simulations are done under the Homogenous Dawid-Skene model with $L = 3$ and $q = 0.3$. Vary the average accuracy of workers and fix $M = 31$ and $N = 200$. Note that all of figures in this section are in log scale and all the results are averaged across 100 repetitions.

oracle MAP rule with its error rate bound (Corollary 3.2). We also measure the performance of the IWMV algorithm and compare it with the performance of oracle MAP rule.

The simulations are run under the Homogenous Dawid-Skene model. Each worker has q =30% chance to label any item which belongs to one of three classes ($L = 3$). The ground truth labels of items are uniformly generated. The accuracies of workers (i.e., $\{w_i\}$) are sampled from a beta distribution $Beta(a, b)$ with $b = 2$. Given an expected average worker accuracy $\bar{w}$, we choose the parameter $a = \frac{2\bar{w}}{1-\bar{w}}$ so that the expected value for worker accuracies under distribution $Beta(a, 2)$ matches with $\bar{w}$. In each random trial in the simulation, we keep sampling M workers from this $Beta$ distribution until the average worker accuracy is within ± 0.01 range from the expected $\bar{w}$. This is to maintain the average worker accuracy at the same level for each trial.

First of all, we fix $M = 31$ and $N = 200$, then control the expected accuracy of the workers varies from 0.38 (slightly larger than random guess $1/L$) to 1 with a step size 0.05. The averaged error rates are displayed in Figure 2.4. Note that the error rate of the oracle MAP rule is bounded by its mean error rate bound tightly (see Figure 2.4). The bound follows the same trend of the true error rate of the oracle MAP rule. The performance of IWMV converges to that of the oracle MAP rule quickly as $\bar{w}$ increases.

By fixing $a = 2.3$ and $q = 0.3$, we then vary one of the two parameters— number of items N (default as 200) and number of workers M (default as 31) — with the other parameters maintained as the default. The corresponding results are presented in Figure 2.5 and 2.6,

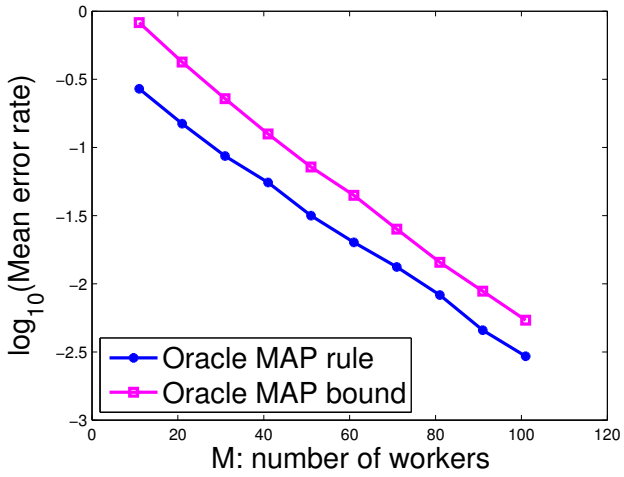


Figure 2.5: Comparing the oracle MAP rule with its theoretical error rate bound by simulation by varying M. These simulations are done under the Homogenous Dawid-Skene model with $L = 3$, $q = 0.3$ and $N = 200$. The reliabilities of workers are sampled based on $w_i \sim$Beta(2.3, 2), $\forall i \in [M]$.

respectively. According to the results of simulation, the error rate bound of the oracle MAP rule and its upper bound do not change when the number of tasks N increases (Figure 2.6), but they change log linearly when the number of workers M increases (Figure 2.5). The upper bound on the oracle MAP rule varies between log linearly and log quadratically when the probability of task assignment q increases (Figure 2.7).

2.6 Discussion and conclusions

In practice, if we want to obtain the error rate bounds for certain aggregation rules that fall in the form of decomposable aggregation rule (2.9), what we can do should be similar to what we did in Section 2.4: (1) for the specific model and task assignment, compute the measure of t_1 and t_2 (and also c and σ^2 if interested in bounds on the mean error rate) according to the descriptions in Section 2.3.1. (2) Compute the corresponding error rate bounds according to the theorems in Section 2.3. The quantities t_1 and t_2 can tell us if we can obtain upper bound or lower bound on the error rate both in probability and in expectation. Note that though the mean error rate bounds (Theorem 2.3) are in a composite form of two exponential bounds, we can choose one of the two to use if the convenience of theoretical analysis is concerned.

In this chapter, we have provided bounds on error rate of general decomposable rules under the Dawid-Skene crowdsourcing model that includes both the Class-Conditional Dawid-Skene and Homogenous Dawid-Skene model as special cases. Our bounds are in probability

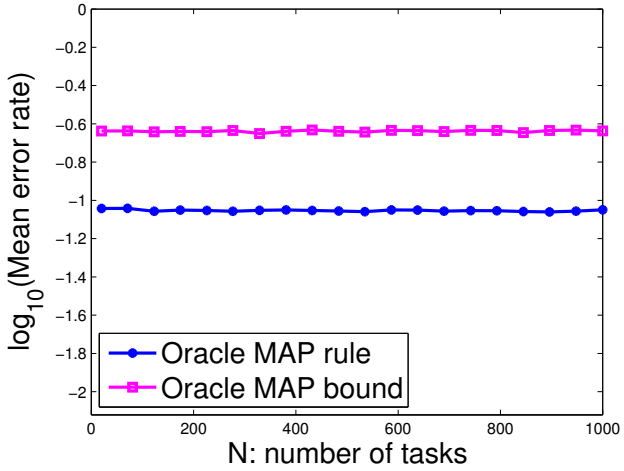


Figure 2.6: Comparing the oracle MAP rule with its theoretical error rate bound by simulation by varying N. These simulations are done under the Homogenous Dawid-Skene model with $L = 3$, $q = 0.3$ and $M = 31$. The reliabilities of workers are sampled based on $w_i \sim$Beta(2.3, 2), $\forall i \in [M]$.

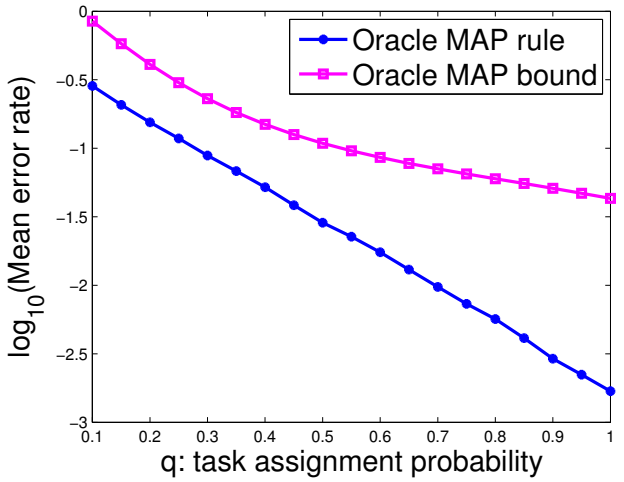


Figure 2.7: Comparing the oracle MAP rule with its theoretical error rate bound by simulation by varying q. These simulations are done under the Homogenous Dawid-Skene model with $L = 3$, $M = 31$ and $N = 200$. The reliabilities of workers are sampled based on $w_i \sim$Beta(2.3, 2), $\forall i \in [M]$.

and in expectation, and they hold for finite numbers of workers and items.

To the best of our knowledge, this is the first extensive work on error rate bounds for gen-

eral aggregation rules under the practical Dawid-Skene model for multi-class crowdsourced labeling. Our bounds are useful for explaining the effectiveness of different aggregation rules and also motivating us to design the iterative weighted majority voting method, which will be presented in the next chapter.

As a future direction for research, it is worth investigating finite sample error rate bounds for the aggregation rules with random score functions, which depend on the data in a complicated manner. For example, the EM-MAP rule can be formulated as a weighted majority voting under the Homogenous Dawid-Skene model. However, the weights are estimated by the EM algorithm (Raykar et al., 2010) and depend on the data complicatedly. Hence the analysis of the EM algorithm is rather difficult, but it will be very useful if we can make any progress on understanding the behavior of the EM algorithm and improving the inference methods in crowdsourcing in general.

Chapter 3

Iterative weighted majority voting method for crowdsourcing

3.1 Motivation

The previous chapter discusses the error rate of general aggregation rules under the General Dawid-Skene model. In this chapter, we demonstrate how the bounds in Chapter 2 can be used to design an iterative weighted majority voting algorithm for inferring the underlying true labels of items.

A crucial problem in crowdsourcing is to design aggregation method so that the quality of the yielded results is as good as possible. As a commonly used method (Smyth et al., 1995, Snow et al., 2008, Sheng et al., 2008), *majority voting* (MV) simply adopts the most common label from the inputs of the workers. However, majority voting treats the input from all workers equally, and it is well-known to be error-prone and inefficient (Karger et al., 2011a). In principle, efficient aggregation approaches should take the reliabilities of different workers into account.

Beyond majority voting, significant efforts have been made to investigate advanced aggregation methods which jointly estimate the reliabilities of workers and labels of items without ground truth (Dawid and Skene, 1979, Whitehill et al., 2009, Raykar et al., 2010, Karger et al., 2011a, Liu et al., 2012, Zhou et al., 2012). The basic paradigm is to model the reliabilities of workers and labels, and then to infer the true labels with EM algorithm (Dawid and Skene, 1979, Raykar et al., 2010) or other inference techniques(Karger et al., 2011a, Liu et al., 2012, Zhou et al., 2012). However, as a natural and intuitive extension to majority voting, the underlining power and limitation of *weighted majority voting* (WMV), which assigns different weights to workers and chooses the label with the highest accumulated weight, have not been well studied to the best of our knowledge. It is still unknown how we could possibly choose the weight based on the worker inputs, and achieve the best performance one can get.

In this chapter, we investigate the weighted majority voting method for multi-class la-

beling tasks in crowdsourcing. The contributions of this chapter are: (1) We obtain finite sample bounds on the expected error rate of weighted majority voting based on the main results from previous chapter. (2) We show that the oracle Maximum A Posteriori (MAP) rule is a weighted majority voting, and it approximately minimizes the error rate bound of weighted majority voting. (3) We propose an intuitive and easy-to-implement algorithm called iterative weighted majority voting (IWMV) to optimize the error rate bound of WMV. Experimental results on both simulation and real data show that IWMV performs at least on par with the state-of-the-art methods, while it has lower computation cost than the state-of-the-art methods.

The rest of the chapter is organized as follows. In Section 3.2, we address the notations and crowdsourcing model used in this chapter. In Section 3.4, we present the error rate bound of WMV. We derive the bound optimal rule in Section 3.5.1, and propose IWMV algorithm in Section 3.5. In Section 3.6, we show the experimental results on both synthetic and real-world data. We conclude this chapter in Section 3.7.

3.2 Background

We follow the nations in Section 2.2 in Chapter 2. Recall that there are M workers and N items for labeling, and each item belongs to one of the L classes. For convenience, we denote the set of workers $[M]$, the set of items $[N]$, and the set of label classes $[L]$, where $[M]$ represents the first M integers. y_j denotes the true label for j-th item, and $\hat{y}_j$ the predicted label for the j-th item by an algorithm. Let $\pi_k = \mathbb{P}(y_j = k)$ denote the prevalence of label "k" in the true labels of the items for any $j \in [N]$ and $k \in [L]$. When task j (i.e., labeling the jth item) is assigned to worker i, a possibly noise label Z_{ij} is collected, otherwise we denote $Z_{ij} = 0$, which means unobserved/missing. Hence, a data matrix $Z \in \overline{[L]}^{M \times N}$ is formed after collecting all the data from workers, where $\overline{[L]} = [L] \cup \{0\}$. The task assignments are denoted by an indicator matrix $T = (T_{ij})_{M \times N}$, where $T_{ij} = 1$ indicates the i-th worker has labeled the j-th item, otherwise Z_{ij} is unobserved. We further assume that the task assignment is random, formally, $\mathbb{P}(T_{ij} = 1) = q \in (0, 1]$.[1]

We adopt a commonly used model in (Raykar et al., 2010, Karger et al., 2011a, Liu et al., 2012) to capture the diversity of worker reliabilities: the Homogenous Dawid-Skene model (Section 2.2). This model assumes the reliability of worker i is measured by a single parameter w_i such that

$$\mathbb{P}(Z_{ij} = l | y_j = k) = \begin{cases} w_i & \text{if} \quad l = k \\ \frac{1-w_i}{L-1} & \text{if} \quad l \neq k. \end{cases} \tag{3.1}$$

Therefore w_i is the probability of that worker i labeling an item correctly. The parameters of

[1] If we further relax our assumption about task assignment to full generality as $\mathbb{P}(T_{ij} = 1) = q_{ij} \in (0, 1]$, our analysis in this chapter can also be applied. For clarity, we adopt this simple assumption.

the crowdsourcing models we describes above is $\Theta \doteq \left\{ \{\pi_k\}_{k\in[L]}, \{w_i\}_{i\in[M]} \right\}$ and the posterior distribution of true labels is $\rho_k^{(j)} \doteq \mathbb{P}\left(y_j = k | Z, \Theta\right), \forall j \in [N], k \in [L]$.

Following the same performance measure from Chapter 2: given an aggregation rule, suppose that its predicted label for item j is $\hat{y}_j$, a common used performance measure is the error rate $\mathrm{ER} = \frac{1}{N}\sum_{j=1}^{N} \mathrm{I}\left(\hat{y}_j \neq y_j\right)$, where $\mathrm{I}\left(\cdot\right)$ is the indicator function. Due to the random nature of the error rate, we are usually interested in its expected value, i.e., the *mean error rate*. Formally, the mean error rate is:

$$\mathrm{MER} \quad = \quad \frac{1}{N}\sum_{j=1}^{N} \mathbb{P}(\hat{y}_j \neq y_j). \tag{3.2}$$

A core problem in crowdsourcing is to design effective algorithms for inferring the underlying true labels based on the noisy inputs from crowds. The major objective of designing such inference algorithms is to reduce the error rate with high probability or the mean error rate. In next section, we present the Maximizing Likelihood method to infer the groundtruth labels.

3.3 Learning crowdsourcing models via Maximum Likelihood method

Recall that if we know the posterior distribution of the label of each item defined as in (2.5), then the Bayesian classifier is

$$\hat{y}_j = \underset{k\in[L]}{\operatorname{argmax}}\, \rho_k^{(j)}$$

One natural way to approach it is to use Maximum Likelihood methods to estimate the parameters of the model, and then further compute the posterior of the label of each time.

3.3.1 Estimating parameters by Expectation Maximization

The maximum likelihood method is a commonly used method for inference parameters in graphical models. However, when there is hidden variables, the maximum likelihood estimate (MLE) is general intractable to derive. A widely used alternative is applying Expectation Maximization (EM) algorithm (Dempster et al., 1977), which is an iterative method for optimizing a lower bound of the likelihood. Under our setting, the objective function of EM algorithm (Bilmes, 1998) will be

$$\mathcal{Q}(\Theta, \Theta^{'}) \quad = \quad \mathbb{E}_{Y|Z,T,\Theta'}[\log \mathbb{P}(Z, T, Y|\Theta)],$$

where $Y = (y_1, y_2, \cdots, y_N)$ as the true labels of all the N items and $\Theta^{'}$ the current estimated parameters. $\mathcal{Q}(\Theta, \Theta^{'})$ can be further factorized by the conditional independences in the

graphical model. Let's use $Z_{\star j} = (Z_{1j}, Z_{2j}, \cdots, Z_{Mj})^T$ denote all the labels associate with the item j,

$$\begin{aligned}
\mathcal{Q}(\Theta, \Theta^{'}) &= \mathbb{E}_{Y|Z,T,\Theta'}[\log \mathbb{P}(Z, Y|T, \Theta)] + \log \mathbb{P}(T|\Theta) \\
&= \sum_{j=1}^{N}\sum_{k=1}^{L} \rho_k^{(j)} \cdot \left[\log \pi_k + \sum_{i=1}^{M}\sum_{l=1}^{L} \mathrm{I}\left(Z_{ij} = l\right) \log(p_{kl}^{(i)})\right] \\
&\quad + \sum_{i=1}^{M}\sum_{j=1}^{N} \left[\mathrm{I}\left(Z_{ij} \neq 0\right) \log(q_{ij}) + \mathrm{I}\left(Z_{ij} = 0\right) \log(1 - q_{ij})\right] \qquad (3.3)
\end{aligned}$$

The EM algorithm iteratively executes two steps to compute posterior and estimate parameters of interest for General Dawid-Skene model as follows:

(E-step): compute posterior distrubtion based on the current estimated parameters by

$$\rho_k^{(j)} = \frac{\pi_k \eta_k^{(j)}}{\sum_{l=1}^{L} \pi_l \eta_l^{(j)}}, \quad \forall j \in [N], k \in [L], \qquad (3.4)$$

where $\eta_k^{(j)}$ is the likelihood of $Z_{\star j}$,

$$\eta_k^{(j)} = \mathbb{P}(Z_{\star j}|y_j = k, T, \Theta) = \prod_{i=1}^{M}\prod_{l=1}^{L} \left(p_{kl}^{(i)}\right)^{\mathrm{I}(Z_{ij}=l)}. \qquad (3.5)$$

(M-step): estimate parameters of interest by optimize

$$\max_{\Theta} \quad \mathcal{Q}(\Theta, \Theta^{'}) \qquad \text{subject to} \qquad \sum_{k=1}^{L} \pi_k = 1. \qquad (3.6)$$

This leads to the parameter update as

$$p_{kl}^{(i)} = \frac{\sum_{j=1}^{N} \rho_k^{(j)} \mathrm{I}\left(Z_{ij} = l\right)}{\sum_{j=1}^{N} \rho_k^{(j)} T_{ij}} \qquad \forall i \in [M], k, l \in [L], \qquad (3.7)$$

$$\pi_k = \frac{1}{N}\sum_{j=1}^{N} \rho_k^{(j)} \qquad \forall k \in [L]. \qquad (3.8)$$

The E-step in EM algorithm for Class-Conditional Dawid-Skene model and Homogenous Dawid-Skene model are the same as for General Dawid-Skene model — replace $p_{kl}^{(i)}, l \neq k$ with $\frac{1-p_{kk}^{(i)}}{L-1}$ for Class-Conditional Dawid-Skene model, and replace $p_{kk}^{(i)}$ with w_i, and $p_{kl}^{(i)}, l \neq k$ with $\frac{1-w_i}{L-1}$ for Homogenous Dawid-Skene model.

Algorithm 1 EM algorithm under Homogenous Dawid-Skene model for crowdsourcing.

Input: Number of workers= M; Number of items= N; label matrix: $Z \in \overline{[L]}^{M \times N}$;
Output: the predicted labels $\{\hat{y}_1, \hat{y}_2, ..., \hat{y}_N\}$
Initialization: $T_{ij} \leftarrow \mathrm{I}\left(Z_{ij} \neq 0\right), \forall i \in [M], \forall j \in [N]$; Intialize $\{\hat{w}_i\}_{i\in[M]}$ and $\{\hat{\pi}_k\}_{k\in[L]}$ with the result from majority voting.
repeat

——— **E-Step:** ———

for $j = 1, \cdots, N$, the posterior updates: **do**

$$\begin{aligned} \hat{\eta}_k^{(j)} &= \left(\prod_{i=1}^{M} (\hat{w}_i)^{\mathrm{I}(Z_{ij}=k)} \left(\frac{1-\hat{w}_i}{L-1}\right)^{\mathrm{I}(Z_{ij}\neq k,0)}\right), \quad \forall k \in [L]; \\ \hat{\rho}_k^{(j)} &= \frac{\hat{\pi}_k \hat{\eta}_k^{(j)}}{\sum_{l=1}^{L} \hat{\pi}_l \hat{\eta}_l^{(j)}}, \quad \forall k \in [L]. \end{aligned}$$

end for

——— **M-Step:** ———

for $i = 1, \cdots, M$, the parameters updates: **do**

$$\hat{w}_i = \frac{\sum_{j=1}^{N} \sum_{k=1}^{L} \hat{\rho}_k^{(j)} \mathrm{I}\left(Z_{ij} = k\right)}{\sum_{j=1}^{N} T_{ij}} \quad .$$

end for

$$\hat{\pi}_k = \frac{1}{N} \sum_{j=1}^{N} \hat{\rho}_k^{(j)}, \quad k \in [L].$$

until reaches certain number of iterations.
Output the prediction: $\forall j \in [N]$, output $\hat{y}_j = \mathrm{argmax}_{k\in[L]}\, \hat{\rho}_k^{(j)}$.

The M-step are slightly different from the General Dawid-Skene model for updating worker quality parameter, which is listed as follows.
M-step under Class-Conditional Dawid-Skene model:

$$p_{kk}^{(i)} = \frac{\sum_{j=1}^{N} \rho_k^{(j)} \mathrm{I}\left(Z_{ij} = k\right)}{\sum_{j=1}^{N} T_{ij}} \quad \text{and} \quad p_{kl}^{(i)} = \frac{1 - p_{kk}^{(i)}}{L-1}, \qquad \forall i \in [M], l, k \in [L]. \tag{3.9}$$

M-step under Homogenous Dawid-Skene model:

$$w_i = \frac{\sum_{j=1}^{N}\sum_{k=1}^{L}\rho_k^{(j)}\mathrm{I}\left(Z_{ij}=k\right)}{\sum_{j=1}^{N}T_{ij}}, \qquad \forall i \in [M]. \tag{3.10}$$

Note that for all three models, the update for π_k in M-step are the same.

As an illustration, the detailed EM algorithm of Homogenous Dawid-Skene model is listed in Alg.1. The algorithm framework can be slightly modified for both General Dawid-Skene and Class-Conditional Dawid-Skene model.

3.3.2 Maximum a posteriori rule

We can apply EM algorithm to estimate the parameters, but after that, we need a rule to decide the item should be labeled with which label. One popular rule for classification is the maximum a posteriori (MAP) rule, which assigned the item to the label that has the largest estimated posterior.

The prediction function for a MAP rule is

$$\hat{y}_j = \underset{k\in[L]}{\operatorname{argmax}}\,\hat{\rho}_k^{(j)} \tag{3.11}$$

, where $\hat{\rho}_k^{(j)}$ is the estimated posterior. In Alg.1, we have applied the MAP rule with the estimated posterior. If the parameters are learned by EM algorithm and then the MAP rule is applied to predict the labels of items, then we called it the *EM-MAP rule*.

However, EM algorithm can not guarantee to converge to the global optimum. Thus, the estimated parameters might be biased from the true parameters and the estimated posterior might be far away from the true one if it starts from a "bad" initialization.

Now, we consider about the *oracle MAP rule*, which assume there is an oracle who knows the true parameters and thus use the true posterior in MAP rule to label items. The prediction function of the oracle MAP rule is

$$\hat{y}_j = \underset{k\in[L]}{\operatorname{argmax}}\,\rho_k^{(j)},$$

where $\rho_k^{(j)}$ is the true posterior of $y_j = k$.

In next section, we show that the oracle MAP rule is a weighted majority voting rule. We provide a bound on the mean error rate of weighted majority voting, therefore we obtain the error rate bound on the oracle MAP rule.

3.4 Finite sample error rate bound of weighted majority voting

In weighted majority voting, each worker is assigned with a weight, and the aggregation label will be the one with most accumulated weight. Formally, the aggregation rule of WMV

is

$$\hat{y}_j = \underset{k\in[L]}{\operatorname{argmax}} \sum_{i=1}^{M} \nu_i \mathrm{I}\left(Z_{ij} = k\right), \tag{3.12}$$

where $\nu_i \in \mathbb{R}$ is the weight associated with the ith worker. Majority voting is a special case of WMV when all the workers have the same positive weight such as $\nu_i = 1$ for all i.

Corollary 3.1. (Error rate bound of weighted majory voting) *For weighted majority voting (2.8) with weights of workers $\nu = (\nu_1, \nu_2, \cdots, \nu_M)$. Assume the task assignment between workers and items is based on probability $q \in (0, 1]$, and assume the workers are modeled with parameter $\{w_i\}_{i=1}^{M}$, then with*

$$t = \frac{q}{(L-1)||\nu||_2} \sum_{i=1}^{M} \nu_i (L w_i - 1), \tag{3.13}$$

(i) if $t \geq 0$, *then* $\frac{1}{N}\sum_{j=1}^{N} \mathbb{P}\left(\hat{y}_j \neq y_j\right) \leq e^{-\frac{t^2}{2}+\ln(L-1)}$,
(ii) if $t \leq 0$, *then* $\frac{1}{N}\sum_{j=1}^{N} \mathbb{P}\left(\hat{y}_j \neq y_j\right) \geq 1 - e^{-\frac{t^2}{2}}$.

Proof. This result is implied by Corollary 2.5 directly. □

Remark: The quantity t is crucial to bound the error rate of WMV and it depends on w, ν and q — how reliable the workers are for this type of tasks, how the task owner assigns weight to workers, and *how many task assignments are done* (i.e., the proportion of the observed entries in data matrix). Based on t, we can judge whether an upper bound or a lower bound on the error rate can be obtained.

If we let $\nu_i = 1$ for all i, with the measure $t = \frac{q}{(L-1)\sqrt{M}} \sum_{i=1}^{M}(L w_i - 1)$, the bounds in Corollary 3.1 (i) and (ii) apply to majority voting.

The result above leads to the next corollary, which shows that the oracle MAP rule is a weighted majority voting rule, and gives its mean error rate bound.

Corollary 3.2. (Error rate bound of the oracle MAP rule) *Suppose the task assignment is based on probability $q \in (0, 1]$ and true labels are class-balanced (i.e.,$\pi_k = 1/L, \forall k \in [L]$), and assume $w_i \in (0, 1), \forall i \in [M]$, then the oracle MAP rule is a weighted majority voting rule with weight $\nu_i' = \ln \frac{(L-1)w_i}{1-w_i}$ for worker i. Let $t_1 = \frac{q}{(L-1)\|\nu'\|} \sum_{i=1}^{M} \nu_i'(L w_i - 1)$, the mean error rate of the oralce MAP rule always has an upper bound as $\frac{1}{N}\sum_{j=1}^{N} \mathbb{P}\left(\hat{y}_j^{omap} \neq y_j\right) \leq e^{-\frac{t_1^2}{2}+\ln(L-1)}$.*

Proof. Based on the model of worker reliabilities, we have $\mathbb{P}(Z_{ij} = k | y_j = k) = w_i$ and $\mathbb{P}(Z_{ij} = h | y_j = k) = \frac{1-w_i}{L-1}$ for all $k, h \in [L], k \neq h$.

The posterior distribution is

$$\rho_k^{(j)} = \frac{\pi_k \eta_k^{(j)}}{\sum_{l=1}^{L} \pi_l \eta_l^{(j)}}, \quad \forall j \in [N], k \in [L],$$

where

$$\eta_k^{(j)} = \prod_{i=1}^{M} (w_i)^{\mathrm{I}(Z_{ij}=k)} \left(\frac{1-w_i}{L-1}\right)^{\mathrm{I}(Z_{ij}\neq k,0)}.$$

For the oracle MAP rule, we have

$$\begin{aligned}
\hat{y}_j^{\text{oracle}} &= \underset{k\in[L]}{\operatorname{argmax}}\, \rho_k^{(j)} \\
&= \underset{k\in[L]}{\operatorname{argmax}}\, \pi_k \eta_k^{(j)} \\
&= \underset{k\in[L]}{\operatorname{argmax}}\, \log(\eta_k^{(j)}) + \log \pi_k \\
&= \underset{k\in[L]}{\operatorname{argmax}} \sum_{i=1}^{M} \left(\mathrm{I}\left(Z_{ij}=k\right) \log w_i + \left(\mathrm{I}\left(Z_{ij}\neq 0\right) - \mathrm{I}\left(Z_{ij}=k\right)\right) \log \frac{1-w_i}{L-1} \right) + \log \frac{1}{L} \\
&= \underset{k\in[L]}{\operatorname{argmax}} \sum_{i=1}^{M} \left(\mathrm{I}\left(Z_{ij}=k\right) \log w_i + \mathrm{I}\left(Z_{ij}\neq k,0\right) \log \frac{1-w_i}{L-1} \right) \\
&= \underset{k\in[L]}{\operatorname{argmax}} \sum_{i=1}^{M} \left(\mathrm{I}\left(Z_{ij}=k\right) \log w_i + \left(\mathrm{I}\left(Z_{ij}\neq 0\right) - \mathrm{I}\left(Z_{ij}=k\right)\right) \log \frac{1-w_i}{L-1} \right) \\
&= \underset{k\in[L]}{\operatorname{argmax}} \left[\sum_{i=1}^{M} \mathrm{I}\left(Z_{ij}=k\right) \log \frac{(L-1)w_i}{1-w_i} + \sum_{i=1}^{M} \mathrm{I}\left(Z_{ij}\neq 0\right) \log \frac{1-w_i}{L-1} \right] \\
&= \underset{k\in[L]}{\operatorname{argmax}} \sum_{i=1}^{M} \left(\log \frac{(L-1)w_i}{1-w_i} \right) \mathrm{I}\left(Z_{ij}=k\right) \\
&= \underset{k\in[L]}{\operatorname{argmax}} \sum_{i=1}^{M} \nu_i' \mathrm{I}\left(Z_{ij}=k\right),
\end{aligned}$$

where $\nu_i' = \log \frac{(L-1)w_i}{1-w_i}$. Therefore the oracle MAP rule is a weigthed majoirity rule. Thus the results from Corollary 3.1 can be directly applied here. i.e., with

$$t_1 = \frac{q}{(L-1)||\nu'||^2} \sum_{i=1}^{M} \nu_i'(Lw_i - 1),$$

if $t_1 \geq 0$, then

$$\frac{1}{N} \sum_{j=1}^{N} \mathbb{P}\left(\hat{y}_j \neq y_j\right) \leq (L-1) \exp\left(-\frac{t_1^2}{2}\right).$$

Now all we need to show is that t_1 is always non-negative. We can see that if $w_i \geq \frac{1}{L}$, then $(Lw_i - 1) \geq 0$ and $\nu_i = \frac{(L-1)w_i}{1-w_i} \geq 0$ for all $i \in [M]$, then $t_1 \geq 0$ in this case. $w_i < \frac{1}{L}$, then $(Lw_i - 1) < 0$ and $\frac{(L-1)w_i}{1-w_i} < 0$, thus $t_1 > 0$ in this case as well. All in all, $t_1 \geq 0$ is always true for the oralce MAP rule. □

Remark: There are two messages the corollary above implies: (1) The oracle MAP rule is a weighted majority voting. (2) Since $t_1 \geq 0$ always holds, the mean error rate of the oracle MAP rule is always upper bounded. That is different from Corollary 3.1 because the oracle MAP rule wisely chooses the weight for each worker so that the t_1 will be always non-negative.

3.5 Iterative weighted majority voting algorithm

In this section, we first study the mean error rate bound of weighted majority voting, then we minimize the bound to get the oracle bound-optimal rule. Finally, we present its connection to the oracle MAP rule. Based on the oracle bound-optimal rule, we propose an iterative weighted majority voting method with performance guarantee on its one-step version.

3.5.1 The oracle bound-optimal rule and the oracle MAP rule

We explore the relationship between the oracle MAP rule and the error rate bound of WMV. Note that the error rate bound of WMV in Corollary 3.1 implies that if $t \geq 0$, then $\frac{1}{N}\sum_{j=1}^{N} \mathbb{P}\left(\hat{y}_j \neq y_j\right) \leq e^{-\frac{t^2}{2}+\ln(L-1)}$, where the bound $e^{-\frac{t^2}{2}+\ln(L-1)}$ are monotonely decreasing w.r.t. $t \in [0, \infty)$. The mean error rate is upper bounded with the condition $t \geq 0$, thus maximizing t will increase the chance of $t \geq 0$ being satisfied and reduce the error bound. With this observation, the bound-optimal weights of workers in weighted majority voting is

$$\begin{aligned} \nu^\star &= \operatorname*{argmin}_{\nu \in \mathbb{R}^M} e^{-\frac{t^2}{2}+\ln(L-1)} = \operatorname*{argmax}_{\nu \in \mathbb{R}^M} t = \operatorname*{argmax}_{\nu \in \mathbb{R}^M} \frac{q}{L-1} \sum_{i=1}^{M} \frac{\nu_i}{||\nu||_2}(Lw_i - 1) \\ &\Longrightarrow \quad \textbf{The oracle bound-optimal rule:} \quad \text{WMV with } \nu_i^\star \propto Lw_i - 1. \end{aligned} \tag{3.14}$$

Therefore a bound-optimal strategy is to choose the weights for WMV according to (3.14). Since this rule requires the information of the true parameters $\{w_i\}_{i\in[M]}$, we call it as *the oracle bound-optimal rule*. In practice, we can estimate the parameters, and plugin them to the oracle bound-optimal rule, which we refer to as *the bound-optimal rule*.

By Corollary 3.2, the oracle MAP rule under the Homogenous Dawid-Skene model is a weighted majority voting rule with weight

$$\nu_i' = \log \frac{(L-1)w_i}{1-w_i} \approx \frac{L}{L-1}(Lw_i - 1).$$

The approximation is due to the Taylor expansion around $x = \frac{1}{L}$,

$$\ln \frac{(L-1)x}{1-x} = \frac{L}{L-1}(Lx-1) + O\left(\left(x - \frac{1}{L}\right)^2\right). \tag{3.15}$$

Thus, *the weight of the oracle bound-optimal rule is the first order Taylor expansion of the weight in the oracle MAP rule.* Similar result and conclusion hold for the Class-Conditional Dawid-Skene model as well, but we omit them here for clarity.

By observing that the oracle MAP rule is very close to the oracle bound-optimal rule, the oracle MAP rule approximately optimizes the upper bound of the mean error rate. This fact also indicates that our bound is meaningful since the oracle MAP rule is the oracle Bayes classifier.

3.5.2 Iterative weighted majority voting with performance guarantee

Based on Section 3.5.1, the oracle bound-optimal rule of choosing weights is $\nu_i \propto Lw_i - 1$. With this strategy, if we have an estimated w_i, we can put more weights to the "better" workers and downplay the "spammers" (those workers with accuracy close to random guess). This strategy can potentially improve the performance of majority voting and result in a better estimate for w_i, which further improves the quality of the weights, and iterate. This inspires us to design an iterative weighted majority voting (IWMV) method as in Algorithm 2.

Algorithm 2 The iterative weighted majority voting algorithm (IWMV)

Input: Number of workers= M; Number of items= N; data matrix: $Z \in \overline{[L]}^{M\times N}$;
Output: the predicted labels $\{\hat{y}_1, \hat{y}_2, ..., \hat{y}_N\}$
Initialization: $\nu_i = 1, \quad \forall i \in [M]; \quad T_{ij} = \mathrm{I}(Z_{ij} \neq 0), \forall i \in [M], \forall j \in [N]$.
repeat

$$\hat{y}_j \leftarrow \underset{k\in[L]}{\operatorname{argmax}} \sum_{i=1}^{M} \nu_i \mathrm{I}(Z_{ij} = k), \qquad \forall j \in [N].$$

$$\hat{w}_i \leftarrow \frac{\sum_{j=1}^{N} \mathrm{I}(Z_{ij} = \hat{y}_j)}{\sum_{j=1}^{N} T_{ij}}, \qquad \forall i \in [M].$$

$$\nu_i \leftarrow L\hat{w}_i - 1, \qquad \forall i \in [M].$$

until converges or reaches S iterations.
Output the predictions $\{\hat{y}_j\}_{j\in[N]}$ by $\hat{y}_j = \operatorname{argmax}_{k\in[L]} \sum_{i=1}^{M} \nu_i \mathrm{I}(Z_{ij} = k)$.

The time complexity of this algorithm is $O((M + L)NS)$, where S is the number of iterations in the algorithm. Empirically, the IWMV method converges fast. But it also

suffers from the local optimal trap as EM does, and is generally hard to analyze its error rate. However, we are able to obtain the error rate bound in the next theorem for a "naive" version of it – *one-step WMV* (osWMV), which executes **(Step 1)** to **(Step 3)** only once as follows:

(Step 1) Use majority voting to estimate labels, which are treated as the "golden standard", i.e. $\hat{y}_j^{\mathrm{MV}} = \mathrm{argmax}_{k\in[L]} \sum_{i=1}^{M} \mathrm{I}\left(Z_{ij} = k\right)$.

(Step 2) Use the current "golden standard" to estimate the worker accuracy $\hat{w}_i = \frac{\sum_{j=1}^{N} \mathrm{I}\left(Z_{ij}=\hat{y}_j^{\mathrm{MV}}\right)}{\sum_{i=1}^{M} \mathrm{I}(Z_{ij}\neq 0)}$ for all i and set $\nu_i = L\hat{w}_i - 1$ for all i.

(Step 3) Use the current weight v in WMV to estimate an updated "gold standard", i.e., $\hat{y}_j = \mathrm{argmax}_{k\in[L]} \sum_{i=1}^{M} \nu_i \mathrm{I}\left(Z_{ij} = k\right)$.

For the succinctness of the result, we focus on the case where $L = 2$ and $q = 1$, but the techniques used can be applied to the general case of L and q as well.

Theorem 3.3. ***(Mean error rate bound of one step WMV for binary labeling)*** *Under the Homogenous Dawid-Skene model, with label sampling probability $q = 1$ and $L = 2$, let $\hat{y}_j^{wmv}$ be the label predicted by one-step WMV for the jth item, if $\bar{w} \geq \frac{1}{2} + \frac{1}{M} + \sqrt{\frac{(M-1)\ln 2}{2M^2}}$, the mean error rate of one-step WMV*

$$\frac{1}{N}\sum_{j=1}^{N} \mathbb{P}\left(\hat{y}_j^{wmv} \neq y_j\right) \leq \exp\left(-\frac{8MN^2\tilde{\sigma}^4(1-\eta)^2}{M^2N + (M+N)^2}\right), \tag{3.16}$$

where $\tilde{\sigma} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(w_i - \frac{1}{2})^2}$ and $\eta = 2\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}-\frac{1}{M})^2}{M-1}\right)$

The proof of this theorem is deferred to Appendix B. It is non-trivial to prove this theorem since the dependency among the weights and labels makes it hard to apply the concentration approach used in proving the previous results. Instead, a martingale-difference concentration bound has to be used.

Remarks:

(1) In the exponent of the bound, there are several important factors: $\tilde{\sigma}$ represents how far away the accuracies of workers are from random guess, and it is a constant smaller than 1; η will be close to 0 given a reasonable M.

(2) The condition on $\bar{w}$ requires that $\bar{w} - \frac{1}{2}$ is $\Omega(M^{-0.5})$, which is easier to satisfy with M large if the average accuracy in the crowd population is better than random guess. This condition ensures that majority voting approximates the true labels. Thus with more items labeled, we can get a better estimate of the workers' accuracies. The performance of the one-step WMV will then be improved with better weights.

(3) We address how M and N affect the bound: First, when both M and N increase but $\frac{M}{N} = r$ is a constant or decreases, the error rate bound decreases. This makes sense because with the number of items labeled per worker increasing, $\hat{w}_i$ will be more accurate. The weights will be closer to the oracle bound-optimal rule. Second, when M is fixed and N increases, i.e., the number of items labeled increases, the upper bound on the error rate

decreases. Third, when N is fixed and M increases, the bound decreases when $M < \sqrt{N}$ and then increases when M is beyond $\sqrt{N}$. Intuitively, when M is larger than N and M increases, the fluctuation of score functions, where $\hat{w}_i$ is the estimated accuracy of the ith worker, will be large. This increases the chance of making more prediction errors. When M is reasonably small (compared with N) but is increasing, i.e., more people label each item, the accuracy of majority voting will be improved according to Corollary 3.2, then the gain on the accuracy of estimating $\hat{w}_i$ results in the weights of the one-step WMV to be closer to the oracle bound-optimal rule.

As an alternative way of assigning weights to workers in each iteration (Alg.2), we can also choose the weight of worker i by plugging $\hat{w}_i$ into the weight in the oracle MAP rule. That is, $\nu_i' = \log \frac{(L-1)\hat{w}_i}{1-\hat{w}_i}$. We refer this variant of IWMV to the *IWMV.log* algorithm. From the practical point of view, however, ν_i' is unbounded and too large (or too small) if estimator $\hat{w}_i$ is close to 1 (or 0). Therefore IWMV.log uses an aggressive way to weigh the workers, and it might be too risky when the estimates $\{\hat{w}_i\}_{i\in[M]}$ are noisy. Recall that given an estimate $\hat{w}_i$ of the reliability of worker i, the way that the IWMV algorithm chooses the weight is $\nu_i = L\hat{w}_i - 1$. As a linearized version of ν_i' (3.15), ν_i is more stable to the noise in the estimate $\hat{w}_i$. Furthermore, IWMV is more convenient for theoretical analysis than the IWMV.log algorithm. In the next section, we will show some comparisons between IWMV and IWMV.log by experiments.

3.6 Experiments

In this section, we first compare IWMV with EM and IWMV.log on synthetic data. We then experimentally test IWMV and compare it with the state-of-art algorithms on real-world data. We show that overall IWMV is robust to model-misspecification. It achieves the best performance (or sometimes as good as the methods with best performance) while uses much less computational time than the other state-of-the-art methods.

We implement majority voting, EM algorithm (Raykar et al., 2010) with MAP rule (also referred as the EM-MAP rule in the experiments), and use public available code [2] — the iterative algorithm in (Karger et al., 2011a) is referred to as KOS, and the variational inference algorithm from (Liu et al., 2012) is referred to as LPI. All results are averaged over 100 random trials. All our experiments are implemented in Matlab 2012a, and run on a PC with Windows 7 operation system, Intel Core i7-3740QM (2.70GHz) CPU and 8GB memory.

3.6.1 Simulation

Our first simulation (Figure 3.1) shows that IWMV, its variant IWMV.log and the EM algorithm achieve the same final prediction accuracy, while IWMV has the lowest computational cost. Specifically, we vary the number of tasks N, and compare the final accuracy with one

[2] `http://www.ics.uci.edu/~qliu1/codes/crowd_tool.zip`

standard deviation error bar imposed (Figure 3.1(a)), the convergence time (Figure 3.1(b)), the number of iterations (i.e. steps) to converge (Figure 3.1(c)) and the total run time for 50 iterations (Figure 3.1(d)). With almost the same accuracies, IWMV converges faster and takes fewer steps to converge than EM and IWMV.log, and the run time of IWMV is prominently lower than EM. Similar conclusions can be also confirmed when changing M, q and L, thus we omit the similar results here.

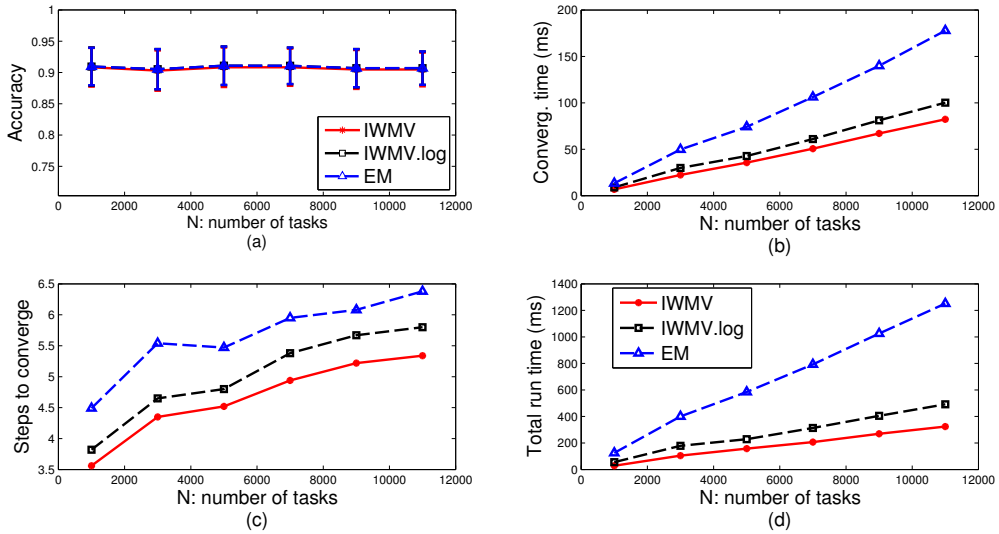
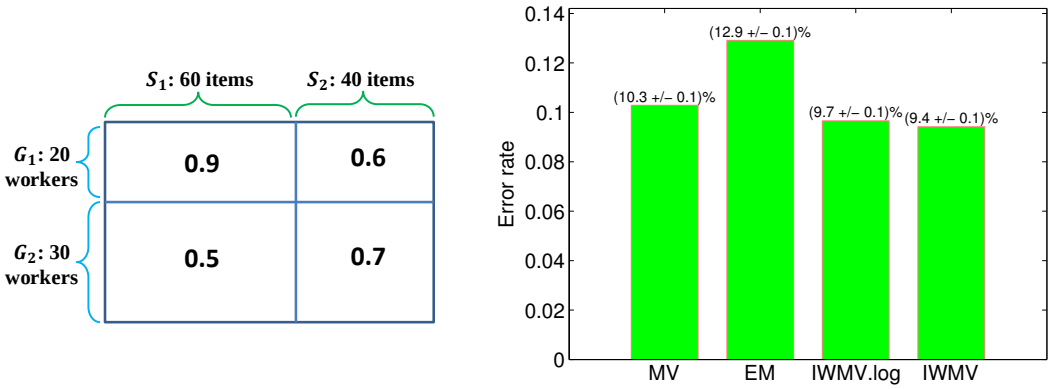


Figure 3.1: Comparison between IWMV, IWMV.log and the EM-MAP rule, with the number of items varying from 1000 to 11000. Simulation is performed with $L = 3, M = 31, q = 0.3$ under the Homogenous Dawid-Skene model. The reliabilities of workers are sampled based on $w_i \sim$Beta(2.3, 2), $i \in [M]$. All the results are averaged across 100 repetitions. (a) Final accuracy with error bar imposed. (b) The time until convergence, and we need to know the ground truth for measuring it. (c) Number of steps to converge. (d) Total run time is computed based on finishing 50 iterations.

The experiments above are strictly simulated based on the Homogenous Dawid-Skene worker model for its simplicity. To compare IWMV with EM when the worker model is violated, we simulate toy data. See Figure 3.2(a) for the setup: suppose there are two group of workers G_1 and G_2, and two sets of items S_1 and S_2. The true labels of items are generated uniformly from $\{\pm 1\}$. The data matrix is generated as follows: $\mathbb{P}(Z_{ij} = y_j) = 0.9$ if $i \in G_1, j \in S_1$; $\mathbb{P}(Z_{ij} = y_j) = 0.6$ if $i \in G_1, j \in S_2$; $\mathbb{P}(Z_{ij} = y_j) = 0.5$ if $i \in G_2, j \in S_1$; $\mathbb{P}(Z_{ij} = y_j) = 0.7$ if $i \in G_2, j \in S_2$. We use $q = 0.3$. The error rate (with one standard deviation) of MV, EM, IWMV.log and IWMV are shown in Figure 3.2 (b), which shows that IWMV achieves lower error rate than EM and IWMV.log do in this model misspecification example. The results in Figure 3.2 shows that IWMV are more robust than EM under



(a) (b)

Figure 3.2: (a) Setting of model misspecification. (b) Performance comparison under model misspecification setting in (a).

model misspecification to some extent. Similar results can be obtained under other different configurations (as in Figure 3.2), and we omit them here.

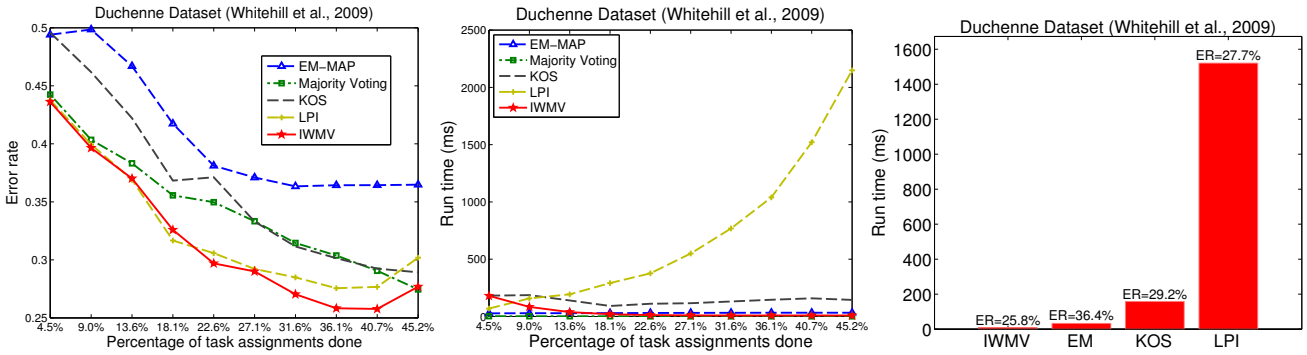
3.6.2 Real data

To compare our proposed iterative weighted majority voting with the state-of-the-art methods (Raykar et al., 2010, Karger et al., 2011a, Liu et al., 2012), we conducted several experiments on real data (most of them are publicly available). We sampled the collected labels in the real data independently with probability $\tilde{q}$ which varies from 0.1 to 1, and see how the error rate and run time change accordingly. For clarity of the figures produced in this section, we omit the results of IWMV.log since its performance is usually worse than IWMV in terms of both accuracy and computational time. Our focus will be the comparisons among IWMV, EM, KOS and LPI.

Table 3.1: The summary of datasets used in the real data experiments. $\bar{w}$ is the average worker accuracy.

Dataset	L classes	M workers	N items	#labels	$\bar{w}$
Duchenne	2	17	159	1221	65.0%
RTE	2	164	800	8000	83.7%
Temporal	2	76	462	4620	84.1%
Web search	5	177	2665	15539	37.1%

Duchenne dataset. The first dataset is from (Whitehill et al., 2009) on identifying Duchenne smile from non-Duchenne smile based on face images. In this data, 159 images are labeled with {*Duchenne, non-Duchenne*} labels by 17 different Mechanical Turk[3] workers. In total, there are 1,221 labels, thus 1221/(17×159)= 45.2% of the potential task assignments are done (i.e., 45.2% of the entries in the data matrix are observed). The ground truth labels are obtained from two certified experts and 58 out of the 159 images contain Duchenne smiles. The Duchenne images are hard to identify (the average accuracy of workers on this task is only 65%).



(a) (b) (c)

Figure 3.3: Duchenne smile dataset (Whitehill et al., 2009). (a) Error rate of different algorithms when the number of labels available increases. (b) Run time comparison. (c) A visualization with both run time and error rate when 40.7% of the task assignments are done.

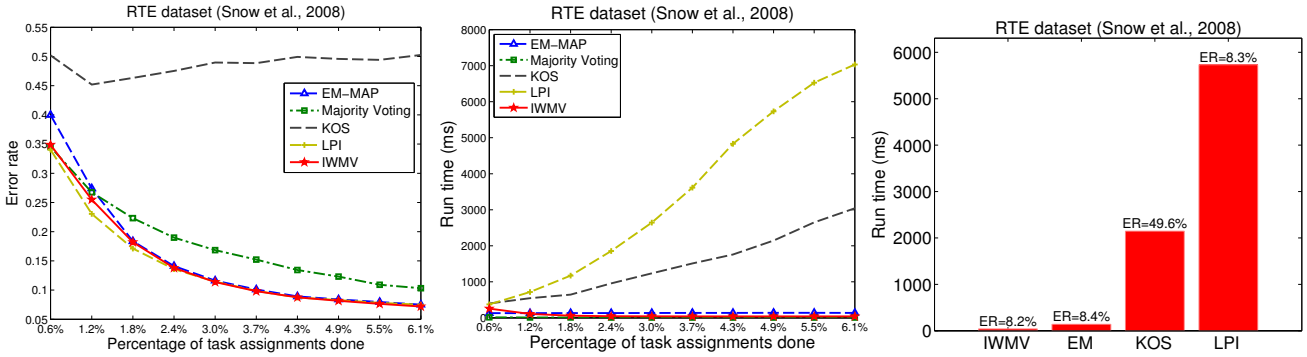
We conducted the experiments by sampling the labels independently with probability $\tilde{q}$ varying from 0.1 to 1. Thus the proportion of non-zero labels in the data matrix will vary from 0.6% to 45.2%. Note that based on our setting, the task assignment probability corresponding to $\tilde{q}$ is $q = \tilde{q} \times 45.2\%$. After sampling the data matrix from the original Duchenne dataset with a given $\tilde{q}$, we then run IWMV, majority voting, the EM algorithm (Raykar et al., 2010) , KOS (the iterative algorithm in (Karger et al., 2011a)), and LPI (the variational inference algorithm from (Liu et al., 2012)). The entire process will be repeated 100 times, and the results will be averaged. The comparison is shown in Figure 3.3(a).

From Figure 3.3(a), we can see that when the available labels are very few, the performance of IWMV is as good as LPI, and these two generally dominate the other algorithms. With more labels available, the error rate of IWMV is around 2% lower than LPI (Figure 3.3(a)). At the same time, we compared the run time of each algorithm (Figure 3.3(b)). With more labels, the run time of LPI increases fast (non-linearly), while the IWMV maintains a lower run time than EM, KOS and LPI. For a better visualization of comparing run

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

time and error rate, we compared the run time of IWMV, EM, KOS and LPI by a bar plot with their error rates imposed on top. Figure 3.3(c) shows the comparison when 40.7% of the labels are available ($\tilde{q} = 0.9$). IWMV is more than 100 times faster than LPI, and achieves the lowest error rate among these algorithms.

An interesting phenomenon in Figure 3.3(a) is that EM performs poorly — it is even worse than MV. The major reason for this is that the workers reliabilities form a pattern similar to our model misspecification example in Figure 3.2(a): some workers are good at a set of images but bad at the complementary set of images, while the other workers are reversed. This is a real-data example of model misspecification, and IWMV is more robust to the model misspecification on this data than EM.



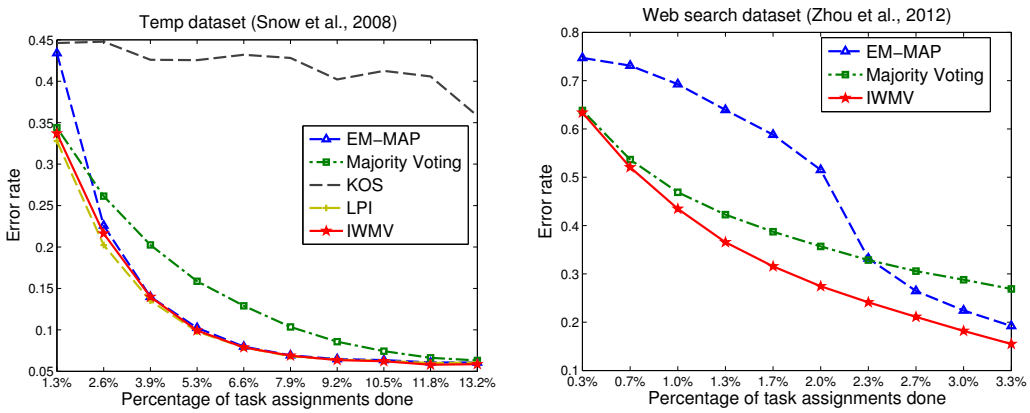
(a) (b) (c)

Figure 3.4: RTE dataset (Snow et al., 2008). (a) Error rate of different algorithms. (b) Run time when the percentage of the task assignments done increases. (c) Run time comparison when 4.9% of the task assignments is done. The error rates of each method are imposed on the top of the bar.

RTE dataset. The RTE data is a language processing dataset from (Snow et al., 2008). The dataset was collected by asking workers to perform recognizing textual entailment (RTE) tasks, i.e., for each question the worker is presented with two sentences and given a binary choice of whether the second sentence can be inferred from the first.

Temporal event dataset. This dataset is also a natural language processing dataset from (Snow et al., 2008). The task is to provide a label from {*strictly before, strictly after*} for event-pairs that represents the temporal relation between them.

We conducted similar experiments on the RTE dataset and the temporal event dataset as the one on the Duchenne dataset. The results on the RTE dataset are shown in Figure 3.4. Figure 3.4(a) is the performance curves of different algorithms when the percentage of task assignments done increases. Figure 3.4(b) is the run time of these algorithms, and it confirms the same observations as the results on the Duchenne dataset: the IWMV runs much faster than the other algorithms except majority voting, and it has similar performance



(a) (b)

Figure 3.5: (a) Results on Temp dataset from (Snow et al., 2008). (b) Results on Web search dataset (Zhou et al., 2012).

to LPI which is the state-of-art method (Figure 3.4(c)). For clarity, we show the performance comparison on temporal event dataset in Figure 3.5(a) and omit the run time comparison.

Web search dataset. In this dataset (Zhou et al., 2012), workers were asked to rate query-URL pairs on a relevance rating scale from 1 to 5. Each pair was labeled by around 6 workers, and around 3.3% of the entries in the data matrix is observed. The ground truth labels were collected via consensus from 9 experts. We treat the task as a multi-class labeling problem, thus $L = 5$. We conduct the experiment in a similar setting to the experiment on the Duchenne dataset — sampling the labels with probability $\tilde{q}$ and varied it to plot the performance curve (Figure 3.5(b)). Since LPI and KOS is constrained to binary labeling so far, we only compared IWMV with the EM-MAP rule and majority voting. The performance of IWMV generally outperforms the EM-MAP rule and majority voting by at least 4%.

3.7 Conclusions

In this chapter, based on a finite sample error rate bound on the mean error rate of weighted majority voting, a data-driven iterative weighted majority voting algorithm is proposed to efficiently infer the true labels of items. This algorithm approximates the oracle MAP with a theoretical guarantee on the error rate of its one-step version.

Through simulations under the Homogenous Dawid-Skene model (for simplicity) and tests on real data, we have the following findings.

1. The iterative weighted majority voting method (IWMV) performs as well as the EM-MAP rule with much lower computational cost in simulation, and IWMV is more robust to model-misspecification than EM.

2. On real data, IWMV achieved performance as good as or even better than that of the state-of-the-art methods with much less computational time.

For the future work, an appealing direction is to study the error rate of the IWMV algorithm after certain number of iterations. It will be of great of interest to the crowdsourcing community because the IWMV algorithm shares similarities with the famous EM algorithm, but differs from it with different steps on updating the weights of workers. Hence, any insightful results we get from studying the error rate of the IWMV algorithm will contribute directly to understand the behavior of the EM algorithm on crowdsourcing problems.

Chapter 4

Worker selection with a few control examples

4.1 Introduction

Besides studying inference algorithms in crowdsourcing as in Chapter 3, there is another important problem — how to choose good workers based on their performance on some initial test questions. In this chapter, we focus on the problem of selecting a small group of good workers under the constraint that we cannot hire more than K workers because of the limited budget.

Since the crowd workers often have different reliabilities due to their diverse backgrounds, it is important to weight their answers properly when aggregating their answers. A large body of work has been proposed to deal with the uncertainty and diversity on the workers' reliabilities; these methods often have a form of weighted majority voting where the answers of the majority of the workers are selected, with a weighting scheme that accounts the importance of the different workers according to their reliabilities. The workers' reliabilities can be estimated either using gold standard questions with known answers (e.g., Von Ahn et al., 2008, Liu et al., 2013), or by statistical methods such as Expectation-Maximization (EM) (see, e.g., Dawid and Skene, 1979, Whitehill et al., 2009, Karger et al., 2011a, Liu et al., 2012, Zhou et al., 2012).

The work in this chapter is motivated by a natural question: do more crowd workers necessarily yield better aggregated results than less workers? The idea of *wisdom of crowds* seems to suggest a confirmative answer, since "*larger crowds* should be *wiser*". From a Bayesian perspective, this would be true if we had perfect knowledge about the workers' prediction model, and we were able to use an oracle aggregation procedure that performs exact Bayesian inference. However, in practice, because the workers' prediction model and reliabilities are never known perfectly, we run the risk of adding noisy information as we increase the number of workers. In the extreme, there may exist a large number of "spammers", who submit completely random answers rather than good-faith attempts to label;

adding these spammers would certainly deteriorate the results, unless we are able to identify them perfectly, and assign them with zero-weights in the label aggregation algorithm. Even if there is no extreme spammers, the median-level workers may still decrease the overall accuracy if they dominate over the small number of high-quality workers. In fact, a recent empirical study (Mannes et al., 2014) shows that the aggregated results of a small number of (3 to 6) high-quality workers are often more accurate than those of much larger crowds.

In this chapter, we study this phenomenon by formulating a worker selection problem under a budget constraint. Assume we have a pool of workers whose reliabilities have been tested by a small number of gold standard questions; under certain label aggregation algorithm, we want to select a subset of workers that maximizes the accuracy, with a budget constraint that the number of workers assigned per task is no more than K. A naïve and commonly used procedure is to simply select the top K workers that have the highest reliabilities. However, due to the noisy nature of the label aggregation algorithms (e.g., majority voting or EM), selecting all the K workers does not necessarily give the best accuracy, and may cause a waste of the resource. We study this problem under a simple label aggregation algorithm based on weighted majority voting, and propose a worker selection method that is able to select fewer ($\leq K$) top-ranked workers, while achieve almost the same, or even better aggregated solutions than the naïve method that uses more (all the top K) workers.

Our method is derived by framing the problem into a combinatorial optimization problem that minimizes an upper bound of the error rate, and deriving a globally optimal algorithm that selects a group of top-ranked workers that optimize the upper bound of the error rate. We demonstrate the efficiency of our algorithm by comprehensive experiments on a number of real-world datasets.

There are many previous works on estimating the workers' reliabilities and eliminating the spammers based on a predefined threshold (Raykar and Yu, 2012, Joglekar et al., 2013). Our work instead focuses on selecting a minimum number of highest-ranked workers while discarding the others (which are not necessarily spammers). Note that our method has the advantage of requiring no pre-specified threshold parameters.

Our work is orthogonal with another line of research on online assignment for crowdsourcing (e.g., Chen et al., 2013, Ho et al., 2013, Tran-Thanh et al., 2013, Karger et al., 2014), which focus on dynamically route tasks to workers; this is distinguish with our work which focuses on an one-shot selection of the worker pool before distributing tasks to workers. Although online assignment generally allows more flexible and efficient budget allocation, they require much more implementation overhead in practice, especially considering the lack of good real time online assignment interfaces in most current crowdsourcing systems. Our approach is much easier (almost free) to implement in practice, and can be followed with a full online assignment process when implementation is possible.

The rest of the chapter is organized as follows. We introduce the background and the problem setting in Section 4.2. We then formulate the worker selection problem into a combinatorial optimization problem and derive our algorithm in Section 4.3. The numerical experiments are presented in Section 4.4. We give further discussions in Section 4.5 and conclude the chapter in Section 4.6.

4.2 Problem setup

Following the notations in Chapter 2, we assume there are M crowd workers and N items (or questions) each with labels from L classes. The set of workers is denoted by $\Omega = [M]$; the set of items is denoted by $[N]$ and the set of label classes by $[L]$, where we use $[M]$ to denote the set of first M integers. We assume each item j is associated with an unknown true label $y_j \in [L],\ j \in [N]$. We also assume that we have n *control* (or *gold standard*) questions whose true labels $y'_j \in [L],\ j \in [n]$ are known.

When item j is assigned to worker i for labeling, we get a possibly inaccuracy answer from the worker, which we denote by $Z_{ij} \in [L]$. The workers often have different expertise and attitude, and hence have different reliabilities. We assume i-th worker labels the items correctly with probability w_i, that is, $w_i = \mathbb{P}(Z_{ij} = y_j)$. In addition, assume we have an estimation of the workers' reliability $\hat{w}_i$, which can be estimated either based on the workers' performance on the control items, or by probabilistic inference algorithms like expectation-maximization (EM). With a known reliability estimation $\hat{w}_i$, most label aggregation algorithms, including the naïve majority voting and EM, can be written into a form of weighted majority voting,

$$\hat{y}_j = \underset{k \in [L]}{\operatorname{argmax}} \sum_{i \in S} f(\hat{w}_i) \cdot \mathrm{I}\left(Z_{ij} = k\right), \tag{4.1}$$

where $f(\hat{w}_i)$ is a monotonic weighting function that decides how much the answers of worker i contribute to the voting according to the reliability $\hat{w}_i$, and $\mathrm{I}\left(\cdot\right)$ is the indicate function. For majority voting, we have $f_{\mathrm{mv}}(\hat{w}_i) = 1$, which ignores the diversity of the workers and may performance badly in practice. In contrast, a log-odds weighting function $f_{\log}(\hat{w}_i) = \mathrm{logit}(\hat{w}_i) - \mathrm{logit}(1/L)$, where $\mathrm{logit}(\hat{w}_i) \stackrel{def}{=} \log(\frac{\hat{w}_i}{1-\hat{w}_i})$, can be derived using Bayesian rule under a simple model that assumes uniform error across classes; here $1/L$ is the probability of random guessing among L classes. However, in practice, the log-odds may be over confident (growing to infinite) when $\hat{w}_i$ is close to 1 or 0. A linearized version $f_{\mathrm{linear}}(\hat{w}_i) = \hat{w}_i - 1/L$ has better stability, and is simpler for theoretical analysis (Li et al., 2013b).

Note that both of $f_{\log}$ and f_{linear} have properties that are desirable for general weighting functions: Both are monotonic increasing functions of $\hat{w}_i$, and take zero value if $\hat{w}_i = 1/L$ (to exclude the labels from random guessers); they are both positive if $\hat{w}_i > 1/L$ (better than random guessers), and are both negative if $\hat{w}_i < 1/L$ (worse than random guessers). These common properties make f_{linear} and $f_{\log}$ work similarly in practice. But since f_{linear} is more stable and simpler for theoretical analysis, we will focus on the linear weighted function f_{linear} for our further development on the worker selection problem, that is, the labels are aggregated via (referred as *WMV-linear*),

$$\hat{y}_j = \underset{k \in [L]}{\operatorname{argmax}} \sum_{i \in S} (L\hat{w}_i - 1) \cdot \mathrm{I}\left(Z_{ij} = k\right). \tag{4.2}$$

In the next section, we study the worker selection problem and propose an efficient algorithm based on the analysis of the WMV-linear aggregation method.

4.3 Worker selection by combinatorial optimization

The problem of selecting an optimal set of workers requires predicting the error rate with a given worker set, which is unfortunately intractable in general. However, it is convenient to obtain an upper bound of the error rate for the linear weighted majority voting.

Theorem 4.1. *Given a set S of workers, using the weighted majority voting in (4.2) with linear weights f_{linear} and an unbiased estimator of the reliabilities $\{\hat{w}_i\}_{i\in S}$ that satisfies $\mathbb{E}[\hat{w}_i] = w_i$. If the workers' labels are generated independently according the following probability*

$$\mathbb{P}(Z_{ij} = l | y_j = k) = \begin{cases} w_i & \text{if } \ l = k, \\ \frac{1-w_i}{L-1} & \text{if } \ l \neq k. \end{cases} \tag{4.3}$$

Then we have

$$\frac{1}{N}\sum_{j=1}^{N} \mathbb{P}(\hat{y}_j \neq y_j) \leq \exp\left[-\frac{2F(S)^2}{L^2(L-1)^2} + \ln(L-1)\right], \tag{4.4}$$

$$\text{where} \qquad F(S) = \frac{1}{\sqrt{|S|}}\sum_{i\in S}(Lw_i - 1)^2. \tag{4.5}$$

Remarks:

(1) Note that the above upper bound depends on the worker set S and their reliabilities w_i only through the term $F(S)$. In fact, according to the proof in the supplementary, the term $F(S)$ corresponds to the expected gap between the voting score of the true label y_i (i.e. $\sum_{i\in S} f_{\text{linear}}(\hat{w}_i)\mathrm{I}\left(Z_{ij} = y_i\right)$) and that of the wrong labels, and hence reflects the confidence of the weighted majority voting. Therefore, $F(S)$ represents a score function for the worker set S: if $F(S)$ is large, the weighted majority voting is more likely to give correct prediction.

(2) The assumption (4.3) used in Theorem 4.1 implies a "one-coin" model on the workers labels, where the labels are correct with probability w_i, and otherwise make mistakes uniformly among the remaining classes. This is a common assumption to make, especially in theoretical works (see e.g., (Karger et al., 2011a, Ghosh et al., 2011, Joglekar et al., 2013)). It is possible to relax (4.3) to a more general "two-coin" model with arbitrary probability $\mathbb{P}(Z_{ij} = l|y_j = k)$, which, however, may lead more complex upper bounds. In our empirical study on various real-world datasets, we find that $F(S)$ remains to be an efficient score function for worker section even when the one-coin assumption does not seem to hold.

Based on (4.5), it is natural to select the workers by maximizing the term $F(S)$, that is,

$$\underset{S\subseteq\Omega}{\operatorname{argmax}} F(S), \qquad s.t. \quad |S| \leq K, \tag{4.6}$$

Unfortunately, $F(S)$ depends on the workers' true reliabilities w_i, which is often unknown. We instead estimate $F(S)$ based on $\hat{w}_i$. The following theorem provides an unbiased estimator.

Lemma 4.2. *Assume $\hat{w}_i$ is an unbiased estimator of w_i that satisfies $\mathbb{E}[\hat{w}_i] = w_i$, and $\hat{\text{var}}(\hat{w}_i)$ is an unbiased estimator of the variance of $\hat{w}_i$. Consider*

$$\hat{F}(S) = \frac{1}{\sqrt{|S|}} \sum_{i \in S} G(\hat{w}_i), \tag{4.7}$$

where

$$G(\hat{w}_i) = (L\hat{w}_i - 1)^2 - L^2 \hat{\text{var}}(\hat{w}_i), \tag{4.8}$$

then $\hat{F}(S)$ is an unbiased estimate of $F(S)$.

Remarks:

(1) The first term $(L\hat{w}_i - 1)^2$ in (4.8) shows that the workers with $\hat{w}_i$ close to either 1 or 0 should be encouraged; these workers tend to answer the questions either all correctly or all wrongly, and hence are "strongly informative" in terms of the predicting the true labels. Note that these workers with $\hat{w}_i = 0$ are strongly informative in that they eliminate one possible value (their answer) for the true labels. On the other side, more workers also means more noise, so there is a term $\sqrt{|S|}$ for balancing the signal-noise ratio — to encourage hiring "strong" workers instead of only hiring more workers.

(2) A simpler estimation of $F(S)$ is to directly plug $\hat{w}_i$ as w_i into (4.5), that is,

$$\hat{F}_{\text{plug}}(S) = \frac{1}{\sqrt{|S|}} \sum_{i \in S} (L\hat{w}_i - 1)^2. \tag{4.9}$$

However, this obviously leads to a biased estimator of $F(S)$ because of the missing of the variance term in (4.8). The existence of the variance term is of critical importance: The workers with large uncertainty on the reliabilities should be less favorable compared with these with a more confident estimation.

Since Lemma 4.2 does not specify $\hat{w}_i$ and $\hat{\text{var}}(\hat{w}_i)$, the next theorem provides a concrete example of $\hat{F}(S)$, based on which a symmetric confidence interval of $F(S)$ can be constructed.

Theorem 4.3. *Assume a group of workers are tested with n control questions, and let c_i be the number of correct answers given by worker i on the n control questions. Then an unbiased estimator $\hat{w}_i$, with an unbiased estimator of var$(\hat{w}_i)$ can be obtained by*

$$\hat{w}_i = \frac{c_i}{n} \quad \textit{and} \quad \hat{\text{var}}(\hat{w}_i) = \frac{c_i(n - c_i)}{n^2(n-1)}. \tag{4.10}$$

With such $\hat{w}_i$ and $\hat{\text{var}}(\hat{w}_i)$, the corresponding $\hat{F}(S)$ in (4.7) is unbiased and the interval $[\hat{F}(S) - \frac{n(L-1)^2}{n-1}\alpha, \hat{F}(S) + \frac{n(L-1)^2}{n-1}\alpha]$ covers $F(S)$ with probability at least $1 - 2e^{-2\alpha^2}$ for any $\alpha > 0$.

Remark: A discussion about the advantage of the unbiasness of $\hat{F}(S)$ and the symmetric confidence interval is deferred to Section 4.5.

Based on the estimation of $F(S)$ in Theorem 4.3, the optimization problem is rewritten into

$$\underset{S\subseteq\Omega}{\operatorname{argmax}}\,\hat{F}(S), \qquad s.t. \quad |S| \leq K, \tag{4.11}$$

where $\hat{F}(S)$ is defined in (4.7). Although this combinatorial problem is neither sub-modular nor super-modular, we show it can be exactly solved with a linearithmic time algorithm shown in Algorithm 3.

Algorithm 3 Worker selection algorithm

1: **Input:** Worker pool $\Omega = \{1, 2, \ldots, M\}$ and estimated reliabilities $\{\hat{w}_i\}_{i\in\Omega}$ from n control questions; Number of label classes L; Cardinality constraint: no more than K workers per item.

2: $x_i \leftarrow G(\hat{w}_i)$, $\forall i \in \Omega$ as in (4.8), and sort $\{x_i\}_{i\in\Omega}$ in descending order so that $x_{\sigma(1)} \geq x_{\sigma(2)} \geq \ldots \geq x_{\sigma(M)}$, where σ is a permutation of $\{1, 2, \cdots, M\}$.

3: $B \leftarrow \min(K, M)$, $\quad g_1 \leftarrow x_{\sigma(1)} \quad$ and $\quad F_1 \leftarrow g_1$.

4: **for** k from 2 to B **do**

5: $\quad g_k \leftarrow g_{k-1} + x_{\sigma(k)} \quad$ and $\quad F_k \leftarrow \dfrac{g_k}{\sqrt{k}}$.

6: **end for**

7: $k^* \leftarrow \min\left\{\underset{1\leq k\leq B}{\operatorname{argmax}} F_k\right\}$.

8: **Output:** The selected subset of workers $S^\star \leftarrow \{\sigma(1), \sigma(2), \cdots, \sigma(k^*)\}$.

Algorithm 3 progresses by ranking the workers according to $G(\hat{w}_i)$ in a decreasing order, and sequentially evaluates the groups of the top-ranked workers, and then finds the smallest group that has the maximal score $\hat{F}(S)$. The time complexity of Algorithm 3 is $O(|\Omega| \log |\Omega|)$ and the space complexity is $O(|\Omega|)$, where Ω is the whole set of workers.

The following theorem shows that Algorithm 3 achieves the global optimality of (4.11).

Theorem 4.4. *For any fixed $\{\hat{w}_i\}_{i\in\Omega}$. The set $S^\star$ given by Algorithm 3 is a global optimum of Problem (4.11), that is, we have $\hat{F}(S^\star) \geq \hat{F}(S)$ for $\forall S \in \Omega$ that satisfies $|S| \leq K$.*

Remark: As a generalization, consider the following multiple-objective optimization problem,

$$\underset{S\subseteq\Omega}{\operatorname{argmax}}(\hat{F}(S), \; -|S|), \qquad s.t. \quad |S| \leq K,$$

which simultaneously maximizes the score $\hat{F}(S)$ and minimizes the number $|S|$ of workers actually deployed. We can show that $S^\star$ is in fact a Pareto optimal solution in the sense that there exist no other feasible S that improves over $\hat{F}(S)$ in terms of both $\hat{F}(S)$ and $|S|$ (details in supplementary).

4.4 Experimental results

We demonstrate our algorithm using empirical experiments based on both simulated and real-world datasets. The empirical results confirm our intuition: Selecting a small number of top-ranked workers may perform as good as, or even better than using all the available workers. In particular, we show that our worker selection algorithm significantly outperforms the naive procedure that uses the entire top K workers. We find that our algorithm tends to select a very small number of workers (less than 10 in all our experiments), which is very close to the optimal number of the top-ranked workers in practice.

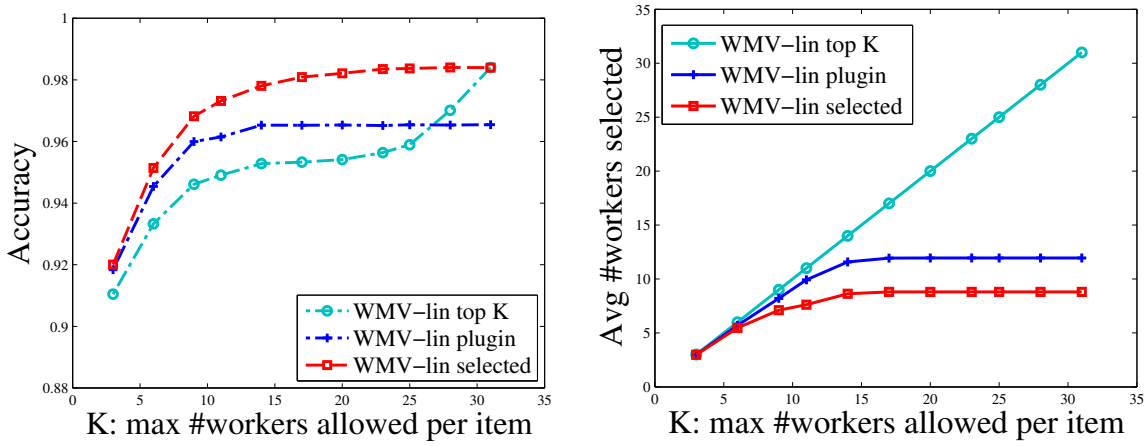


Figure 4.1: Performance of different worker selection methods on simulated data. WMV-linear aggregation is used in all the cases. We simulated 31 workers and 1000 items with binary labels, and use 10 control questions. The workers' reliabilities are drawn independently from $Beta(2.3, 2)$. (a) The accuracies when the budget K varies. (b) The actual number of workers used by different worker selection methods when K increases.

To be specific, we consider the following practical scenario in the experiments: (i) Assume there is a worker pool Ω where each worker has completed a "qualify exam" with n control questions, which is required by either the platform or a particular task owner. (ii) The task owner selects a subset of workers from Ω using a worker selection algorithm such as Algorithm 3 based on their performance on the qualify exam. (iii) The selected workers are distributed to answer the N questions of the main interest. (iv) Label aggregation algorithms such as WMV-linear or EM are applied to predict the final labels of these N items.

Even though our worker selection algorithm is derived when using WMV-linear, we can still use other label aggregation algorithms such as EM, once the worker set is selected. This gives the following possible combinations of the algorithms that we test: WMV-linear on the top K workers (`WMV top K`), WMV-linear on the worker set $S^\star$ selected by Algo-

rithm 3 (`WMV-lin selected`), and WMV with log ratio weights on the selected worker set $S^\star$ (`WMV-log selected`), the EM algorithm on randomly selected K workers (referred as `EM random K`), EM on the top K workers ranked (`EM top K`) and EM on the worker set $S^\star$ selected (`EM selected`). We also implement the worker selection algorithm based on the plugin estimator in (4.9) (which is the same as Algorithm 3, except replacing $G(\hat{w}_i)$ with $(L\hat{w}_i - 1)^2$), followed with a WMV-linear aggregation algorithm (referred as `WMV-lin plugin`). Since the majority voting tends to perform much worse all the other algorithms, we omit it in the plots for clarity.

In each trial of the algorithms on both the simulated and real-world datasets, 10 items are randomly picked from the collected data as the control items, and the workers' reliabilities $\{\hat{w}_i\}_{i\in\Omega}$ are estimated based on the accuracy on the control items as (4.10). In each trial, the number of workers selected by Algorithm 3 was stored and the average number of workers was computed for each budget K. We terminate all the iterative algorithms at a maximum of 100 iterations. All results are averaged over 100 random trials.

4.4.1 Simulated data

We generate the simulated data by drawing 31 workers with reliability w_i from $Beta(2.3, 2)$, and we randomly generated 1000 items with true labels uniformly distributed on $\{\pm 1\}$. The budget K varies from 3 to 31. Figure 4.1(a) shows the accuracy of WMV-linear with different worker selection strategies as the budget K changes. We can see that `WMV-lin selected` dominates the other methods. Figure 4.1(b) shows the actual number of workers selected by worker selection algorithm (Algorithm3). WMV-linear based on our selected workers uses a relatively small number of (always < 10) workers (the red curve in Figure 4.1(b)), and achieve even better performance than `WMV-lin top K` that uses the entire available budge (the blue line in Figure 4.1(b)). We find that the worker selection algorithm based on the plugin estimator $\hat{F}_{\text{plug}}(S)$ tends to select slightly more workers, but achieves slightly worse performance than Algorithm 3 based on the bias-corrected estimator $\hat{F}(S)$ (see `WMV-lin select` vs. `WMV-lin plugin` in Figure 4.1(a)). This implies the importance of the variance term in (4.8), which penalizes the workers with noisy reliability estimation.

The number of control questions n controls the variance of the reliability estimation $\hat{w}_i$, and hence influences the results of the worker selection algorithms. Figure 4.2(a) shows the results when we vary n from 3 to 45, with the budget fixed at $K = 20$. We see that the performance of all the algorithms increases when n increases, because we know more accurate information about the workers' true reliabilities, and can make better decision on both choosing the top K workers and selecting workers by Algorithm3. In addition, when n increases, the variance of $\hat{w}_i$ decreases and the difference between `WMV-linear selected` and `WMV-linear plugin` decreases.

Figure 4.2(b) shows the results when when we vary the prior parameter a where $w_i \sim Beta(a, 2)$, fixed $K = 20$ and $n = 10$. Larger a means the workers are more likely to have high reliabilities (i.e., close to 1). We see from Figure 4.2(b) that `WMV-lin top K` increases as a increases, due to the overall improvement of the reliabilities of the top K workers. The

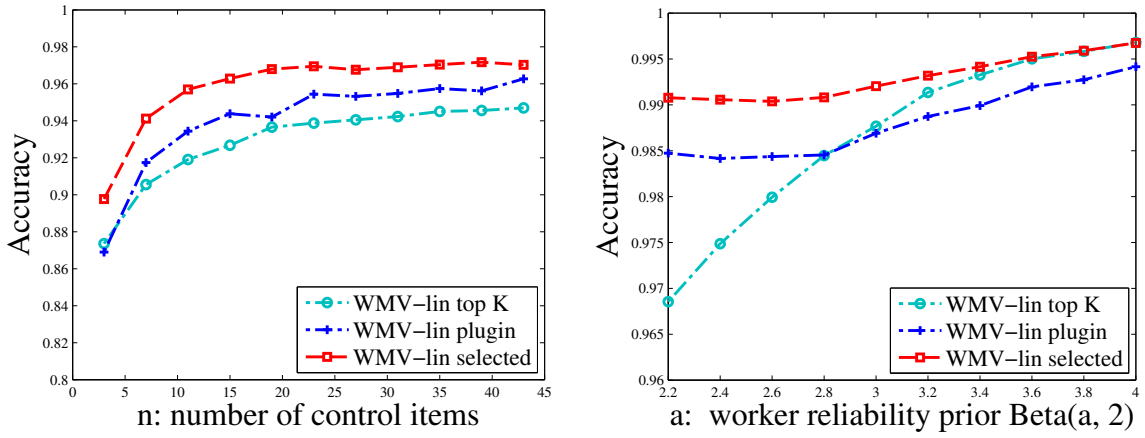


Figure 4.2: Performance of different worker selection methods, (a) when changing the number of control questions n, and (b) when changing the parameter a in the reliability prior $Beta(a, 2)$. The budget K is fixed at 20. We use the WMV-linear aggregation method in all the cases.

performances of `WMV-lin selected` and `WMV-lin plugin` improves only slightly, probably because they only select several top workers which is not heavily affected by a.

4.4.2 Real data

We test the different worker selection methods on three real-world datasets: two collected by ourselves from the crowdsourcing platform Clickworkers [1], and one by Welinder et al. (2010) from Amazon Mechanical Turk.

Crowd-test dataset: In this dataset, 31 workers are asked to answer 75 knowledge-based questions from *allthetests.com,* which cover topics such as science, math, common knowledge, sports, geography, U.S. history and politics and India. All these questions have 4 options, and we know the all the ground truth beforehand. We required each worker to finish all the questions. A typical example of the knowledge-test question is as follows:

(Question): In what year was the Internet created?

(Options): A. 1951; B. 1969; C. 1985; D. 1993.

Figure 4.3(a) shows the performance of the different methods as the budget K changes. Since EM is widely used in practice, we include the results when using it as the label aggregation algorithm after the workers are selected. We find that the performance of `EM Top K` first increases when K is small and then decreases when K is large enough (≥ 10 in

[1] *http://www.clickworker.com/en*

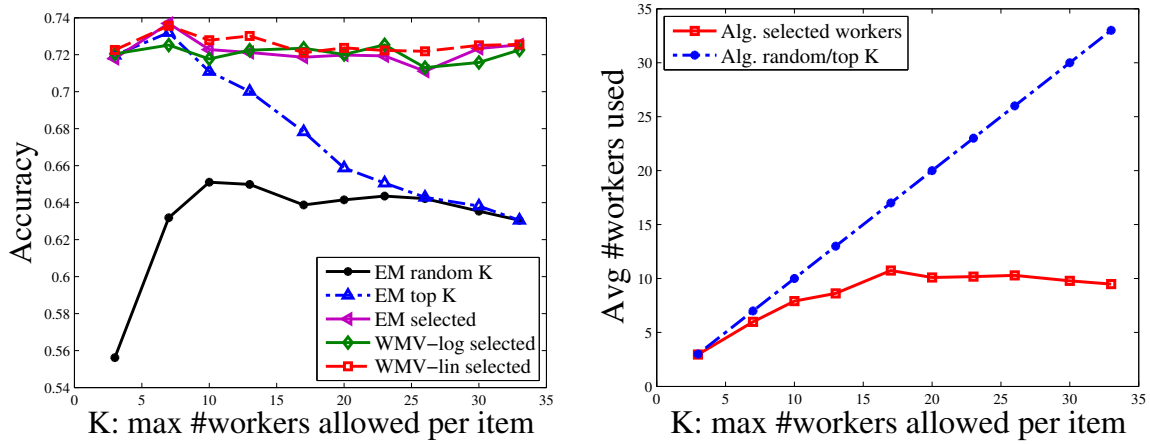


Figure 4.3: Crowd-test data. 10 items were randomly selected as control. (a) Performance curve of algorithms with K increasing. (b) Number of workers the algorithms actually used for each K.

this case). Our worker selection algorithm selects much smaller number of workers, while much better performance, compared to the `top K` and `random` selection methods.

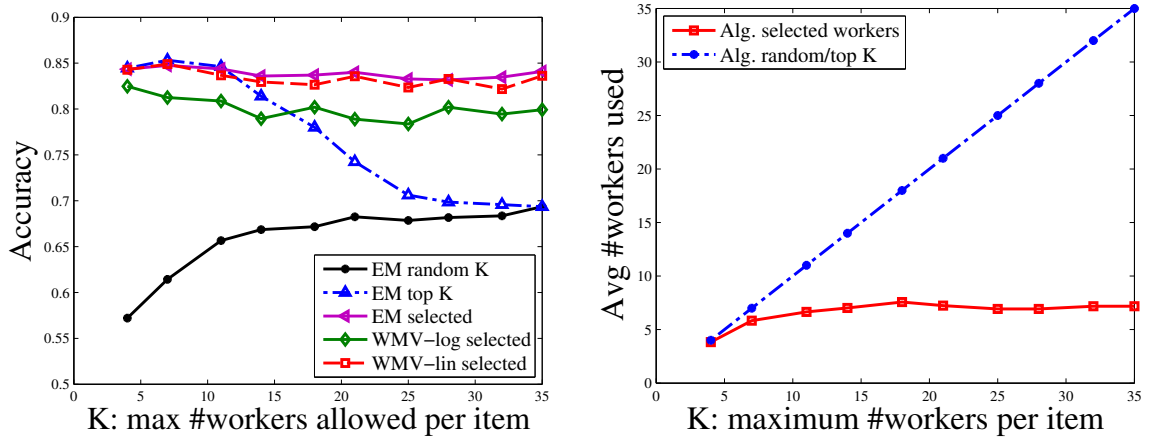
Disambiguation dataset: The task here is to identify which Wikipedia page (within 4 possible options) a given highlighted entity in a sentence actually refers to. We collected 50 such questions in the technology domain with ground truth available, and hire 35 workers through Clickworkers, each of which is required to complete all the questions. A typical example is as follows:

(Question): "The Microsoft .Net Framework 4 redistributable package install the .NET Framework ***runtime*** *and associated files that are required to run and develop applications to target the .NET Framework 4". Which Wiki page does "runtime" refer to?*
(Options):

A. http://en.wikipedia.org/wiki/Run-time_system
B. http://en.wikipedia.org/wiki/Runtime_library
C. http://en.wikipedia.org/wiki/Run_time_(program_lifecycle_phase)
D. http://en.wikipedia.org/wiki/Run_Time_ Infrastructure_(simulation)

Bluebird dataset: It is collected by Welinder et al. (2010) and is publicly available. In this dataset, 39 workers are asked if a presented image contains Indigo Bunting or Blue GroBeak. There are 108 images in total.

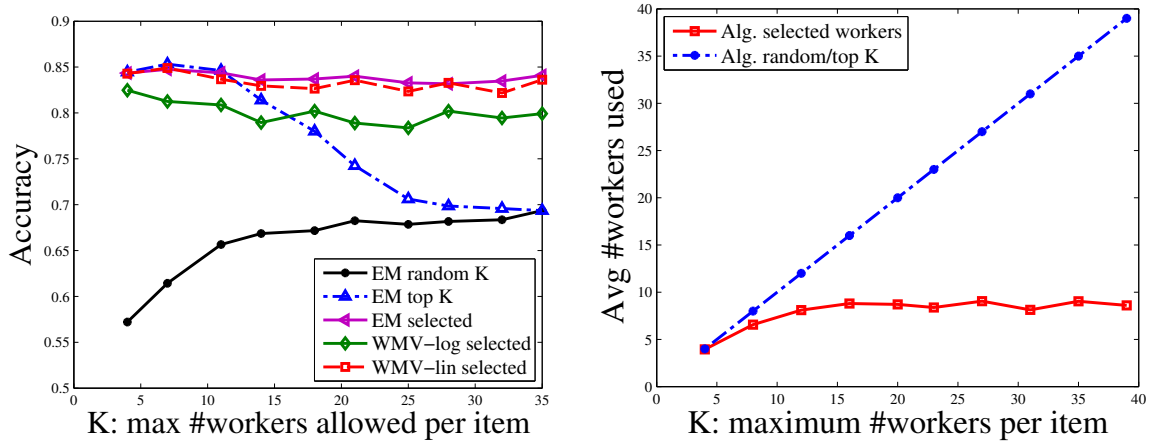
Figure 4.4 and Figure 4.5 show the performance of the different algorithms on the bluebird and the disambiguation dataset, respectively. The results are similar to the one in Figure 4.3. For the disambiguation dataset, the number of workers selected is usually no more than



(a) (b)

Figure 4.4: The disambiguation dataset: 35 workers and 50 questions in total. The settings are the same as that of Figure 4.3. (a) Performance curve of algorithms with K increasing. (b) Number of workers the algorithms actually used for each K.

6, and the corresponding number for bluebird dataset is 9.



(a) (b)

Figure 4.5: The bluebird dataset: 39 workers and 108 questions in total. The settings are the same as that of Figure 4.3. (a) Performance curve of algorithms with K increasing. (b) Number of workers the algorithms actually used for each K.

Note that `WMV-lin selected`, `WMV-log selected` and `EM selected` are based on the

workers selected by Algorithm 3. They achieve better performance than EM based on the top K or the random selected workers when K is large. This shows that aggregation based on inputs from selected workers not only saves budget but also maintains good performance.

4.5 Discussion

What is the advantage of ensuring that $\hat{F}(S)$ is an unbiased estimate of $F(S)$? The true objective function $F(S)$ is unknown, and we can only optimize over a random estimation $\hat{F}(S)$. If $\hat{F}(S)$ is a biased estimator and the bias depends on $\{\hat{w}_i\}_{i\in\Omega}$, then the optimum solution may be very different from the underlying true solution. With the unbiased estimator and the symmetric confidence interval gurantee shown in Lemma 4.2 and 4.3, optimizing $\hat{F}(S)$ is equivalent to optimizing a proper confidence bound, because the margin in the confidence interval often does not depend on the workers' reliabilities. The results in Figure 4.1 confirm that with the unbiased estimator $\hat{F}(S)$, the performance of WMV on the selected workers is better than that with the biased plugin estimator $\hat{F}_{\text{plug}}(S)$.

Why does WMV-linear perform better than WMV with $f_{\log}$? In some of our empirical results (e.g., Figure 4.4), we find that WMV with log ratio weight is not as good as the one with the linear weight. It is mainly because there is a high chance that some workers get estimated reliability $\hat{w}$ close to 0 or 1 when the number of control questions is small (e.g., $n = 10$). Even if we do truncation to prevent a weight $f_{\log}(\hat{w}_i)$ from going to ∞, the large weights of some workers may still lead to unstable aggregations. However, the performance of WMV with $f_{\log}$ improves when we use larger n or heavier truncation on $\hat{w}_i$.

Why does EM with top-K workers perform poorly as K increases? Within the given pool of workers, we add increasingly less reliable workers (compared with the workers already selected) as K increases; these less reliable workers may confuse the EM algorithm, causing worse reliability estimation as well as final prediction accuracy. This intuition matches with our empirical results in Figure 4.3, Figure 4.4 and Figure 4.5: the performance of EM generally first increases when K is small (with increasingly more top-quality workers), but then decreases when K is large (as more less reliable workers are added).

4.6 Conclusions

In this chapter, we studied the problem of selecting a set of crowd workers to achieve the best accuracy for crowdsourcing labeling tasks. We demonstrated that our worker selection algorithm can simultaneously minimize the number of selected workers and minimize the prediction error rate, achieving the best in terms of both cost and efficiency. For future directions, we are interested in developing better selection algorithms based on more advanced label aggregation algorithms such as EM, or more complex probabilistic models.

Chapter 5

Crowd Targeting: discovering and targeting the right group of workers

5.1 Motivation

The worker selection problem in Chapter 4 assumes that the selected group of workers from the crowd who take the test are willing to perform further tasks (i.e., the selected group will be sure to perform the assigned tasks in the future). Therefore, it is a pure subgroup selection problem. However, this assumption holds only when crowdsourcing platforms support real time decision of hiring workers (a task owner makes real time selection and then hire the selected group of workers immediately to perform the future tasks). Most of the current crowdsourcing platforms do not support that so far, but it is an important feature that might be supported in the near future. On the contrary, the work in this chapter overcomes the issue above by discovering that good workers can be identified with certain characteristic information. Therefore, even if we cannot hire the same selected people for future tasks, we can hire the workers who have similar backgrounds with them. This ensures that the strategy in this chapter be practical to all the current crowdsourcing platforms and can be directly deployed if necessary.

To aggregate the responses from multiple workers, the simplest approach would be to treat the response from each worker equally and take the majority. This majority voting approach could be reasonably effective but it is very often sub-optimal. Intuitively, some workers are better than others and those higher quality workers are not necessarily the majority, as philosopher Soren Kierkegaard once said, "Truth always rests with the minority, and the minority is always stronger than the majority, because the minority is generally formed by those who really have an opinion". Soren's claim might be too strong, but it is true in many cases.

Therefore, how to find the "minority" in the crowd that truly have wisdom is a very important problem. There are quite a few previous works that try to solve this problem (Dawid and Skene, 1979, Karger et al., 2011a, Liu et al., 2012, Raykar et al., 2010, Zhou

et al., 2012). The general approach is that the quality of each worker and the true answer of each task are modeled as latent variables that depend on each other, and the values of these variables can be jointly estimated by optimizing certain objective functions. It is shown those methods can almost always outperform simple majority voting for deciding the final output.

In practice, previous worker quality estimation approaches still have some limitations. First, those approaches mainly utilize signals from the distribution of workers' responses, but in practice, there could be many factors that are related to worker quality. Among those factors, various characteristics of the worker, such as demographics information and educational background, can be very indicative of worker quality on some tasks, as shown in some recent empirical studies (Ipeirotis, 2010b, Kazai et al., 2012, 2013). Intuitively, for a given task, if some characteristics of workers are indeed related to their quality, they should be taken into worker quality estimation. On the other hand, it is also possible that none of the available worker attributes are related to quality in some tasks, and in such cases, they should not play a role in quality estimation.

The second limitation of previous approaches is that they can only estimate the quality of each worker after he/she performs the task, which means it is difficult to improve budget efficiency for future tasks of the same kind. There are still some ways, for example, we can certainly identify spammers and refuse to pay them and even ban them from doing the task in the future. However, for identified high quality workers, it could be difficult to reach them and have them do the task when needed, since they may not be available or willing do the task with the same price.

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

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

To solve this problem, in this chapter, we propose a general crowd targeting framework, that can automatically discover if any specific groups of workers based on their characteristics have higher quality on average for a given task, and target those groups, if they exist, for future work of the same task to improve data quality and budget efficiency. The general idea is to send a small part of the task to the entire crowd at the beginning, and learn the selection criteria from the data and use those criteria to target workers for the remaining and future work. Since the targeting criteria are automatically learned from the data, its

effectiveness is more stable. Note that the definition of worker characteristics can be very general, e.g., workers' available profile information, or their responses to any questions we put in the task.

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

In the rest of this chapter, Section 5.2 introduces the crowd targeting framework. Section 5.3 presents notations and setup for the crowd targeting problem, and derives a new worker effect measure reflecting the quality of a worker. Section 5.3 discusses how we measure the quality of workers. Section 5.4 focuses on the details of two algorithms for implementing the crowd-targeting framework. Section 5.5 contains experimental results on two real datasets that we have collected and a publicly available dataset to verify the framework, and Section 5.6 discusses the most related work to this chapter.

5.2 Crowd Targeting framework

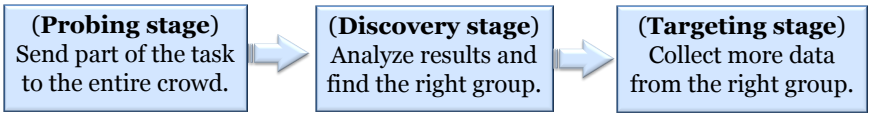


Figure 5.1: The crowd targeting framework.

In this section, we describe our crowd targeting framework in details. Figure 5.1 illustrates the three main stages of the framework: probing stage, discovery stage and the targeting stage.

5.2.1 Probing stage

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

In the meantime, we will need to gather characteristics of workers at this stage. On some crowdsourcing platforms, basic information is available in workers' profiles, such as gender and location. Otherwise, we can do the same as in (Kazai et al., 2012, 2013) and simply add survey questions as part of the task. We can require workers to answer these survey

questions by utilizing the screening mechanism provided by most crowdsourcing platforms or some other techniques like JavaScript.

The notion of worker characteristics is very general. The default questions would be demographics (major, education level, age, language, etc.) or questions that capture personality traits (Kazai et al., 2012, 2013). And task owners can customize and choose any questions that they think might be relevant except for those that violate laws or invade privacy of workers, such as asking for their social security numbers.

Figure 5.2 illustrates part of our design for collecting worker characteristic information. Note that we assure the workers that we protect their privacy and make clear that we will not identify any individual worker based on the information.

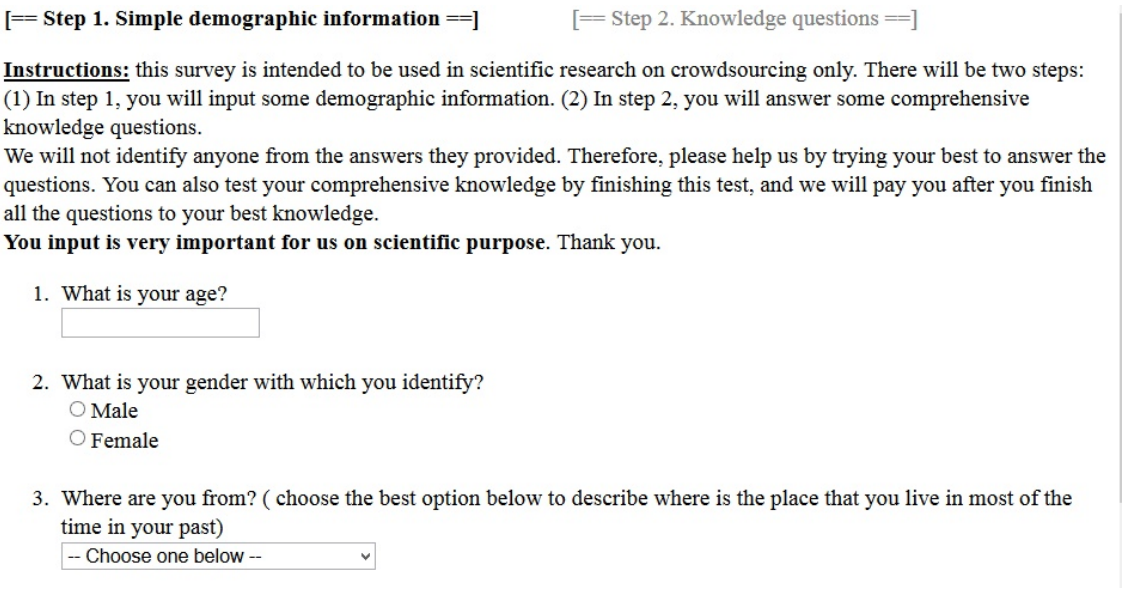


Figure 5.2: A screen shot of part of the design for collecting worker information.

5.2.2 Discovery stage

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

Quality is the most important criteria for worker group discovery: the target groups have to be better than other workers based on some measures. There are some standard measures that can be used to judge how good or useful a worker is, and in our framework we propose a new measure that can perform better in practice. Further discussions on measures are in Section 5.3.

Availability or the population ratio of the target groups is another important criterion for discovery. We cannot target groups which are too rare in the crowd such that we have

a small chance of immediately getting new workers from these groups for future tasks. This is even more important for tasks that are preferred to be finished within a short time. We are utilizing a population threshold in the group discovery algorithms, which can be tuned by task owners based on the urgency of their tasks.

5.2.3 Targeting stage

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

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

5.3 Worker effects

Before analyzing the association between worker characteristics and reliability, we first need to formally define some measures for worker reliability, which can be treated as the *effect* on workers caused by certain factors (their characteristics). We will first setup the crowdsourcing scenario we focus on in this chapter, and then introduce three effects that can be used in the analysis.

5.3.1 Setup of the crowdsourcing scenario

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

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

We use a succinct model (Karger et al., 2011a, Raykar et al., 2010) to model the relation between worker reliability and true answers by assuming that worker i is characterized by

accuracy $w_i \in [0, 1]$, i.e. how often her answer is correct. And when she makes mistakes, her answer is uniformly distributed among all the wrong answers. Formally,

$$\mathbb{P}(Z_{ij} = k | y_j = k) = w_i, \quad \forall k \in [L] \tag{5.1}$$

$$\mathbb{P}(Z_{ij} = l | y_j = k) = \frac{1 - w_i}{L - 1}, \forall l, k \in [L], l \neq k. \tag{5.2}$$

The posterior probability for the k-th answer to be true in task j given the collected answers, i.e. $\mathbb{P}(y_j = k | Z, \{w_i\}_{i=1}^M)$, and the accuracy of each worker i, i.e. w_i, are unknown at the beginning, but they can be estimated iteratively by the EM algorithm (Dempster et al., 1977, Raykar et al., 2010). In literature, this method is referred to as the maximum likelihood method on the Class-Conditional Dawid-Skene model or one-coin model (Karger et al., 2011a, Raykar et al., 2010, Li et al., 2013a,b), which is a special case of the full Dawid-Skene model (Dawid and Skene, 1979). We use this method due to its simplicity and effectiveness in many scenarios, and our crowd targeting framework can be easily generalized when other methods are applied for estimating worker accuracy and predicting true answers.

5.3.2 Effects

Now, we will introduce several measures for worker reliability that can be treated as *effects* on the worker. These measures should effectively reflect how good or how useful a worker's answers are for the final prediction of true answers. So intuitively, the measure should be a function of the worker's accuracy. Formally the effect of the i-th worker is $\tau_i = f(w_i)$.

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

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

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

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

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

Based on the analysis of the EM algorithm on Dawid-Skene models (Li et al., 2013a,b), we can know that if we can correctly estimate each worker's accuracy w_i, then we will give higher positive weights to answers provided by more accurate workers; on the other hand, for workers who have accuracy worse than random guessing, we can effectively give their answers negative weights, which can help filter out potential wrong answers. In another word, a less accurate worker, if her accuracy can be effectively estimated, could still make positive contribution for the final prediction.

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

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

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

where *constant* is indepdent of w_i.

So the data likelihood contributed by worker i is essentially:

$$C_i = S_i \ln w_i + (n_i - S_i)\ln\frac{1-w_i}{L-1}, \tag{5.3}$$

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

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

$$\begin{aligned} \hat{C}_i &= n_i\left(\hat{w}_i\ln\hat{w}_i + (1-\hat{w}_i)\ln\frac{1-\hat{w}_i}{L-1}\right) \\ &= -n_i\cdot\text{Entropy}\left(\left\{\hat{w}_i, \frac{1-\hat{w}_i}{L-1}, \cdots, \frac{1-\hat{w}_i}{L-1}\right\}\right). \end{aligned} \tag{5.4}$$

Notice that if $\hat{w}_i$ can freely take values from $[0,1]$, then $\hat{C}_i$ is maximized when $\hat{w}_i = 1$, i.e. worker i is a perfect worker; and $\hat{C}_i$ is minimized when $\hat{w}_i = 1/L$, which is equivalent to the accuracy of a worker who randomly guesses the answer. This reveals the fact that if EM is used for finding the true answer, a random guesser is the most helpless worker comparing with workers with higher or even lower accuracy. Note that $\ln L$ is the entropy of the answer distribution of a random guesser. Therefore, (5.4) is equivalent to the *information gain* from the answers of worker i compared to a random guesser, up to a constant $n_i \ln L$.

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

$$f(w_i) = \ln L + w_i\ln w_i + (1-w_i)\ln\frac{1-w_i}{L-1}, \tag{5.5}$$

where $f(w_i)\in[0,\ln L]$.

As we have explained, this measure is well justified by information theory and has strong connection with the EM algorithm.

5.4 Worker group discovery algorithms

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

As we have mentioned in Section 5.2, the problem is to automatically find groups of workers that are more helpful on the task, which is measured by the *effects* proposed in the previous section. To compute effects, we need to estimate worker accuracy w_i from the data we gathered in the probing stage. We could either leverage some ground truth data to directly compute it, or estimate it using the EM algorithm if there is no ground truth.

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

Based on the ideas above, we designed two algorithms: the Bottom-up Discovery Algorithm and the Top-down Discovery Algorithm as follows.

5.4.1 Bottom-up Discovery Algorithm

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

For analyzing the association between the worker features and the worker effect, intuitively we can apply the the fixed effect model (Mickey et al., 2004):

$$\tau_i \sim \beta_0 + \beta_1 X_i^{(1)} + \cdots + \beta_t X_i^{(t)} + \epsilon, \quad \forall i \in [M], \tag{5.6}$$

where $\beta = (\beta_0, \beta_1, \cdots, \beta_t)$ are coefficients and $\epsilon \sim \mathcal{N}(0, \sigma^2)$, i.e., Gaussian noise with mean 0. Note that X_i could be categorical variables, and coded as a vector via dummy coding (Mickey et al., 2004). In that case β_i will be a vector and each element of it will be an fixed effect coeffecient of a level of F_i.

The Bottom-up Discovery Algorithm is described in Algorithm 4. The fixed effect model as in (5.6) is easy to fit by multiple regression after dummy coding, and many packages are available in public software such as R[1] and Matlab[2]. After fitting this fixed effect model

[1] http://www.ats.ucla.edu/stat/r/modules/dummy_vars.htm

[2] http://www.mathworks.com/help/stats/linearmodel.fit.html

Algorithm 4 Bottom-up Discovery Algorithm

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

1: Fit the fixed effect model (5.6) to learn the model parameters $(\hat{\beta}_0, \hat{\beta}_1, \cdots, \hat{\beta}_t)$.
2: Obtain the fitted value for each workers by
$\hat{\tau}_i \leftarrow \hat{\beta}_0 + \hat{\beta}_1 X_i^{(1)} + \cdots + \hat{\beta}_t X_i^{(t)}, \quad \forall i \in [M]$
3: Rank fitted effect $\{\hat{\tau}_i\}$ in descending order: $\hat{\tau}_{\sigma(1)} \geq \hat{\tau}_{\sigma(2)} \geq \cdots \geq \hat{\tau}_{\sigma(M)}$, where σ is a permutation of worker index $\{1, 2, \cdots, M\}$.
4: The threshold $\tau_0 \leftarrow \min\left\{\hat{\tau}_{\sigma(i)} \middle| i \in [M] \text{ and } \frac{i}{M} \geq \lambda\right\}$.

Output: The learned model parameter $(\hat{\beta}_0, \hat{\beta}_1, \cdots, \hat{\beta}_t)$ and the effect threshold τ_0.

to data gathered during the probing stage, we will learn the coefficients $\hat{\beta}$ and a threshold τ_0 on predict effect for a worker to be in the target group. For a worker with features $(X_1, X_2, \cdots, X_t)$ in the targeting stage, we evaluate the effect of the subgroup he/she belongs to with $\hat{\tau} \leftarrow \hat{\beta}_0 + \sum_{k=1}^{t} \hat{\beta}_k X_k$. The worker will be qualified for the task if $\hat{\tau} > \tau_0$.

Note that the accessibility parameter λ reflects roughly how many worker in the crowd satisfy the criterion — with predicted effect greater than τ_0. It thus controls how accessible the target group will be. For example, if λ is close to 0, then τ_0 will be close to 1, thus there might be only very few qualifying workers. In contrast, if λ is close to 1, τ_0 will be small, and most workers will qualify.

5.4.2 Top-down Discovery Algorithm

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

The general idea is we should choose subgroups based on features that are significantly associated with worker effects. ANOVA (ANalysis Of VAriance) (Stahle and Wold, 1989) based on the fixed effect model (5.6) is an appropriate tool for testing feature significance.

One remaining issue is that when there are multiple features in $\mathbb{F}$, and each feature have multiple levels (multiple possible values), the number of workers that have the same features might be too small, especially when M is already very small (typically 100 or 200 in real crowd-sourcing settings). The multiple-way ANOVA will be unstable in such case. More importantly, for achieving interpretability and reducing the risks of over-fitting, we also hope

that output worker subgroups are not too many.

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

$$\tau_i \sim \beta_0 + \beta_1 X_i^{(k)} + \epsilon, \quad \forall i \in [M], F_k \in \mathbb{F}. \tag{5.7}$$

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

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

Algorithm 5 Top-down Discovery Algorithm

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

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

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

Algorithm 5 is the detailed description of the Top-down Discovery Algorithm. The general steps include: (1) sequentially testing if features are significantly associated with worker effects, (2) splitting the crowd using the most significant features, and (3) picking the sub-

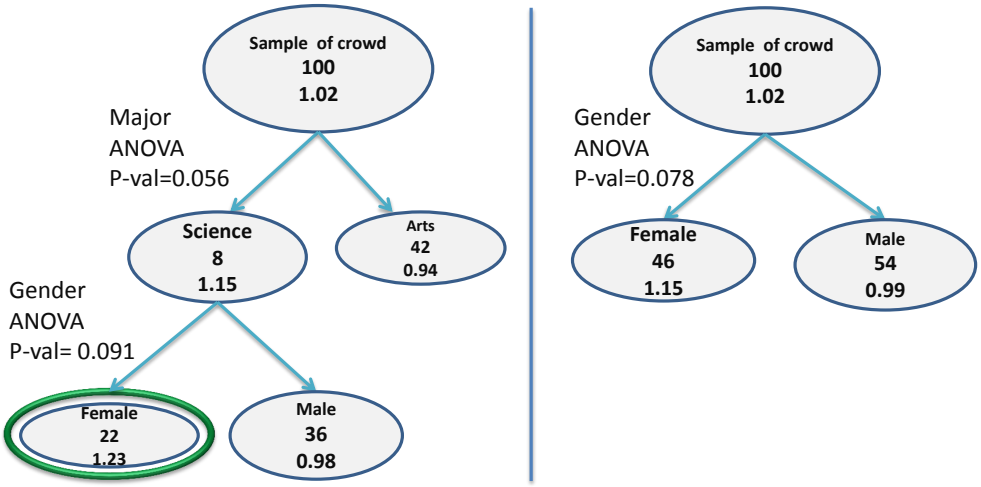


Figure 5.3: Illustration of group discovery algorithm by a running example: suppose we have 100 workers and then only have features "Gender" and "Major". Each circle represents a group of workers, the integer in each circle is the number of workers in the group and the real value is the average worker effect. Since the feature "Major" has smaller p-value than "Gender" (more significant), the algorithm splits the crowd on "Major" and chooses the worker group with the highest average effect: "Science". Then it continues with the rest of the features. In the end, the algorithm outputs Major = Science and Gender = female as the target group.

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

As an illustration, Figure 5.3 shows a running example of this algorithm. Suppose we have hired 100 workers in the probing stage, and the feature pool we choose to run the Top-down Discovery Algorithm is $\mathbb{F}$ ={Major, Gender}, and the feature "Major" has two levels {"Science", "Arts"}. We choose the accessibility parameter $\lambda = 0.20$ and the default significance threshold $p_{sig} = 0.10$. We first conduct one-way ANOVA test on both of the features, and find that "Major" has the lower p-value 0.056, which is significant. Then we choose the level "Science" to be the current target crowd. By testing feature "Gender" again, we find it is still significant, then we further split the target crowd and reach {Major = Science, Gender = Female}. If the size of this target group is larger than $100 \cdot \lambda = 20$, then we directly output the group. Otherwise, we just output the parent group {Major = Science}.

Feature merging step

One practical issue in the Top-down Discovery Algorithm is that for some characteristics of workers, such as major or nationality, there might be too many levels. Therefore, the

number of workers in each level could be too small, which could be a serious issue for the Top-down Discovery Algorithm since the algorithm may potentially partition the crowd into too small sets, which will easily violate the accessibility criterion (controlled by parameter λ in Algorithm 5).

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

Algorithm 6 Feature Merging

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

if $K \leq 2$ **then**
  $F^* \leftarrow F$ and no need to merge its levels.
else
  for k from 1 to K **do**
    $S_k \leftarrow \{i \in [M] | X_i = \mathcal{L}_k\}$
    $E_k \leftarrow \frac{1}{|S_k|} \sum_{i \in S_k} \tau_i$
  end for
  Ranking $\{E_k\}$ to obtain the order σ such that $E_{\sigma(1)} \geq E_{\sigma(2)} \geq \cdots \geq E_{\sigma(K)}$.
  $k^* \leftarrow \max\{k | \sum_{1 \leq i \leq k-1} S_{\sigma(i)} < \frac{M}{2}, \sum_{1 \leq i \leq k} S_{\sigma(i)} \geq \frac{M}{2}\}$.
  $k^* \leftarrow \min\{k^*, K-1\}$ for avoiding combine all levels.
  $\mathcal{L}'_1 \leftarrow \mathcal{L}_{\sigma(1)} \cup \cdots \cup \mathcal{L}_{\sigma(k^*)}$, $\mathcal{L}'_2 \leftarrow F \backslash \mathcal{L}'_1$, and $F^* \leftarrow \{\mathcal{L}'_1, \mathcal{L}'_2\}$.
end if

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

Algorithm 6 presents the details for merging multi-level features. The major idea of this algorithm is that: (1) Partitioning the M workers from the probing stage into subgroups by feature F. (2) Ranking the mean effect of subgroups in descending order. (3) Combining the levels in the sorted order into one target level until the size of the combined group is larger than half, i.e. $\frac{M}{2}$.

Figure 5.4 illustrate the idea of Algorithm 6 with a running example. Suppose we have 100 workers in the probing stage and the feature "Major" has 4 levels {Art, Business, Science, Engineering}. We first compute the average worker effect of the workers in each major, then rank them from high to low. After combing major levels from the higher effect to the lower one until the number of workers in the group is above 50 (i.e., half of the crowd), "Science/Engineering" is a new level, and "Business/Arts" is another new level.

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

(1)

	Art.	Bus.	Sci.	Eng.
Effect	0.9	1.0	1.2	1.1
Popul.	25	17	30	28

(2)

	Sci.	Eng.	Bus.	Art.
Effect	1.2	1.1	1.0	0.9
Popul.	30	28	17	25

(3)

	Science/Engineering	Business/Arts
Effect	1.15	0.94
Popul.	58	42

Figure 5.4: A running example of the feature merging algorithm: (1) suppose we have 100 workers, and the feature "Major" contains 4 levels. "Popul." represents the number of workers in each major. (2) Combing majors with higher average effect until the number of workers in the combined group is above 50 (half of the crowd). (3) "Science/Engineering" is a new level, and "Business/Arts" is another new level.

5.5 Experiments

5.5.1 Setup

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

Knowledge test data

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

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

Normally, aggregating the answers from 10 to 20 workers for a question is good enough empirically (Snow et al., 2008). However, we want to investigate how the performance of

aggregation algorithms such as majority voting or EM increases w.r.t. the number of answers provided for each question. Therefore, we required each worker to finish all the questions. We distributed the tasks on UHRS and set the number of workers needed to 100, and the price was $1.5 for finishing all the questions.

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

Disambiguation data

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

Figure 5.5 shows a typical example of this task. The word "runtime" has different meanings in different contexts. In this question, the correct answer is the second option.

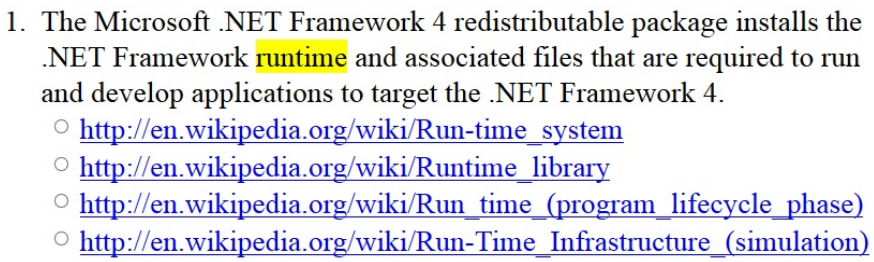


Figure 5.5: A typical example of disambiguation questions. The word "runtime" has different meanings, and the task is to identify which Wikipedia page it actually refers to in this sentence. The correct answer is the second option.

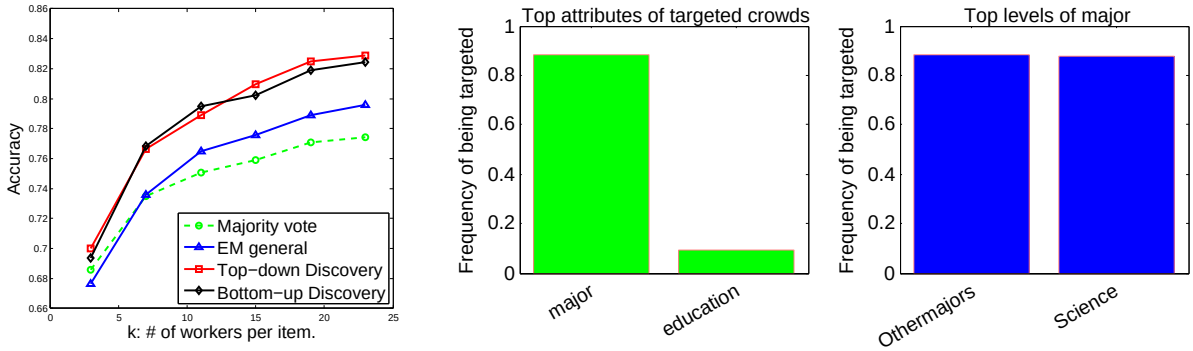
We collected 50 of such questions in the technology domain with ground truth available, and each question has 4 options. Then we randomly partitioned them into two even sets, which we call disambiguation data part 1 and part 2.

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

Brief statistics on the disambiguation data is presented in Figure 5.6. Part one has 144 workers and part two has 133. We can compute the "true" accuracy of each worker using the ground truth: the average worker accuracy is 55.9% on part 1 and 57.2% on part 2, which is fairly low meaning the task is quite difficult. Majority voting and EM have reasonably good accuracy, but note that it is achieved by running against all the labels in each dataset.

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

Figure 5.6: A brief summary statistics of the disambiguation dataset. The performance of both MV and EM are based on all the labels in each dataset.



(a) (b) (c)

Figure 5.7: Knowledge test data. (a) Comparing majority voting, EM without targeting, and EM with Bottom-up Discovery Algorithm and Top-down Discovery Algorithm respectively. (b) Plotting the frequency of significant features output by Top-down Discovery Algorithm. (c) The frequency plot of top features in "Major".

Recognizing textual entailment data

This data is sampled from the well known public crowdsourcing dataset by Snow *et al.* (Snow et al., 2008). For each question, a worker is presented with two sentences and a binary choice of whether the second sentence can be inferred from the first. As a typical example given in (Snow et al., 2008), the sentence "Oil prices drop" would constitute a true entailment from the text "Crude Oil Prices Slump", but a false entailment from "The government announced last week that it plans to raise oil prices".

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

5.5.2 Experimental results

In the experiments, the estimated worker accuracy is transformed into the worker effect based on Information Gain defined in (5.5). Worker characteristic information is collected before the worker answers any regular questions. We investigated which group will be more suitable for the task by Top-down Discovery Algorithm after merging the multi-level features to bi-level by Algorithm 6. We also apply the Bottom-up Discovery Algorithm to learn the fixed effect model and effect threshold τ_0 without merging the feature.

Results on knowledge test data

In this experiment, we randomly sample 5 out of 75 questions as the questions used in the probing stage. We estimate the worker accuracy by comparing the answers of these 5 questions from each worker against the ground truth, and then run worker group discovery to get the targeted worker groups.

For the rest of the questions (excluding the 5 exam questions), we basically simulate the targeting stage by randomly sampling k workers from the targeted groups for each question, and then we run the EM algorithm on the answers provided by the selected workers. We call this *EM with targeting*.

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

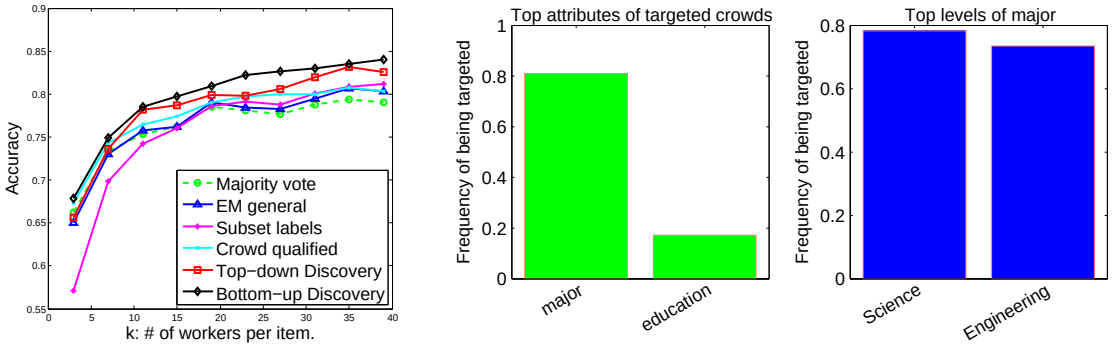
Figure 5.7(a) shows the comparison. We can see generally the performance of EM with targeting is dominating the results of EM and MV without targeting. This indicates that the data quality is significantly improved by our crowd-targeting framework since we compare exactly the same EM algorithm on the data collected from two different ways — one is only from the targeted group discovered in the probing stage, another is from the general crowd.

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

Figure 5.7 (b) and (c) visualized the frequencies of the features selected for identifying targeted groups. Apparently, "Major" is the most important feature, and "Other majors" and "Science" are the two top majors. Note that "Other major" represent all majors except "Science", "Engineering", "Literature" and "Arts".

Results on Disambiguation data

In this experiment, we randomly divided the question set into to two parts, which we called Disambiguation dataset part 1 and part 2 respectively as we have discribed in Section 5.5.1.



(a) (b) (c)

Figure 5.8: Disambiguation dataset. (a) Comparing majority voting, EM without targeting, and EM with targeting. We randomly sample 5 questions as "Exam" questions to workers, and tested on the rest questions in this part of data. (b) The top two targeted features and their frequency of been targeted. (c) The top two levels of the feature "Major" and their frequencies.

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

Figure 5.8(a) shows the comparison of different methods on the Disambiguation dataset 1. We get similar results on part 2, so we omit them due to limited space.

In the results, we refer the "Top-down Discovery" as running EM with targeting crowd learned from Top-down Discovery Algorithm, and "Bottom-up Discovery' as from Bottom-up Discovery Algorithm. "EM general" is running EM without targeting any specific groups, and majority voting is also running on general crowd.

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

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

What we can observe from Figure 5.8(a) is that the performance of "EM general" is better than that of majority voting, but worse than that of "Crowd qualified". And EM on the targeted crowd from both Top-down Discovery and Bottom-up Discovery performs better than all the others. One interesting phenomenon we can see from the figure is about

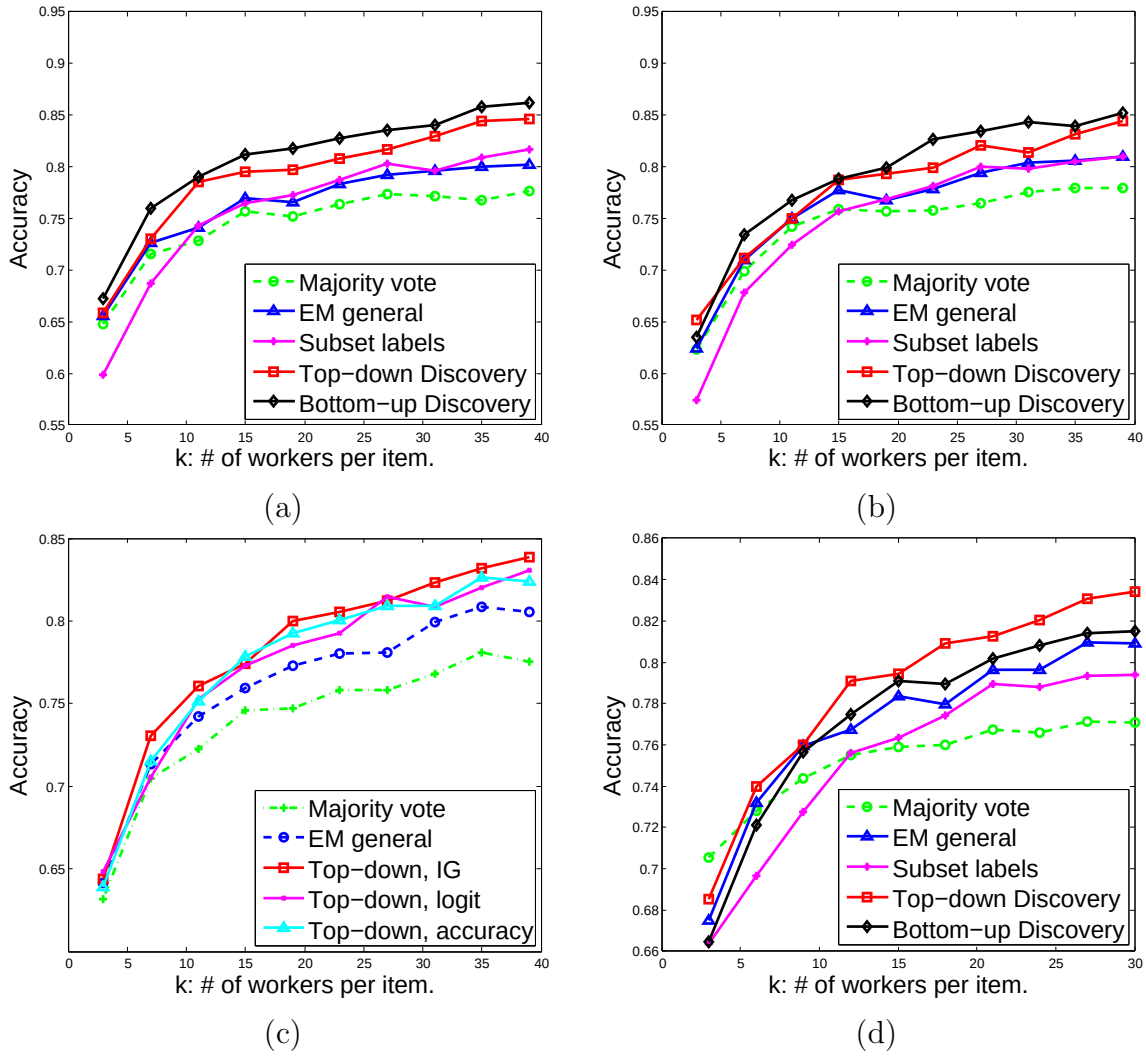


Figure 5.9: (a),(b) and (c) is on Disambiguation data, and (d) is on RTE data: comparing EM on general crowd and target crowd with majority voting. Use Disambiguation data part 1 for learning in probing stage, and tested on data part 2. (a) With ground truth labels: evaluate workers by comparing with the ground truth labels of 5 randomly drawn "exam" questions (b) Without ground truth labels: draw k items in dataset part 1 and inference the worker effect without ground truth labels. Discover groups based estimated worker effects. (c) Compare three effect types by running Top-down Discovery Algorithm without ground truth labels on Disambiguation dataset. (d) RTE dataset, learning without ground truth and using same setting as (b).

comparing "EM general" with "Subset labels". Since both of them are running the same algorithm – EM, but the latter use only a subset data of the previous one. When k is small, the subset data will contains much less "useful" information than the data used by "EM

general", so the performance of EM on "Subset labels" is much less than "EM general". However, when k increases to a certain level, using the subset data will even be better than using more data. This could be due to the Signal-Noise-Ratio is much higher in the data provided by targeted crowd than that by general crowd. This is the essential philosophy of "the wisdom of the minority".

Figure 5.8(b)&(c) shows the frequency plot as explained in Figure 5.7. For the disambiguation dataset, the most important feature is major, and the target group concentrated on the workers who majored in "Science" and "Engineering".

Since having seen promising results of utilizing crowd-targeting with the "Exam" questions and testing on the rest of the questions from the same dataset, we want to know if the similar results hold when we do crowd-targeting on one crowd and question set, then test on a different crowd and question set. Figure 5.9(a) shows the results when we randomly choose 5 questions which we know the ground truth from dataset part 1, conduct the crowd-targeting, and then test on the dataset part 2 which is collected form a different crowd and question pool. The EM algorithm with both Bottom-up Discovery and Top-down Discovery perform better than that with data collected from the general crowd. The frequency plot based on the output from Top-down Discovery Algorithm are very similar to Figure 5.8 (b)&(c). We omit them here for saving space.

Next, what should we do if there is no ground truth answers available? Can the crowd-targeting framework still be applied? In the next experiment, we randomly sample k workers for each question in Disambiguation dataset 1, then infer the true labels and the worker accuracy by EM algorithm without any ground truth labels. After obtaining the estimated worker accuracy, we apply the both Bottom-up Discovery Algorithm and Top-down Discovery Algorithm to the workers in dataset part 1. On the test stage, we also randomly sampled k workers for each question in dataset part 2 from the target group, run EM algorithm, and then compare with running EM on the data sampled from general crowd. The results are shown in figure 5.9(b). What we can see from these two figures is basically the same with the previous results (targeting crowd with several ground truth labels). The only difference is that the performance of the EM algorithm drop slightly. The features selected by the Top-down Discovery Algorithm is almost the same as what is shown in Figure 5.8(b)&(c).

In the last experiments on this dataset, we compare the different effects — Information Gain, accuracy and logistic as discussed in Section 5.3. The results is shown in Figure 5.9(c). The likelihood contribution performs the best but not significantly better than the other two. This is because when a workers is better than random guessing, this three effect measures will be highly correlated.

Results on RTE data

We repeated the similar experiments on RTE dataset and obtained similar results — the performance of the EM algorithm on the data collected from the target group, discovered by both Top-down Discovery Algorithm and Bottom-up Discovery Algorithm, are better than the EM algorithm on the data collected without crowd targeting. Note that in RTE

data $L = 2$ and crowd size are smaller than the previous two datasets. We run worker group discovery algorithms on RTE dataset 1 without ground truth labels as the same as 5.9(b), and test on RTE dataset 2. The Top-down Discovery Algorithm performs better than Bottom-up Discovery Algorithm t from the experimental results on the RTE data.

5.6 Related work

In recent years, many works have been done on improving data quality for crowdsourcing. A lot of them focused on the joint inference of true labels of items and worker reliability after data are collected (Dawid and Skene, 1979, Whitehill et al., 2009, Ipeirotis et al., 2010, Raykar et al., 2010, Welinder et al., 2010, Karger et al., 2011a, Bachrach et al., 2012, Liu et al., 2012, Zhou et al., 2012). These methods are generally post-processing methods, so they cannot be easily used during task assignment to improve data quality. Our proposed crowd targeting framework is complementary with these methods, and they can be used in our framework to estimate worker quality when ground truth is not available, as shown in the experiments.

Some recent work started to investigate quality and budget optimization techniques that can be more tightly integrated with task assignment (Yan et al., 2011, Ertekin et al., 2012, Karger et al., 2011b, Pfeiffer et al., 2012, Chen et al., 2013, Ho et al., 2013). However, these methods often implicitly assume that task owners have full control of the crowdsourcing system so that when they need a specific worker to label an item, this worker will be available and willing to complete the task, which may not be a very practical assumption in most of the current crowdsourcing platforms. In our crowd targeting framework, we only target on high quality worker groups instead of individual workers, and therefore the availability of workers is more likely to be guaranteed.

Quite a few works focused on worker characteristics in crowdsourcing. (Ipeirotis, 2010a, Ross et al., 2010) provided detailed analysis of demographic distribution of workers on Amazon Mechanic Turk. (Schmidt, 2010) proposed to collected demographic information for human subject research so that researchers can know better who are participating in the studies. Recently, Kazai et al. (Kazai et al., 2012, 2013) performed some empirical studies on how worker reliability fluctuates cross different worker demographic groups for some relevance assessment tasks.

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

5.7 Discussion

The Crowd Targeting framework can be applied to many tasks as long as we can measure the worker effect. It is also useful in practice since it can be implemented in the current crowd-

sourcing platform without too much effort. However, there might be some concerns that task owners might have, such as the question — *what should we do if some characteristics are too sensitive to us?* For example, the task owners might do not want to identify gender for this task. What he/she could do is simply not include the feature in the candidate feature pool, thus the algorithms will not take this feature into account. Note that even the Top-down Discovery Algorithm outputs "gender as female as an important feature of workers to be in the target group, this does not imply that for all type of tasks, female workers in the crowd will be suitable. The target crowd is very much task-dependent. Therefore, this framework has no bias to any worker groups in general.

5.8 Conclusions

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

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

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

Bibliography

Dana Angluin and Philip Laird. Learning from noisy examples. *Machine Learning*, 2(4): 343–370, 1988.

Yoram Bachrach, Thore Graepel, Tom Minka, and John Guiver. How to grade a test without knowing the answers — a bayesian graphical model for adaptive crowdsourcing and aptitude testing. In *ICML*, pages 1183–1190, New York, NY, USA, 2012.

Jeff A. Bilmes. A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. *Technique Report*, 1198 (510), 1998.

Xi Chen, Qihang Lin, and Dengyong Zhou. Optimistic Knowledge Gradient Policy for Optimal Budget Allocation in Crowdsourcing. In *ICML*, 2013.

Fan RK Chung and Linyuan Lu. *Old and new concentration inequalities, Chapter 2 in Complex Graphs and Networks*. AMS, 2010. ISBN ISBN-10:0-8218-3657-9.

Seth Cooper, Firas Khatib, Adrien Treuille, Janos Barbero, Jeehyung Lee, Michael Beenen, Andrew Leaver-Fay, David Baker, Zoran Popović, et al. Predicting protein structures with a multiplayer online game. *Nature*, 466(7307):756–760, 2010.

Nilesh Dalvi, Anirban Dasgupta, Ravi Kumar, and Vibhor Rastogi. Aggregating crowdsourced binary ratings. In *WWW*, 2013.

Alexander P. Dawid and Allan M. Skene. Maximum Likelihood Estimation of Observer Error-Rates Using the EM Algorithm. *Journal of the Royal Statistical Society.*, 28(1): 20–28, 1979.

Ofer Dekel and Ohad Shamir. Good learners for evil teachers. *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1–8, 2009. doi: 10.1145/1553374.1553404.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern classification*. John Wiley & Sons, 2012.

Seyda Ertekin, Haym Hirsh, and Cynthia Rudin. Approximating the Wisdom of the Crowd. In *NIPS Workshop on Computational Social Science and the Wisdom of Crowds*, pages 1–5, 2011.

Seyda Ertekin, Haym Hirsh, and Cynthia Rudin. Learning to predict the wisdom of crowds. In *Collective Intelligence conference*, 2012.

Francis Galton. The ballot-box. *Nature*, 75:509–510, 1907a.

Francis Galton. Vox populi (the wisdom of crowds). *Nature*, 75:450–451, 1907b.

Chao Gao and Dengyong Zhou. Minimax optimal convergence rates for estimating ground truth from crowdsourced labels. *arXiv:1310.5764*, 2014.

Arpita Ghosh, Satyen Kale, and Preston McAfee. Who moderates the moderators?: crowdsourcing abuse detection in user-generated content. In *Proceedings of the 12th ACM conference on Electronic commerce*, pages 167–176. ACM, 2011.

Mark Herbster and Manfred K. Warmuth. Tracking the best expert. *Machine Learning*, 32 (2):151–178, 1998.

Chien-Ju Ho, Shahin Jabbari, and Jennifer W. Vaughan. Adaptive Task Assignment for Crowdsourced Classification. In *ICML*, 2013.

Wassily Hoeffding. On the distribution of the number of successes in independent trials. *The Annals of Mathematical Statistics*, pages 713–721, 1956.

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

Panagiotis G. Ipeirotis. Demographics of mechanical turk. In *NYU Digital Working Paper CeDER-10-01*, 2010a.

Panagiotis G. Ipeirotis. Analyzing the amazon mechanical turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students*, 17(2):16–21, 2010b.

Panagiotis G Ipeirotis, Foster Provost, and Jing Wang. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 64–67. ACM, 2010.

Rong Jin and Zoubin Ghahramani. Learning with Multiple Labels. In *NIPS*, 2002.

Manas Joglekar, Hector Garcia-Molina, and Aditya Parameswaran. Evaluating the crowd with confidence. In *SIGKDD*, pages 686–694. ACM, 2013.

David R. Karger, Sewoong Oh, and Devavrat Shah. Iterative learning for reliable crowdsourcing systems. In *NIPS*, 2011a.

David R. Karger, Sewoong Oh, and Devavrat Shah. Budget-optimal crowdsourcing using low-rank matrix approximations. In *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*, pages 284–291. IEEE, 2011b.

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

Gabriella Kazai, Jaap Kamps, and Natasa Milic-Frayling. The face of quality in crowdsourcing relevance labels: Demographics, personality and labeling accuracy. In *Proceedings of the 21st ACM Conference on Information and Knowledge Management (CIKM 2012)*. ACM Press, New York NY, 2012.

Gabriella Kazai, Jaap Kamps, and Natasa Milic-Frayling. An analysis of human factors and label accuracy in crowdsourcing relevance judgments. *Information Retrieval*, 16:138–178, 2013.

Josef Kittler. Combining classifiers: A theoretical framework. *Pattern analysis and Applications*, 1(1):18–27, 1998.

Hongwei Li and Qiang Liu. Cheaper and better: Selecting good workers for crowdsourcing. *arXiv preprint arXiv:1502.00725*, 2015.

Hongwei Li and Bin Yu. Error rate bounds and iterative weighted majority voting for crowdsourcing. *arXiv preprint arXiv:1411.4086*, 2014.

Hongwei Li, Bin Yu, and Dengyong Zhou. Error Rate Analysis of Labeling by Crowdsourcing. In *ICML Workshop: Machine Learning Meets Crowdsourcing. Atalanta, Georgia, USA.*, 2013a.

Hongwei Li, Bin Yu, and Dengyong Zhou. Error rate bounds in crowdsourcing models. *arXiv preprint arXiv:1307.2674*, 2013b.

Hongwei Li, Bo Zhao, and Ariel Fuxman. The wisdom of minority: discovering and targeting the right group of workers for crowdsourcing. In *Proceedings of the 23rd international conference on World wide web*, pages 165–176, 2014.

Qiang Liu, Jian Peng, and Alex T. Ihler. Variational Inference for Crowdsourcing. In *NIPS*, 2012.

Qiang Liu, Alex T Ihler, and Mark Steyvers. Scoring workers in crowdsourcing: How many control questions are enough? In *NIPS*, pages 1914–1922, 2013.

Albert E. Mannes, Jack B. Soll, and Richard P. Larrick. The wisdom of select crowds. *Journal of personality and social psychology*, 107(2):276, 2014.

Colin McDiarmid. Concentration. *Technique Report*, 1998. URL http://cgm.cs.mcgill.ca/~breed/conc/colin.pdf.

Ruth M Mickey, Olive Jean Dunn, and Virginia Clark. *Applied statistics: analysis of variance and regression*. Wiley-Interscience, 2004.

Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *NIPS*, pages 1196–1204, 2013.

Hung Q. Ngo. Tail and Concentration Inequalities. *Lecture Notes*, pages 1–6, 2011. URL http://www.cse.buffalo.edu/~hungngo/classes/2011/Spring-694/lectures/l4.pdf.

Thomas Pfeiffer, Xi Alice Gao, Yiling Chen, Andrew Mao, and David G Rand. Adaptive polling for information aggregation. In *AAAI*, 2012.

Vikas C Raykar and Shipeng Yu. Eliminating spammers and ranking annotators for crowdsourced labeling tasks. *The Journal of Machine Learning Research*, 13:491–518, 2012.

Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning From Crowds. *Journal of Machine Learning Research*, 11:1297–1322, 2010.

Joel Ross, Lilly Irani, M Silberman, Andrew Zaldivar, and Bill Tomlinson. Who are the crowdworkers?: shifting demographics in mechanical turk. In *CHI*, pages 2863–2872. ACM, 2010.

Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173, 2008.

Lauren A. Schmidt. Crowdsourcing for human subjects research. *Proceedings of CrowdConf*, 2010.

Victor S Sheng, Foster Provost, and Panagiotis G. Ipeirotis. Get Another Label? Improving Data Quality and Data Mining Using Multiple, Noisy Labelers Categories and Subject Descriptors. *SIGKDD*, pages 614–622, 2008.

Padhraic Smyth, Usama Fayyad, Michael Burl, Pietro Perona, and Pierre Baldi. Inferring Ground Truth from Subjective Labelling of Venus Images. In *NIPS*, 1995.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. Cheap and Fast - But is it Good ? Evaluating Non-Expert Annotations for Natural Language Tasks. *EMNLP*, 2008.

Lars Stahle and Svante Wold. Analysis of variance (anova). *Chemometrics and intelligent laboratory systems*, 6(4):259–272, 1989.

James Surowiecki. *The wisdom of crowds*. Anchor, 2005.

Long Tran-Thanh, Matteo Venanzi, Alex Rogers, and Nicholas R Jennings. Efficient budget allocation with accuracy guarantees for crowdsourcing classification tasks. In *Proceedings of the 2013 international conference on autonomous agents and multi-agent systems*, pages 901–908. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

Luis Von Ahn. Games with a purpose. *Computer*, 39(6):92–94, 2006.

Luis Von Ahn and Laura Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326. ACM, 2004.

Luis Von Ahn, Manuel Blum, Nicholas J Hopper, and John Langford. Captcha: Using hard ai problems for security. In *Advances in CryptologyEUROCRYPT 2003*, pages 294–311. Springer, 2003.

Luis Von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. reCAPTCHA: Human-based character recognition via web security measures. *Science*, 321(5895):1465–1468, 2008.

Peter Welinder, Steve Branson, Pietro Perona, and Serge J. Belongie. The Multidimensional Wisdom of Crowds. In *NIPS*, 2010.

Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R. Movellan, and Paul L. Ruvolo. Whose Vote Should Count More : Optimal Integration of Labels from Labelers of Unknown Expertise. In *NIPS*, 2009.

Yan Yan, Rómer Rosales, Glenn Fung, Mark W Schmidt, Gerardo H Valadez, Luca Bogoni, Linda Moy, and Jennifer G. Dy. Modeling annotator expertise : Learning when everybody knows a bit of something. In *International conference on artificial intelligence and statistics*, volume 9, pages 932–939, 2010.

Yan Yan, Glenn M Fung, Rómer Rosales, and Jennifer G. Dy. Active Learning from Crowds. In *ICML*, pages 1161–1168, 2011.

Man-Ching Yuen, Irwin King, and Kwong-Sak Leung. A survey of crowdsourcing systems. In *Privacy, security, risk and trust (passat), 2011 ieee third international conference on and 2011 ieee third international conference on social computing (socialcom)*, pages 766–773. IEEE, 2011.

Dengyong Zhou, Sumit Basu, Yi Mao, and John C. Platt. Learning from the Wisdom of Crowds by Minimax Entropy. In *NIPS*, 2012.

Appendix A

Proofs for Chapter 2

Since the proof of Theorem 2.1 requires other theorems in this paper, we will not present the proofs in the same order as in the paper. The order of our proofs will be: Proposition A.1, Theorem 2.3, Theorem 2.1 and then Theorem 2.2. After proving the first four main results, we also prove Corollary 2.8.

Before presenting the proofs, we would like to propose some notations for simplicity and clarity.

We simplify $f_i(k,h)$ to

$$\mu_{kh}^{(i)} \doteq f_i(k,h), \quad \forall k \in [L], h \in \overline{[L]}, \tag{A.1}$$

then each worker is associated with a vote matrix $\mu^{(i)} = (\mu_{kh}^{(i)}), k \in [L], h \in \overline{[L]}$, where $\mu_{kh}^{(i)}$ is the voting score when worker i labels item j whose true label is k, as class h if $h \neq 0$. Then the aggregation rule (2.9) is equivalent to

$$\hat{y}_j = \underset{k\in[L]}{\operatorname{argmax}} \left(\sum_{i=1}^{M} \sum_{h=1}^{L} \mu_{kh}^{(i)} \mathrm{I}\left(Z_{ij} = h\right) \ + \ a_k \right), \tag{A.2}$$

Note that

$$s_k^{(j)} = \sum_{i=1}^{M} \sum_{h=1}^{L} \mu_{kh}^{(i)} \mathrm{I}\left(Z_{ij} = h\right) + a_k, \quad \forall k \in [L], j \in [N] \tag{A.3}$$

is the aggregated score of label class k on the jth item, and the general aggregation rule is

$$\hat{y}_j = \underset{k\in[L]}{\operatorname{argmax}}\, s_k^{(j)}. \tag{A.4}$$

We will frequently discuss conditional probability, expectation and variance conditioned on the event $\{y_j = k\}$. For simplicity of notations, we define:

$$\mathbb{P}_k(\ \cdot\ ) \doteq \mathbb{P}(\ \cdot\ | y_j = k) \tag{A.5}$$

$$\mathbb{E}_k[\ \cdot\ ] \doteq \mathbb{E}[\ \cdot\ | y_j = k] \tag{A.6}$$

$$\mathrm{Var}_k(\ \cdot\ ) \doteq \mathrm{Var}(\ \cdot\ | y_j = k). \tag{A.7}$$

Note that

$$\mathbb{E}_k\left[s_l^{(j)}\right] = \sum_{i=1}^{M}\sum_{h=1}^{L} q_{ij}\mu_{lh}^{(i)}p_{kh}^{(i)} + a_l, \quad \forall l,k \in [L]. \tag{A.8}$$

A.1 Proposition A.1 and its proof

Proposition A.1. ***(Bounding the mean error rate of labeling each item)*** *Following the setting of Theorem 2.1, and with $\tau_{j,min}$ and $\tau_{j,max}$ defined as in (2.14) , we have $\forall j \in [N]$,*
(1) if $\tau_{j,min} \geq 0$, then $\mathbb{P}(\hat{y}_j \neq y_j) \leq (L-1)\cdot \min\left\{\exp\left(-\frac{\tau_{j,min}^2}{2}\right), \exp\left(-\frac{\tau_{j,min}^2}{2(\sigma^2+c\tau_{j,min}/3)}\right)\right\}$;
(2) if $\tau_{j,max} \leq 0$, then $\mathbb{P}(\hat{y}_j \neq y_j) \geq 1 - \min\left\{\exp\left(-\frac{\tau_{j,max}^2}{2}\right), \exp\left(-\frac{\tau_{j,max}^2}{2(\sigma^2-c\tau_{j,max}/3)}\right)\right\}$.

Remark: Proposition A.1 provides the mean error rate bounds of labeling any specific item, and the bounds depend on the minimum and maximum values of $\left\{\Lambda_{kl}^{(j)}\right\}_{k,l\in[L]}$. Note that the subscript j only comes from the assignment distribution q_{ij}. If a specific worker has the same assignment probability to label all items, say q_i, then we can drop the subscript j from $\tau_{j,\min}$ and $\tau_{j,\max}$, which means the error rate bounds of each item are eventually the same under that task assignment.

Proof. First of all, we expand the error probability of labeling the j-th item wrong in terms of the conditional probabilities:

$$\mathbb{P}(\hat{y}_j \neq y_j) \;=\; \sum_{k\in[L]} \mathbb{P}(y_j = k)\mathbb{P}(\hat{y}_j \neq k | y_j = k) \;=\; \sum_{k\in[L]} \pi_k \mathbb{P}_k\left(\hat{y}_j \neq k\right). \tag{A.9}$$

Our major focus in this proof is to bound the term $\mathbb{P}_k\left(\hat{y}_j \neq k\right)$. Our approach will be based on the following events relations:

$$\bigcup_{l\in[L],l\neq k}\left\{s_l^{(j)} > s_k^{(j)}\right\} \;\subseteq\; \{\hat{y}_j \neq k\} \;\subseteq\; \bigcup_{l\in[L],l\neq k}\left\{s_l^{(j)} \geq s_k^{(j)}\right\}. \tag{A.10}$$

(1). Assuming $\tau_{j,\min} \geq 0$, we want to show the lower bound for $\mathbb{P}(\hat{y}_j \neq y_j)$. Note that

$$\mathbb{P}_k\left(\hat{y}_j \neq k\right) \;\leq\; \mathbb{P}_k\left(\bigcup_{l\in[L],l\neq k}\left\{s_l^{(j)} \geq s_k^{(j)}\right\}\right) \leq \sum_{l\in[L],l\neq k}\mathbb{P}_k\left(s_l^{(j)} \geq s_k^{(j)}\right). \tag{A.11}$$

With $s_l^{(j)}$ defined as in (A.3), $\Lambda_{kl}^{(j)}$ defined as in (2.13) and

$$\xi_{kl}^{(i)} \doteq \sum_{h=1}^{L}(\mu_{lh}^{(i)} - \mu_{kh}^{(i)})\mathrm{I}\left(Z_{ij} = h\right), \tag{A.12}$$

$$\mathbb{E}_k\left[\xi_{kl}^{(i)}\right] = \sum_{h=1}^{L} q_{ij}\left(\mu_{lh}^{(i)} - \mu_{kh}^{(i)}\right)p_{kh}^{(i)}, \tag{A.13}$$

we have

$$\begin{aligned}
\mathbb{P}_k\left(s_l^{(j)} \geq s_k^{(j)}\right) &= \mathbb{P}_k\left(\sum_{i=1}^{M}\sum_{h=1}^{L}\left(\mu_{lh}^{(i)}-\mu_{kh}^{(i)}\right)\mathrm{I}\left(Z_{ij}=h\right) \geq a_k - a_l\right) \\
&= \mathbb{P}_k\left(\sum_{i=1}^{M}\xi_{kl}^{(i)} \geq a_k - a_l\right) \\
&= \mathbb{P}_k\left(\sum_{i=1}^{M}\xi_{kl}^{(i)} - \sum_{i=1}^{M}\mathbb{E}_k\left[\xi_{kl}^{(i)}\right] \geq (a_k - a_l) - \sum_{i=1}^{M}\mathbb{E}_k\left[\xi_{kl}^{(i)}\right]\right), \\
&= \mathbb{P}_k\left(\sum_{i=1}^{M}\xi_{kl}^{(i)} - \sum_{i=1}^{M}\mathbb{E}_k\left[\xi_{kl}^{(i)}\right] \geq \Lambda_{kl}^{(j)}\right) \qquad \text{(A.14)}
\end{aligned}$$

Note that $\left\{\xi_{kl}^{(i)}\right\}_{i\in[M]}$ are conditionally independent when given $\{y_j = k\}$, and they are bounded given the voting weights $\left\{\mu_{kh}^{(i)}\right\}$ are bounded. Therefore, we can apply the Hoeffding concentration inequality (Hoeffding, 1956) to further bound $\mathbb{P}_k\left(s_l^{(j)} \geq s_k^{(j)}\right)$.

We have that $\min_{l,k,h\in[L],k\neq l}\left\{\mu_{lh}^{(i)}-\mu_{kh}^{(i)}\right\} \leq \xi_{kl}^{(i)} \leq \max_{l,k,h\in[L],k\neq l}\left\{\mu_{lh}^{(i)}-\mu_{kh}^{(i)}\right\}$, and

$$\begin{aligned}
\sum_{i=1}^{M}\left[\max_{l,k,h\in[L],k\neq l}\left\{\mu_{lh}^{(i)}-\mu_{kh}^{(i)}\right\} - \min_{l,k,h\in[L],k\neq l}\left\{\mu_{lh}^{(i)}-\mu_{kh}^{(i)}\right\}\right]^2 &\leq \sum_{i=1}^{M}\left(2\max_{l,k,h\in[L],k\neq l}\left|\mu_{lh}^{(i)}-\mu_{kh}^{(i)}\right|\right)^2 \\
&= 4\Gamma^2
\end{aligned}$$

When $\Lambda_{kl}^{(j)} \geq \tau_{j,\min}\cdot\Gamma \geq 0$, by applying the Hoeffding inequality to (A.14), we have

$$\begin{aligned}
\mathbb{P}_k\left(s_l^{(j)} \geq s_k^{(j)}\right) &\leq \mathbb{P}_k\left(\sum_{i=1}^{M}\xi_{kl}^{(i)} - \sum_{i=1}^{M}\mathbb{E}_k\left[\xi_{kl}^{(i)}\right] \geq \Lambda_{kl}^{(j)}\right) \\
&\leq \exp\left(-\frac{2{\Lambda_{kl}^{(j)}}^2}{\sum_{i=1}^{M}\left[\max_{l,k,h\in[L],k\neq l}\left\{\mu_{lh}^{(i)}-\mu_{kh}^{(i)}\right\} - \min_{l,k,h\in[L],k\neq l}\left\{\mu_{lh}^{(i)}-\mu_{kh}^{(i)}\right\}\right]^2}\right) \\
&\leq \exp\left(-\frac{{\Lambda_{kl}^{(j)}}^2}{2\Gamma^2}\right) \qquad \text{(because of (A.15))} \\
&\leq \exp\left(-\frac{\tau_{j,\min}^2}{2}\right). \qquad \text{(based on the definition of } \tau_{j,\min}\text{)}
\end{aligned}$$

The right hand side of the last inequality does not depend on k, l or i, then

$$\mathbb{P}_k\left(\hat{y}_j \neq k\right) \leq \sum_{l\in[L],l\neq k}\mathbb{P}_k\left(s_l^{(j)} \geq s_k^{(j)}\right) \leq (L-1)\exp\left(-\frac{\tau_{j,\min}^2}{2}\right). \qquad \text{(A.15)}$$

Because the RHS does not depend on k, we have

$$\begin{aligned}\mathbb{P}(\hat{y}_j \neq y_j) &= \sum_{k\in[L]} \pi_k \mathbb{P}_k\left(\hat{y}_j \neq k\right) \\ &\leq (L-1)\exp\left(-\frac{\tau_{j,\min}^2}{2}\right)\left(\sum_{k\in[L]}\pi_k\right) \\ &= (L-1)\exp\left(-\frac{\tau_{j,\min}^2}{2}\right) \qquad \text{(A.16)}\end{aligned}$$

The Hoeffding inequality does not take the variance information of the independent random variables into account, thus a "stronger" concentration inequality can be applied when the fluctuation of $\xi_{kl}^{(i)}$ is available.

Note the definition of c and σ^2 are defined as

$$\begin{aligned}c &= \frac{1}{\Gamma}\cdot \max_{i\in[M],k,l,h\in[L],k\neq l} |\mu_{kh}^{(i)} - \mu_{lh}^{(i)}|, \\ \sigma^2 &= \frac{1}{\Gamma^2}\cdot \max_{j\in[N]}\max_{k,l\in[L],k\neq l} \sum_{i=1}^{M}\sum_{h=1}^{L} q_{ij}\left(\mu_{kh}^{(i)} - \mu_{lh}^{(i)}\right)^2 p_{kh}^{(i)}.\end{aligned}$$

The sum of the second moment of $\xi_{kl}^{(i)}$ can be bounded as

$$\sum_{i=1}^{M}\mathbb{E}_k\left[\left(\xi_{kl}^{(i)}\right)^2\right] = \sum_{i=1}^{M}\mathbb{E}_k\left[\left(\sum_{h=1}^{L}(\mu_{lh}^{(i)} - \mu_{kh}^{(i)})\mathrm{I}\left(Z_{ij}=h\right)\right)^2\right] = \sum_{i=1}^{M}\sum_{h=1}^{L} q_{ij}\left(\mu_{kh}^{(i)} - \mu_{lh}^{(i)}\right)^2 p_{kh}^{(i)} \leq \sigma^2\Gamma^2.$$

$\xi_{kl}^{(i)}$ can be bounded as $|\xi_{kl}^{(i)}| \leq \max_{i\in[M],h\in[L]}|\mu_{lh}^{(i)} - \mu_{kh}^{(i)}| = c\Gamma$.

By applying the Bernstein-type concentration inequality (Chung and Lu, 2010), Theorem 2.8) with that $\Lambda_{kl}^{(j)} \geq \tau_{j,\min}\Gamma \geq 0$,

$$\begin{aligned}\mathbb{P}_k\left(s_l^{(j)} \geq s_k^{(j)}\right) &\leq \mathbb{P}_k\left(\sum_{i=1}^{M}\xi_{kl}^{(i)} - \sum_{i=1}^{M}\mathbb{E}_k\left[\xi_{kl}^{(i)}\right] \geq \Lambda_{kl}^{(j)}\right) \\ &\leq \exp\left(-\frac{{\Lambda_{kl}^{(j)}}^2}{2\left(\sigma^2 + c\Gamma\Lambda_{kl}^{(j)}/3\right)}\right), \\ &\leq \exp\left(-\frac{\tau_{j,\min}^2}{2\left(\sigma^2 + c\tau_{j,\min}/3\right)}\right) \qquad \left(\text{because } \frac{\Lambda_{kl}^{(j)}}{\Gamma} \geq \tau_{j,\min} \geq 0\right), \qquad \text{(A.17)}\end{aligned}$$

where the RHS does not depend on k, l. Then, we have

$$\mathbb{P}_k\left(\hat{y}_j \neq k\right) \leq (L-1)\exp\left(-\frac{\tau_{j,\min}^2}{2\left(\sigma^2 + c\tau_{j,\min}/3\right)}\right).$$

Furthermore,

$$\mathbb{P}(\hat{y}_j \neq y_j) = \sum_{k\in[L]} \pi_k \mathbb{P}_k\left(\hat{y}_j \neq k\right) \leq (L-1)\exp\left(-\frac{\tau_{j,\min}^2}{2\left(\sigma^2 + c\tau_{j,\min}/3\right)}\right). \tag{A.18}$$

Combining inequalities (A.16) and (A.18) together, we can get the desired result in Proposition A.1.(1).

(2). Assuming that $\tau_{j,\max} \leq 0$, we want to show the upper bound for $\mathbb{P}(\hat{y}_j \neq y_j)$.

Using the same argument as in (1), we provide a lower bound for $\mathbb{P}_k\left(\hat{y}_j \neq k\right)$.

$$\begin{aligned}\mathbb{P}_k\left(\hat{y}_j \neq k\right) \geq \mathbb{P}_k\left(\bigcup_{l\in[L],l\neq k}\left\{s_l^{(j)} > s_k^{(j)}\right\}\right) &\geq \max_{l\in[L],l\neq k}\mathbb{P}_k\left(s_l^{(j)} > s_k^{(j)}\right)\\ &= 1 - \min_{l\in[L],l\neq k}\mathbb{P}_k\left(s_l^{(j)} \leq s_k^{(j)}\right)\end{aligned}$$

Given $\Lambda_{kl}^{(j)} \leq \tau_{j,\max}\cdot\Gamma \leq 0$, by applying the Hoeffding and the Bernstein inequality as in (1), we can obtain $\mathbb{P}_k\left(s_l^{(j)} \leq s_k^{(j)}\right) \leq \exp\left(-\frac{{\Lambda_{kl}^{(j)}}^2}{2\Gamma^2}\right) \leq \exp\left(-\frac{\tau_{j,\max}^2}{2}\right)$, and $\mathbb{P}_k\left(s_l^{(j)} \leq s_k^{(j)}\right) \leq \exp\left(-\frac{{\Lambda_{kl}^{(j)}}^2}{2(\sigma^2\Gamma^2 - c\Gamma\Lambda_{kl}^{(j)})}\right) \leq \exp\left(-\frac{\tau_{j,\max}^2}{2(\sigma^2 - c\tau_{j,\max})}\right)$. Since the RHS of the two inequalities do not depend on k or l, $\mathbb{P}_k\left(\hat{y}_j \neq k\right) \geq 1 - \min\left\{\exp\left(-\frac{\tau_{j,\max}^2}{2}\right), \exp\left(-\frac{\tau_{j,\max}^2}{2(\sigma^2 - c\tau_{j,\max})}\right)\right\}$.

With (A.9), we have $\mathbb{P}\left(\hat{y}_j \neq y_j\right) \geq 1 - \min\left\{\exp\left(-\frac{\tau_{j,\max}^2}{2}\right), \exp\left(-\frac{\tau_{j,\max}^2}{2(\sigma^2 - c\tau_{j,\max})}\right)\right\}$.

□

A.2 Proof of Theorem 2.3

Proof. Given $\sigma^2 \geq 0$ and $c > 0$, both functions $\exp\left(-\frac{t^2}{2}\right)$ and $\exp\left(-\frac{t^2}{2(\sigma^2 + ct/3)}\right)$ are monotonely increasing on $t \in [0, \infty)$. On the other hand, both functions $\exp\left(-\frac{t^2}{2}\right)$ and $\exp\left(-\frac{t^2}{2(\sigma^2 - ct/3)}\right)$ are monotonely decreasing on $t \in (-\infty, 0]$.

Given $t_1 \geq 0$, then $\tau_{j,\min} \geq t_1 \geq 0$. By Proposition A.1,

$$\begin{aligned}\frac{1}{N}\sum_{j=1}^{N}\mathbb{P}\left(\hat{y}_j \neq y_j\right) &\leq \frac{1}{N}\sum_{j=1}^{N}(L-1)\min\left\{\exp\left(-\frac{\tau_{j,\min}^2}{2}\right), \exp\left(-\frac{\tau_{j,\min}^2}{2\left(\sigma^2 + c\tau_{j,\min}/3\right)}\right)\right\}\\ &\leq \frac{L-1}{N}\sum_{j=1}^{N}\min\left\{\exp\left(-\frac{t_1^2}{2}\right), \exp\left(-\frac{t_1^2}{2\left(\sigma^2 + t_1/3\right)}\right)\right\}\\ &= (L-1)\min\left\{\exp\left(-\frac{t_1^2}{2}\right), \exp\left(-\frac{t_1^2}{2\left(\sigma^2 + t_1/3\right)}\right)\right\}.\end{aligned}$$

Thus, we have proved Theorem 2.3.(1).
With the same argument, we can straightforwardly prove Theorem 2.3.(2).

□

A.3 Proof of Theorem 2.1

So far, we have bounded the mean error rate, but we still need more tools for bounding the error rate in the practical case with high probability. The following lemma is another form of the Bernstein-Chernoff-Hoeffding theorem (Ngo, 2011).

Lemma A.2. ***(Bernstein-Chernoff-Hoeffding)*** *Let $\xi_i \in [0,1]$ be independent random variables where $\mathbb{E}\xi_i = p_i, i \in [n]$. Let $\bar{\xi} = \frac{1}{n}\sum_{i=1}^{n}\xi_i$ and $\bar{p} = \frac{1}{n}\sum_{i=1}^{n}p_i$. Then,*
(1) for any m such that $\bar{p} \leq \frac{m}{n} < 1$, $\mathbb{P}\left(\bar{\xi} > m/n\right) \leq e^{-nD(m/n||\bar{p})}$,
(2) for any m such that $0 < \frac{m}{n} \leq \bar{p}$, $\mathbb{P}\left(\bar{\xi} < m/n\right) \leq e^{-nD(m/n||\bar{p})}$.

The proof of Theorem 2.1 is as follows:

Proof. (Theorem 2.1)
Proof of Theorem 2.1 (1) Let $\mu = \frac{1}{N}\sum_{j=1}^{N}\mathbb{P}\left(\hat{y}_j \neq y_j\right)$. By Theorem 2.3.(1), we have that $\mu \leq (L-1)e^{-t_1^2/2} = (L-1)\phi(t_1)$. Assume $t_1 \geq \sqrt{2\ln\frac{L-1}{\epsilon}}$, then we can get $(L-1)\phi(t_1) \leq \epsilon$, which gives us $0 \leq \mu \leq (L-1)\phi(t_1) \leq \epsilon$. Then by the Bernstein-Chernoff-Hoeffding Theorem, i.e. Lemma A.2, we get $\mathbb{P}\left(\frac{1}{N}\sum_{j=1}^{N}\mathrm{I}\left(\hat{y}_j \neq y_j\right) > \epsilon\right) \leq e^{-N\mathrm{D}(\epsilon||\mu)} \leq e^{-N\mathrm{D}(\epsilon||(L-1)\phi(t_1))}$. Therefore, we have

$$\mathbb{P}\left(\frac{1}{N}\sum_{j=1}^{N}\mathrm{I}\left(\hat{y}_j \neq y_j\right) \leq \epsilon\right) \geq 1 - e^{-N\mathrm{D}(\epsilon||(L-1)\phi(t_1))}.$$

Proof of Theorem 2.1 (2) With the same argument as above, assuming $t_2 \leq -\sqrt{2\ln\frac{1}{1-\epsilon}}$, then $1 \geq \mu \geq 1 - \phi(t_2) \geq \epsilon$, which gives us

$$\mathbb{P}\left(\frac{1}{N}\sum_{j=1}^{N}\mathrm{I}\left(\hat{y}_j \neq y_j\right) \geq \epsilon\right) \geq 1 - e^{-N\mathrm{D}(\epsilon||1-\phi(t_2))}.$$

Thus, we have proved Theorem 2.1.

□

A.4 Proof of Theorem 2.2

Before proving Theorem 2.2, we are going to prove an important lemma for bounding the average of a group of independent Bernoulli random variables. The proof of this lemma relies on Hoeffding bounds and the Bernstein-Chernoff-Hoeffding theorem (Ngo, 2011).

Lemma A.3. *Suppose* $\forall j \in [N]$, $\xi_j \sim$ *Bernoulli*(p_j) *with* $p_j \in (0,1)$, *and* ξ_j*'s are independent of each other. Let* $\bar{\xi} = \frac{1}{N}\sum_{j=1}^{N} \xi_j$ *and* $\bar{p} = \mathbb{E}\bar{\xi} = \frac{1}{N}\sum_{j=1}^{N} p_j$ *Given any* $\epsilon, \delta \in (0,1)$*:*

(1) If $0 < \bar{p} \leq \frac{1}{1+\exp\left(\frac{1}{\epsilon}\left[H_e(\epsilon)+\frac{1}{N}\ln\frac{1}{\delta}\right]\right)}$, *then* $\mathbb{P}(\bar{\xi} \leq \epsilon) \geq 1-\delta$.

(2) If $\frac{1}{1+\exp\left(-\frac{1}{1-\epsilon}\left(H_e(\epsilon)+\frac{1}{N}\ln\frac{1}{\delta}\right)\right)} \leq \bar{p} < 1$, *then* $\mathbb{P}(\bar{\xi} \leq \epsilon) < \delta$.

Proof. For simplicity let's define $A = \left(H_e(\epsilon) + \frac{1}{N}\ln\frac{1}{\delta}\right)$

The proof of Lemma A.3.(1): We will finish the proof in several steps:

Assume $0 < \bar{p} \leq \frac{1}{1+\exp\left(\frac{1}{\epsilon}\left[H_e(\epsilon)+\frac{1}{N}\ln\frac{1}{\delta}\right]\right)}$.

Step 1. we want to show $\bar{p} < \epsilon$ *:*

$$\begin{aligned}
\exp\left(\frac{A}{\epsilon}\right) &= \exp\left(\frac{\epsilon\ln\frac{1}{\epsilon} + (1-\epsilon)\ln\frac{1}{1-\epsilon} + \frac{1}{N}\ln\frac{1}{\delta}}{\epsilon}\right) \\
&= \exp\left(\ln\frac{1}{\epsilon} + \frac{1-\epsilon}{\epsilon}\ln\frac{1}{1-\epsilon} + \frac{1}{N\epsilon}\ln\frac{1}{\delta}\right) \\
&> \exp\left(\ln\frac{1}{\epsilon}\right) \qquad (\text{because } \epsilon, \delta \in (0,1), N > 0) \\
&= \frac{1}{\epsilon}
\end{aligned}$$

$$\implies \quad 1 + \exp\left(\tfrac{A}{\epsilon}\right) > \tfrac{1}{\epsilon} \quad \implies \quad \tfrac{1}{1+\exp\left(\frac{A}{\epsilon}\right)} < \epsilon \ .$$

Since $0 < \bar{p} \leq \frac{1}{1+\exp(A/\epsilon)}$, then we have $\bar{p} < \epsilon$.

Step 2. We want to show $\mathbb{P}\left(\bar{\xi} \leq \epsilon\right) \geq 1 - e^{-N\cdot D(\epsilon||\bar{p})}$ *:*

This is obtained by the Bernstein-Chernoff-Hoeffding Theorem ((Ngo, 2011, McDiarmid, 1998)), which leads to:

$$\text{If } 0 < \bar{p} \leq \epsilon, \text{ then } \mathbb{P}(\bar{\xi} \leq \epsilon) \leq 1 - e^{-ND(\epsilon||\bar{p})} \tag{A.19}$$

$$\text{If } \epsilon \leq \bar{p} < 1, \text{ then } \mathbb{P}(\bar{\xi} \leq \epsilon) \leq e^{-ND(\epsilon||\bar{p})} \tag{A.20}$$

Since we have shown in step 1 that $\bar{p} < \epsilon$, then we can get the desired result in this step easily.

Step 3. We want to show $e^{-ND(\epsilon||\bar{p})} \leq \delta$ *:*

Note:

$$\begin{aligned} & e^{-ND(\epsilon||\bar{p})} \leq \delta \\ \iff \quad & D(\epsilon||\bar{p}) \geq \frac{1}{N}\ln\frac{1}{\delta} \\ \iff \quad & \ln\frac{\epsilon^{\epsilon}(1-\epsilon)^{1-\epsilon}}{\bar{p}^{\epsilon}(1-\bar{p})^{1-\epsilon}} \geq \ln\left(\frac{1}{\delta}\right)^{\frac{1}{N}} \\ \iff \quad & \bar{p}^{\epsilon}(1-\bar{p})^{1-\epsilon} \leq \exp\left(-\left(H_e(\epsilon)+\frac{1}{N}\ln\frac{1}{\delta}\right)\right) = e^{-A} \end{aligned} \tag{A.21}$$

From the condition we have,

$$\bar{p} \leq \frac{1}{1+\exp(A/\epsilon)} \qquad \Longrightarrow \qquad \left(\frac{\bar{p}}{1-\bar{p}}\right)^{\epsilon} \leq e^{-A}$$

Note that $\bar{p}^{\epsilon}(1-\bar{p})^{1-\epsilon} = \left(\frac{\bar{p}}{1-\bar{p}}\right)^{\epsilon}(1-\bar{p}) < \left(\frac{\bar{p}}{1-\bar{p}}\right)^{\epsilon}$ (because $1-\bar{p}<1$)

$\Longrightarrow$ Inequality (A.21) holds: $\bar{p}^{\epsilon}(1-\bar{p})^{1-\epsilon} \leq e^{-A} \quad \Longrightarrow \quad e^{-ND(\epsilon||\bar{p})} \leq \delta$

By step 2 and step 3, we can easily get that if $\bar{p} \leq \frac{1}{1+e^{A/\epsilon}}$, then $\mathbb{P}(\bar{\xi} \leq \epsilon) \geq 1-\delta$, which is the results we want.

The proof of Lemma A.3.(2): We will also finish the proof in several steps:
Assume

$$\frac{1}{1+\exp\left(-\frac{1}{1-\epsilon}\left(H_e(\epsilon)+\frac{1}{N}\ln\frac{1}{\delta}\right)\right)} \leq \bar{p} < 1$$

Step 1. We want to show $\bar{p} > \epsilon$

We show it by proving as follows:

$$\begin{aligned} & \frac{1}{1+\exp\left(-\frac{1}{1-\epsilon}\left(H_e(\epsilon)+\frac{1}{N}\ln\frac{1}{\delta}\right)\right)} > \epsilon \\ \iff \quad & 1+\exp\left(-\frac{1}{1-\epsilon}\left(H_e(\epsilon)+\frac{1}{N}\ln\frac{1}{\delta}\right)\right) < \frac{1}{\epsilon} \\ \iff \quad & \left(H_e(\epsilon)+\frac{1}{N}\ln\frac{1}{\delta}\right) > (1-\epsilon)\ln\frac{\epsilon}{1-\epsilon} \\ \iff \quad & \epsilon\ln\frac{1}{\epsilon}+(1-\epsilon)\ln\frac{1}{1-\epsilon}+\frac{1}{N}\ln\frac{1}{\delta} > (1-\epsilon)\ln\frac{1}{1-\epsilon}-(1-\epsilon)\ln\frac{1}{\epsilon} \\ \iff \quad & \ln\frac{1}{\epsilon}+\frac{1}{N}\ln\frac{1}{\delta} > 0 \end{aligned}$$

which is of course true since $\epsilon, \delta \in (0,1)$. Therefore, we have proved $\bar{p} > \epsilon$.

Step 2. We want to show $\mathbb{P}(\bar{\xi} \leq \epsilon) \leq e^{-ND(\epsilon||\bar{p})}$

By the Bernstein-Chernoff-Hoeffding Theorem, since $\epsilon < \bar{p} = \mathbb{E}\bar{\xi}$ and $\xi_j \sim \text{Bernoulli}(p_j)$ independently, we can directly prove this step.

Step 3. we want to show $e^{-ND(\epsilon||\bar{p})} < \delta$

Note that

$$\begin{aligned} e^{-ND(\epsilon||\bar{p})} < \delta &\iff D(\epsilon||\bar{p}) > \frac{1}{N}\ln\frac{1}{\delta} \\ &\iff \bar{p}^{\epsilon}(1-\bar{p})^{1-\epsilon} < \exp\left(-\left(H_e(\epsilon) + \frac{1}{N}\ln\frac{1}{\delta}\right)\right) = e^{-A} \end{aligned} \tag{A.22}$$

From the condition we have

$$\begin{aligned} \bar{p} \geq \frac{1}{1+\exp -\frac{A}{1-\epsilon}} &\implies \frac{1-\bar{p}}{\bar{p}} \leq e^{-\frac{A}{1-\epsilon}} \\ &\implies \left(\frac{1-\bar{p}}{\bar{p}}\right)^{1-\epsilon} \leq \exp\left(-\left(H_e(\epsilon) + \frac{1}{N}\ln\frac{1}{\delta}\right)\right) \end{aligned} \tag{A.23}$$

And note that

$$\bar{p}^{\epsilon}(1-\bar{p})^{1-\epsilon} = \left(\frac{1-\bar{p}}{\bar{p}}\right)^{1-\epsilon} \cdot \bar{p} < \left(\frac{1-\bar{p}}{\bar{p}}\right)^{1-\epsilon} \qquad (\because \bar{p} < 1) \tag{A.24}$$

By combining inequalities (A.23) and (A.24), we can prove inequality (A.22). Thus we obtained $e^{-ND(\epsilon||\bar{p})} < \delta$. Finally, by step 2 and 3, we get : if $\bar{p} \geq \frac{1}{1+\exp\left(-\frac{1}{1-\epsilon}\left(H_e(\epsilon)+\frac{1}{N}\ln\frac{1}{\delta}\right)\right)}$, then $\mathbb{P}\left(\bar{\xi} \leq \epsilon\right) < \delta$

□

Now, we are going to prove Theorem 2.2 with the results we obtained in Lemma A.3

Proof. **of Theorem 2.2**

Let $\zeta_j = \mathrm{I}\left(\hat{y}_j \neq y_j\right) \sim \text{Bernoulli}(1-\theta_j)$ and let $\bar{p} = \mathbb{E}\bar{\zeta} = \frac{1}{N}\sum_{j=1}^{N}\mathbb{E}\zeta_j = 1-\bar{\theta}$.

The proof of Theorem 2.2.(1):

Assume that $t_1 \geq \sqrt{2\ln\left[(L-1)C(\epsilon,\delta)\right]}$, where $C(\epsilon,\delta) = 1+\exp\left(\frac{1}{\epsilon}\left[H_e(\epsilon)+\frac{1}{N}\ln\frac{1}{\delta}\right]\right)$, then $t_1 \geq 0$.

By Theorem 2.3, we have

$$\bar{\theta} = 1 - \frac{1}{N}\sum_{j=1}^{N}\mathbb{P}\left(\hat{y}_j \neq y_j\right) \;\geq\; 1-(L-1)e^{-\frac{t_1^2}{2}} \tag{A.25}$$

Let $A = \left(H_e(\epsilon) + \frac{1}{N}\ln\frac{1}{\delta}\right)$, then

$$\begin{aligned} & t_1 \geq \sqrt{2\ln\left[(L-1)C(\epsilon,\delta)\right]} = \sqrt{2\ln\left[(L-1)(1+\exp(A/\epsilon))\right]} \\ \Longrightarrow \quad & (L-1)\exp\left(-\frac{t_1^2}{2}\right) \leq \frac{1}{1+\exp\left(\frac{A}{\epsilon}\right)} \\ \Longrightarrow \quad & \bar{\theta} \geq 1-(L-1)\exp\left(-\frac{t_1^2}{2}\right) \geq 1-\frac{1}{1+\exp\left(\frac{A}{\epsilon}\right)} \qquad (\because (A.25)) \\ \Longrightarrow \quad & 1-\bar{\theta} \leq \frac{1}{1+\exp\left(\frac{A}{\epsilon}\right)}. \end{aligned} \tag{A.26}$$

By inequality (A.26) and by Lemma A.3, we have

$$\mathbb{P}(\bar{\zeta} \leq \epsilon) \geq 1-\delta$$

which is to say,

$$\mathbb{P}\left(\frac{1}{N}\sum_{j=1}^{N}\mathrm{I}\left(\hat{y}_j \neq y_j\right) \leq \epsilon\right) \;\geq\; 1-\delta$$

Therefore, we have proved (1).

The proof of Theorem 2.2.(2):

Assume $t_2 \leq -\sqrt{2\ln G} \leq 0$, where $G = 1+\exp\left(\frac{1}{1-\epsilon}\left(H_e(\epsilon)+\frac{1}{N}\ln\frac{1}{\delta}\right)\right)$. Then by Theorem 2.3.(2), we have

$$\bar{\theta} = \frac{1}{N}\sum_{j=1}^{N}\mathbb{P}(\hat{y}_j = y_j) \leq \exp\left(-\frac{t_2^2}{2}\right) \tag{A.27}$$

From the conditions in (2)

$$\begin{aligned} & t_2 \leq -\sqrt{2\ln\left(1+\exp\left(\frac{1}{1-\epsilon}\left[H_e(\epsilon)+\frac{1}{N}\ln\frac{1}{\delta}\right]\right)\right)} \\ \Longrightarrow \quad & \exp\left(-\frac{t_2^2}{2}\right) \leq \frac{1}{1+\exp\left(\frac{A}{1-\epsilon}\right)} \\ \Longrightarrow \quad & 1-\bar{\theta} \geq 1-\exp\left(-\frac{t_2^2}{2}\right) \geq 1-\frac{1}{1+\exp\left(\frac{A}{1-\epsilon}\right)} = \frac{1}{1+\exp\left(-\frac{A}{1-\epsilon}\right)} \end{aligned}$$

By Lemma A.3.(2), we have $\mathbb{P}(\bar{\zeta} \leq \epsilon) < \delta$ which implies the desired result.

□

Appendix B

Proofs for Chapter 3

B.1 Preparations before proving Theorem 3.3

In the proof of this result, we focus on $\mathbb{P}(T_{ij} = 1) = q = 1, \forall i \in [M], j \in [N]$, i.e., every worker labels any item with probability q. Meanwhile, we assume $L = 2$, and the label set $[L] \doteq \{\pm 1\}$. It's not hard to generalize our results to $q \in (0, 1]$ and general L case, which is more practical, but the bound will be much more complicated. We omit it here for clarity.

The prediction of the one-step Weighted Majority Voting for the jth item is

$$\hat{y}_j^{wmv} = \text{sign}\left(\sum_{i=1}^{M}(2\hat{w}_i - 1)Z_{ij}\right), \tag{B.1}$$

where $\hat{w}_i$ is the estimated worker accuracy by taking the output from majority voting as "true" labels. That is to say,

$$\hat{w}_i = \frac{1}{N}\sum_{j=1}^{N}\mathrm{I}\left(Z_{ij} = \hat{y}_j^{mv}\right), \qquad \text{where} \qquad \hat{y}_j^{mv} = \text{sign}\left(\sum_{i=1}^{M} Z_{ij}\right). \tag{B.2}$$

Note that the average accuracy of workers is $\bar{w} = \frac{1}{M}\sum_{i=1}^{M} w_i$.

In fact, Theorem 3.3 is a direct implication by the following result.

Proposition B.1. *If $\bar{w} \geq \frac{1}{2} + \frac{1}{M} + \sqrt{\frac{(M-1)\ln 2}{2M^2}}$, then the mean error rate of one-step Weighted Majority Voting for the jth item will be:*

$$\mathbb{P}\left(\hat{y}_j^{wmv} \neq y_j\right) \leq \exp\left(-\frac{8MN^2\tilde{\sigma}^4(1-\eta)^2}{M^2N + (M+N)^2}\right), \tag{B.3}$$

where $\tilde{\sigma}$ and η is as defined in Theorem 3.3 .

In the next section, we will focus on proving this result first, and then Theorem 3.3 can be obtained directly.

Before we present our proof for Proposition B.1, we need to prove several useful results which we can use later.

With the same notations as we have used for proving Theorem 1, we simplified some notations as follows for convenience and clearance:

$$\mathbb{P}_+^{(j)}(\ \cdot\ ) \doteq \mathbb{P}(\ \cdot\ |y_j=+1) \quad \text{and} \quad \mathbb{P}_-^{(j)}(\ \cdot\ ) \doteq \mathbb{P}(\ \cdot\ |y_j=-1), \tag{B.4}$$

$$\mathbb{E}_+^{(j)}[\ \cdot\ ] \doteq \mathbb{E}[\ \cdot\ |y_j=+1] \quad \text{and} \quad \mathbb{E}_-^{(j)}[\ \cdot\ ] \doteq \mathbb{E}[\ \cdot\ |y_j=-1], \tag{B.5}$$

where " $\cdot$ " denotes any event belonging to the σ-algebra generated by Z.

The following lemma enable us to bound the probability of where the majority vote of the jth item agrees with the label given by the ith worker given the true label and Z_{ij}.

Lemma B.2. *$\forall j \in [N]$ and $\forall i \in [M]$, we have*
(1) if $\bar{w} > \frac{1}{2}$, then

$$\mathbb{P}\left(\hat{y}_j^{mv}=+1|y_j=+1, Z_{ij}=+1\right) \geq 1-\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}+\frac{1-w_i}{M})^2}{M-1}\right), \tag{B.6}$$

and the same bound holds for $\mathbb{P}\left(\hat{y}_j^{mv}=-1|y_j=-1, Z_{ij}=-1\right)$.
(2) if $\bar{w} \geq \frac{1}{2}+\frac{1}{M}$, then

$$\mathbb{P}\left(\hat{y}_j^{mv}=-1|y_j=+1, Z_{ij}=-1\right) \leq \exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}-\frac{w_i}{M})^2}{M-1}\right), \tag{B.7}$$

and the same bound holds for $\mathbb{P}\left(\hat{y}_j^{mv}=+1|y_j=-1, Z_{ij}=+1\right)$.

Proof. (1) Notice that for any $i \in [M]$ and $j \in [N]$, given y_j, Z_{ij} is independent of $\{Z_{lj}\}_{l\neq i}$, $\mathbb{E}_+^{(j)} Z_{lj} = 2w_l - 1$ and $\mathbb{E}_-^{(j)} Z_{lj} = -(2w_l-1)$, then

$$\sum_{l\neq i} \mathbb{E}_+^{(j)} Z_{lj} + 1 = \sum_{l\neq i}(2w_l-1)+1 = 2M\left(\bar{w}-\frac{1}{2}+\frac{1-w_i}{M}\right) > 0, \tag{B.8}$$

since $\bar{w}-\frac{1}{2}+\frac{1-w_i}{M} > \bar{w}-\frac{1}{2} > 0$. Therefore, we can apply the Hoeffding inequality to get:

$$\begin{aligned}
\mathbb{P}\left(\hat{y}_j^{mv}=+1|y_j=+1, Z_{ij}=+1\right) &= \mathbb{P}_+^{(j)}\left(\hat{y}_j^{mv}=+1|Z_{ij}=+1\right) \\
&= \mathbb{P}_+^{(j)}\left(\sum_{l=1}^{M} Z_{lj} > 0 \middle| Z_{ij}=+1\right) \\
&= \mathbb{P}_+^{(j)}\left(\sum_{l\neq i} Z_{lj} - \sum_{l\neq i}\mathbb{E}_+^{(j)} Z_{lj} > -(\sum_{l\neq i}\mathbb{E}_+^{(j)} Z_{lj}+1)\right) \\
&\geq 1-\exp\left(-\frac{[\sum_{l\neq i}\mathbb{E}_+^{(j)} Z_{lj}+1]^2}{2(M-1)}\right) \qquad \text{(by Hoeffding)} \\
&= 1-\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}+\frac{1-w_i}{M})^2}{M-1}\right) \qquad \text{(by (B.8))}
\end{aligned}$$

Note that with the same argument,

$$\begin{aligned}\mathbb{P}(\hat{y}_j^{mv} = -1 | y_j = -1, Z_{ij} = -1) &= \mathbb{P}_-^{(j)}\left(\sum_{l=1}^{M} Z_{lj} < 0 | Z_{ij} = -1\right) \\ &= \mathbb{P}_-^{(j)}\left(\sum_{l\neq i} Z_{lj} - \sum_{l\neq i}\mathbb{E}_-^{(j)} Z_{lj} < -\sum_{l\neq i}\mathbb{E}_-^{(j)} Z_{lj} + 1\right) \\ &\geq 1 - \exp\left(-\frac{2M^2(\bar{w} - \frac{1}{2} + \frac{1-w_i}{M})^2}{M-1}\right),\end{aligned}$$

provided that $-\sum_{l\neq i}\mathbb{E}_-^{(j)} Z_{lj} + 1 = 2M\left(\bar{w} - \frac{1}{2} + \frac{1-w_i}{M}\right) > 0$, which is satisfied by the assumption $\bar{w} > \frac{1}{2}$.

(2) With the same argument as above, notice that $\sum_{l\neq i}\mathbb{E}_+^{(j)} Z_{lj} - 1 = 2M\left(\bar{w} - \frac{1}{2} - \frac{w_i}{M}\right) \geq 0$ because $\bar{w} \geq \frac{1}{2} + \frac{1}{M}$.

By applying the Hoeffding inequality:

$$\begin{aligned}\mathbb{P}\left(\hat{y}_j^{mv} = -1 | y_j = +1, Z_{ij} = -1\right) &= \mathbb{P}_+^{(j)}(\sum_{l\neq i} Z_{lj} < 0 | Z_{ij} = -1) \\ &= \mathbb{P}_+^{(j)}\left(\sum_{l\neq i} Z_{lj} - \sum_{l\neq i}\mathbb{E}_+^{(j)} Z_{lj} < -(\sum_{l\neq i}\mathbb{E}_+^{(j)} Z_{lj} - 1)\right) \\ &\leq \exp\left(-\frac{[\sum_{l\neq i}\mathbb{E}_+^{(j)} Z_{lj} - 1]^2}{2(M-1)}\right) \\ &= \exp\left(-\frac{2M^2(\bar{w} - \frac{1}{2} - \frac{w_i}{M})^2}{M-1}\right)\end{aligned}$$

Following the same argument, we can show that the same bound holds for $\mathbb{P}\left(\hat{y}_j^{mv} = +1 | y_j = -1, Z_{ij} = +1\right)$.

□

Our next lemma will bound the probability that the label of item j given by worker i agrees with majority voting.

Lemma B.3. *Given $\bar{w} \geq \frac{1}{2} + \frac{1}{M}$, then $\forall j \in [N]$, we have*

$$w_i - \xi_i^{(1)} \leq \mathbb{P}(Z_{ij} = \hat{y}_j^{mv} \mid y_j) \leq w_i + \xi_i^{(2)}, \tag{B.9}$$

where $\xi_i^{(1)} = w_i \exp\left(-\frac{2M^2(\bar{w} - \frac{1}{2} + \frac{1-w_i}{M})^2}{M-1}\right)$ and $\xi_i^{(2)} = (1-w_i)\exp\left(-\frac{2M^2(\bar{w} - \frac{1}{2} - \frac{w_i}{M})^2}{M-1}\right)$ Furthermore, we have

$$w_i - \xi_i^{(1)} \leq \mathbb{P}(Z_{ij} = \hat{y}_j^{mv}) \leq w_i + \xi_i^{(2)}, \tag{B.10}$$

Remark: This result implies that under mild conditions, the probability that the label of the jth item given by the ith worker matches the majority vote will be close to w_i, i.e., the accuracy of this worker. As the number of workers increase, it will be closer and closer. Intuitively, this makes sense since if M is large, majority voting will be close to the true label if $\bar{w} > 0.5$.

Proof.

$$\mathbb{P}(Z_{ij} = \hat{y}_j^{mv}) = \pi \mathbb{P}_+^{(j)}(Z_{ij} = \hat{y}_j^{mv}) + (1-\pi)\mathbb{P}_-^{(j)}(Z_{ij} = \hat{y}_j^{mv}). \tag{B.11}$$

$$\mathbb{P}_+^{(j)}(Z_{ij} = \hat{y}_j^{mv}) \quad = \quad w_i \mathbb{P}_+^{(j)}(\hat{y}_j^{mv} = +1 | Z_{ij} = +1) + (1-w_i)\mathbb{P}_+^{(j)}(\hat{y}_j^{mv} = +1 | Z_{ij} = -1).$$

Applying $\mathbb{P}_+^{(j)}(\hat{y}_j^{mv} = +1 | Z_{ij} = +1) \geq 1 - \exp\left(-\frac{2M^2(\bar{w} - \frac{1}{2} + \frac{1-w_i}{M})^2}{M-1}\right)$ from Lemma B.2.(1) and $(1-\pi)\mathbb{P}_-^{(j)}(Z_{ij} = \hat{y}_j^{mv}) \geq 0$ we can get

$$\mathbb{P}_+^{(j)}(Z_{ij} = \hat{y}_j^{mv}) \geq w_i - w_i \exp\left(-\frac{2M^2(\bar{w} - \frac{1}{2} + \frac{1-w_i}{M})^2}{M-1}\right). \tag{B.12}$$

By applying $\mathbb{P}_+^{(j)}(\hat{y}_j^{mv} = +1 | Z_{ij} = +1) \leq 1$ and $\mathbb{P}_-^{(j)}(Z_{ij} = \hat{y}_j^{mv}) \leq \exp\left(-\frac{2M^2(\bar{w} - \frac{1}{2} - \frac{w_i}{M})^2}{M-1}\right)$ from Lemma B.2.(2), we can get

$$\mathbb{P}_+^{(j)}(Z_{ij} = \hat{y}_j^{mv}) \leq w_i + (1-w_i) \exp\left(-\frac{2M^2(\bar{w} - \frac{1}{2} - \frac{w_i}{M})^2}{M-1}\right). \tag{B.13}$$

Similarly, we can obtain the same bounds for $\mathbb{P}_-^{(j)}(Z_{ij} = \hat{y}_j^{mv})$, i.e.,

$$\mathbb{P}_-^{(j)}(Z_{ij} = \hat{y}_j^{mv}) \geq w_i - w_i \exp\left(-\frac{2M^2(\bar{w} - \frac{1}{2} + \frac{1-w_i}{M})^2}{M-1}\right), \tag{B.14}$$

$$\mathbb{P}_-^{(j)}(Z_{ij} = \hat{y}_j^{mv}) \leq w_i + (1-w_i) \exp\left(-\frac{2M^2(\bar{w} - \frac{1}{2} - \frac{w_i}{M})^2}{M-1}\right) \tag{B.15}$$

Since $\mathbb{P}_+^{(j)}(Z_{ij} = \hat{y}_j^{mv})$ and $\mathbb{P}_-^{(j)}(Z_{ij} = \hat{y}_j^{mv})$ have the same bounds, and

$$\mathbb{P}(Z_{ij} = \hat{y}_j^{mv} \mid y_j) = \mathrm{I}(y_j = 1)\, \mathbb{P}_+^{(j)}(Z_{ij} = \hat{y}_j^{mv}) + \mathrm{I}(y_j = -1)\, \mathbb{P}_-^{(j)}(Z_{ij} = \hat{y}_j^{mv}),$$

then (B.9) holds. Furthermore, since

$$\mathbb{P}(Z_{ij} = \hat{y}_j^{mv}) = \pi \mathbb{P}_+^{(j)}(Z_{ij} = \hat{y}_j^{mv}) + (1-\pi)\mathbb{P}_-^{(j)}(Z_{ij} = \hat{y}_j^{mv}),$$

then (B.12) to (B.15) implies (B.10) .

□

The next lemma will be crucial for applying concentration measure results to bound $\mathbb{P}(\hat{y}_j^{wmv} \neq y_j \mid \{y_k\}_{k=1}^N)$.

For measuring the fluctuation of a function $g_j : \{0, \pm 1\}^{M \times N} \to \mathbb{R}$ if we change one entry of the data matrix, we define a quantity as follows:

$$d_{i^\star j^\star}^{(j)} \doteq \inf \{d : |g_j(Z) - g_j(Z')| \leq d, \text{where } Z \text{ and } Z' \text{ only differ on } (i^\star, j^\star)\}. \tag{B.16}$$

The constraints Z and Z' only differ on $(i^\star, j^\star)$, which means that $Z'_{i^\star j^\star}$ is a independent copy of $Z_{i^\star j^\star}$, and $Z_{ij} = Z'_{ij}$ for $(i,j) \neq (i^\star, j^\star)$.

Lemma B.4. *Let $g_j(Z) \doteq \sum_{i=1}^M (2\hat{w}_i - 1) Z_{ij}$, $\forall j \in [N]$, where Z is the data matrix and $\hat{w}_i$ is as defined in (B.2), with $d_{i^\star j^\star}^{(j)}$ defined in (B.16), we have*
(1)if $j^\star \neq j$, then $d_{i^\star j^\star}^{(j)} \leq \frac{2(M-1)}{N}$;
(2)if $j^\star = j$, then $d_{i^\star j^\star}^{(j)} \leq \frac{2(M-1)}{N} + 2$.

Proof. Since $Z'_{i^\star j^\star}$ is an independent copy of $Z_{i^\star j^\star}$, $Z'_{i^\star j^\star} = Z_{i^\star j^\star}$ or $Z'_{i^\star j^\star} = -Z_{i^\star j^\star}$. When $Z'_{i^\star j^\star} = Z_{i^\star j^\star}$, $Z' = Z$, then of course $|g_j(Z) - g_j(Z')| = 0$, which satisfies the inequality trivially.

Next, we focus on the non-trivial case $Z'_{i^\star j^\star} = -Z_{i^\star j^\star}$.

Note that $Z_{ij} = Z'_{ij}$ when $(i,j) \neq (i^\star, j^\star)$. Let $\hat{y}_j^{mv}$ be the majority voting of the jth column of Z, and $\hat{y}_j^{mv'}$ be the majority voting by the jth column of Z'.

Recall that $|g_j(Z) - g_j(Z')| = |\sum_{i=1}^M (2\hat{w}_i - 1) Z_{ij} - \sum_{i=1}^M (2\hat{w}'_i - 1) Z'_{ij}|$.

If $j^\star \neq j$, then $Z_{ij} = Z'_{ij}$, $\forall i \in [M]$, so,

$$|g_j(Z) - g_j(Z')| = |\sum_{i=1}^M (2\hat{w}_i - 2\hat{w}'_i) Z_{ij}| \leq 2 \sum_{i=1}^M |(\hat{w}_i - \hat{w}'_i) Z_{ij}| = 2 \sum_{i=1}^M |\hat{w}_i - \hat{w}'_i|. \tag{B.17}$$

If $j^\star = j$, we have $Z_{ij^\star} = Z'_{ij^\star}$ for $i \neq i^\star$, and $Z_{i^\star j^\star} = -Z'_{i^\star j^\star}$, then,

$$\begin{aligned} |g_j(Z) - g_j(Z')| &= |\sum_{i \neq i^\star} 2(\hat{w}_i - \hat{w}'_i) Z_{ij} + 2(\hat{p}_{i^\star} + \hat{p}'_{i^\star} - 1) Z_{i^\star j^\star}| \\ &\leq 2 \sum_{i \neq i^\star} |\hat{w}_i - \hat{w}'_i| + 2|\hat{p}_{i^\star} + \hat{p}'_{i^\star} - 1| \end{aligned} \tag{B.18}$$

We can see that the difference $g_j(Z) - g_j(Z')$ depends heavily on the two quantities $|\hat{w}_i - \hat{w}'_i|$ and $|\hat{p}_{i^\star} + \hat{p}'_{i^\star} - 1|$.

Next we bound $|\hat{w}_i - \hat{w}'_i|$:

$$|\hat{w}_i - \hat{w}'_i| = |\frac{1}{N} \sum_{i=1}^M \left(\mathrm{I}\left(Z_{ik} = \hat{y}_k^{mv}\right) - \mathrm{I}\left(Z'_{ik} = \hat{y}_k^{mv'}\right) \right)| = \frac{1}{N} |\mathrm{I}\left(Z_{ij^\star} = \hat{y}_{j^\star}^{mv}\right) - \mathrm{I}\left(Z'_{ij^\star} = \hat{y}_{j^\star}^{mv'}\right)|,$$

because $Z_{ik} = Z'_{ik}$ and $\hat{y}_k^{mv} = \hat{y}_k^{mv'}$ if $k \neq j^\star$.

(a). If $\hat{y}_{j^\star}^{mv} = \hat{y}_{j^\star}^{mv'}$, then by (B.19)

$$|\hat{w}_i - \hat{w}_i'| = \begin{cases} 0 & \text{if } i \neq i^\star, \\ \frac{1}{N} & \text{if } i = i^\star. \end{cases}$$

In this case,

$$\sum_{i=1}^{M} |\hat{w}_i - \hat{w}_i'| \;=\; \frac{1}{N}, \quad \text{and} \quad \sum_{i \neq i^\star} |\hat{w}_i - \hat{w}_i'| = 0. \tag{B.19}$$

(b). If $\hat{y}_{j^\star}^{mv} \neq \hat{y}_{j^\star}^{mv'}$, then by (B.19)

$$|\hat{w}_i - \hat{w}_i'| = \begin{cases} \frac{1}{N} & \text{if } i \neq i^\star, \\ 0 & \text{if } i = i^\star. \end{cases}$$

In this case,

$$\sum_{i=1}^{M} |\hat{w}_i - \hat{w}_i'| \;=\; \frac{M-1}{N}, \quad \text{and} \quad \sum_{i \neq i^\star} |\hat{w}_i - \hat{w}_i'| = \frac{M-1}{N}. \tag{B.20}$$

Now, we are going to bound $|\hat{p}_{i^\star} + \hat{p}_{i^\star}' - 1|$:

$$\begin{aligned} |\hat{p}_{i^\star} + \hat{p}_{i^\star}' - 1| \;&=\; \frac{1}{N} \Big| \sum_{k=1}^{L} \left(\mathrm{I}\left(Z_{i^\star k} = \hat{y}_k^{mv}\right) + \mathrm{I}\left(Z_{i^\star k}' = \hat{y}_k^{mv'}\right) - 1 \right) \Big| \\ &\leq\; \frac{1}{N} \sum_{k=1}^{L} \left| \mathrm{I}\left(Z_{i^\star k} = \hat{y}_k^{mv}\right) + \mathrm{I}\left(Z_{i^\star k}' = \hat{y}_k^{mv'}\right) - 1 \right|. \end{aligned} \tag{B.21}$$

(c). If $\hat{y}_{j^\star}^{mv} = \hat{y}_{j^\star}^{mv'}$, then

$$\left| \mathrm{I}\left(Z_{i^\star k} = \hat{y}_k^{mv}\right) + \mathrm{I}\left(Z_{i^\star k}' = \hat{y}_k^{mv'}\right) - 1 \right| = \begin{cases} 1 & \text{if } k \neq j^\star, \\ 0 & \text{if } k = j^\star. \end{cases}$$

So in this case, $|\hat{p}_{i^\star} + \hat{p}_{i^\star}' - 1| = \frac{N-1}{N}$.

(d). If $\hat{y}_{j^\star}^{mv} \neq \hat{y}_{j^\star}^{mv'}$, then

$$\left| \mathrm{I}\left(Z_{i^\star k} = \hat{y}_k^{mv}\right) + \mathrm{I}\left(Z_{i^\star k}' = \hat{y}_k^{mv'}\right) - 1 \right| = \begin{cases} 1 & \text{if } k \neq j^\star, \\ 1 & \text{if } k = j^\star. \end{cases}$$

So in this case, $|\hat{p}_{i^\star} + \hat{p}_{i^\star}' - 1| = 1$. Putting together all the results above, if $Z_{i^\star j^\star} \neq Z_{i^\star j^\star}'$, then we have,

(1') If $\hat{y}_{j^\star}^{mv} = \hat{y}_{j^\star}^{mv'}$,

$$|g_j(Z) - g_j(Z')| \leq \begin{cases} \frac{2}{N} & \text{if } j^\star \neq j, \\ \frac{2(N-1)}{N} & \text{if } j^\star = j. \end{cases}$$

(2') If $\hat{y}_{j^\star}^{mv} \neq \hat{y}_{j^\star}^{mv'}$,

$$|g_j(Z) - g_j(Z')| \leq \begin{cases} \frac{2(M-1)}{N} & \text{if } j^\star \neq j, \\ \frac{2(N-1)}{N} + 2 & \text{if } j^\star = j. \end{cases}$$

The upper bound in the case when $\hat{y}_j^{mv} \neq \hat{y}_j^{mv'}$ is also an upper bound for the case when $\hat{y}_j^{mv} = \hat{y}_j^{mv'}$. By the definition of $d_{i^\star j^\star}^{(j)}$ and noting that Z can only take a finite number of values, we have

$$d_{i^\star j^\star}^{(j)} \leq \begin{cases} \frac{2(M-1)}{N} & \text{if } j^\star \neq j, \\ \frac{2(N-1)}{N} + 2 & \text{if } j^\star = j. \end{cases}$$

□

Remark: $d_{i^\star j^\star}^{(j)}$ is the smallest upper bound on the difference between $g_j(Z)$ and $g_j(Z')$. From the proof, we can see that the bound we get is achievable, thus the bounds are tight. This result basically says that if we change only one entry of the data matrix, the fluctuation of the prediction score function of one-step Weighted Majority Voting, i.e., $g_j(Z)$, will be large if M increases and will decrease if N increases.

B.2 Proofs of Proposition B.1 and Theorem 3.3

In this section, we will use the lemmas we obtained in the last section to prove Proposition B.1 and then easily derive the bounds on the expected error rate of one-step WMV from it.

Proof. of Proposition B.1

$$\mathbb{P}(\hat{y}_j^{wmv} \neq y_j) = \sum_{y_1,\cdots,y_N} \mathbb{P}\left(\hat{y}_j^{wmv} \neq y_j \mid y_1, \cdots, y_N\right) \cdot \mathbb{P}(y_1, \cdots, y_N)$$

If we can get an unified upper bound on $\mathbb{P}\left(\hat{y}_j^{wmv} = y_j \mid y_1, \cdots, y_N\right)$, say B, which is independent of $\{y_k\}_{k=1}^N$, then this bound will also be an upper bound of $\mathbb{P}(\hat{y}_j^{wmv} \neq y_j)$ since

$$\sum_{y_1,\cdots,y_N} \mathbb{P}\left(\hat{y}_j^{wmv} \neq y_j \mid y_1, \cdots, y_N\right) \cdot \mathbb{P}(y_1, \cdots, y_N) \leq \sum_{y_1,\cdots,y_N} B \cdot \mathbb{P}(y_1, \cdots, y_N) = B.$$

Note that the Z_{ij}'s are not independent of each other unless conditioned on all the true labels of $y_1, \cdots, y_N$. For convenience of notation, we define the conditional probability and conditional expectation as follows:

$$\widetilde{\mathbb{P}}_{+}^{(j)}(\ \cdot\ ) \doteq \mathbb{P}\left(\ \cdot\ \mid y_j = +1, \{y_k\}_{k\neq j}\right), \tag{B.22}$$

$$\widetilde{\mathbb{P}}_{-}^{(j)}(\ \cdot\ ) \doteq \mathbb{P}\left(\ \cdot\ \mid y_j = -1, \{y_k\}_{k\neq j}\right), \tag{B.23}$$

$$\widetilde{\mathbb{E}}_{+}^{(j)}[\ \cdot\ ] \doteq \mathbb{E}\left[\ \cdot\ \mid y_j = +1, \{y_k\}_{k\neq j}\right], \tag{B.24}$$

$$\widetilde{\mathbb{E}}_{-}^{(j)}[\ \cdot\ ] \doteq \mathbb{E}\left[\ \cdot\ \mid y_j = -1, \{y_k\}_{k\neq j}\right], \tag{B.25}$$

where "$\cdot$" denotes any event with respect to the σ-algebra generated by Z and $\{y_j\}_{j=1}^N$. Note that in these conditional notations, all true labels of item k, $k \neq j$ remain unknown but are conditioned on, e.g., $\widetilde{\mathbb{E}}_{+}^{(j)}[y_k] = y_k$ for $k \neq j$.

Notice that

$$\begin{aligned}
\mathbb{P}\left(\hat{y}_j^{wmv} = y_j \mid y_1, \cdots, y_N\right) &= \mathrm{I}\left(y_j = +1\right) \cdot \mathbb{P}\left(\hat{y}_j^{wmv} = -1 \mid y_j = +1, \{y_k\}_{k\neq j}\right) \\
&\quad + \mathrm{I}\left(y_j = -1\right) \cdot \mathbb{P}\left(\hat{y}_j^{wmv} = +1 \mid y_j = -1, \{y_k\}_{k\neq j}\right) \\
&= \mathrm{I}\left(y_j = +1\right) \cdot \widetilde{\mathbb{P}}_{+}^{(j)}\left(\hat{y}_j^{wmv} = -1\right) \\
&\quad + \mathrm{I}\left(y_j = -1\right) \cdot \widetilde{\mathbb{P}}_{-}^{(j)}\left(\hat{y}_j^{wmv} = +1\right) \\
&= \mathrm{I}\left(y_j = +1\right) \cdot \widetilde{\mathbb{P}}_{+}^{(j)}\left(g_j(Z) < 0\right) \\
&\quad + \mathrm{I}\left(y_j = -1\right) \cdot \widetilde{\mathbb{P}}_{-}^{(j)}\left(g_j(Z) > 0\right),
\end{aligned}$$

where $g_j(Z) = \sum_{i=1}^M (2\hat{w}_i - 1) Z_{ij}$ and $\hat{w}_i$ is defined as (B.2).

We want to provide the upper bound on both $\widetilde{\mathbb{P}}_{+}^{(j)}\left(\hat{y}_j^{wmv} = -1\right) = \widetilde{\mathbb{P}}_{+}^{(j)}\left(g_j(Z) < 0\right)$ and $\widetilde{\mathbb{P}}_{-}^{(j)}\left(\hat{y}_j^{wmv} = +1\right) = \widetilde{\mathbb{P}}_{+}^{(j)}\left(g_j(Z) > 0\right)$.

We complete our proof in several steps.

Step 1. Providing an upper bound on $\widetilde{\mathbb{P}}_{+}^{(j)}\left(g_j(Z) < 0\right)$

Once we condition on $\{y_k\}_{k=1}^N$, all the entries in Z will be independent of each other, and then we can apply McDiarmid Inequality (McDiarmid, 1998) to the probability $\widetilde{\mathbb{P}}_{+}^{(j)}\left(g_j(Z) < 0\right)$.

From Lemma B.4, we get that if Z and Z' only differ on entry $(i^\star, j^\star)$, $Z'_{i^\star j^\star}$ is an independent copy of $Z_{i^\star j^\star}$, and so

$$\left|g_j(Z) - g_j(Z')\right| \leq d_{i^\star j^\star}^{(j)}.$$

Combining this with the results from Lemma B.4 we have

$$\begin{aligned}\sum_{i^\star=1}^{M}\sum_{j^\star=1}^{N}\left(d_{i^\star j^\star}^{(j)}\right)^2 &\leq M(N-1)\left(\frac{2(M-1)}{N}\right)^2 + M\left(2+\frac{2(M-1)}{N}\right)^2 \\ &\leq MN\left(\frac{2M}{N}\right)^2 + M\left(2+\frac{2M}{N}\right)^2 \\ &\leq \frac{4M}{N^2}\left[M^2N+(M+N)^2\right] \qquad \text{(B.26)}\end{aligned}$$

Applying the McDiamid Inequality, we get

$$\begin{aligned}\widetilde{\mathbb{P}}_+^{(j)}\left(g_j(Z)<0\right) &= \widetilde{\mathbb{P}}_+^{(j)}\left(g_j(Z)-\widetilde{\mathbb{E}}_+^{(j)}[g_j(Z)] < -\widetilde{\mathbb{E}}_+^{(j)}[g_j(Z)]\right) \\ &\leq \exp\left(-\frac{2\left(\widetilde{\mathbb{E}}_+^{(j)}[g_j(Z)]\right)^2}{\sum_{i^\star=1}^{M}\sum_{j^\star=1}^{N}\left(d_{i^\star j^\star}^{(j)}\right)^2}\right) \\ &\leq \exp\left(-\frac{N^2\left(\widetilde{\mathbb{E}}_+^{(j)}[g_j(Z)]\right)^2}{2M\left[M^2N+(M+N)^2\right]}\right), \qquad \text{(B.27)}\end{aligned}$$

provided $\widetilde{\mathbb{E}}_+^{(j)} g_j(Z) \leq 0$.

Now, if we can provide a lower bound of $\widetilde{\mathbb{E}}_+^{(j)}\left[g_j(Z)\right]$, then by replacing $\widetilde{\mathbb{E}}_+^{(j)}\left[g_j(Z)\right]$ with that lower bound in the last inequality, we can further bound $\widetilde{\mathbb{P}}_+^{(j)}\left(g_j(Z)<0\right)$ from above. Next, we aim at deriving a good lower bound of $\widetilde{\mathbb{E}}_+^{(j)}\left[g_j(Z)\right]$.

We can expand $g_j(Z)$ so that

$$\begin{aligned}\widetilde{\mathbb{E}}_+^{(j)}\left[g_j(Z)\right] = \widetilde{\mathbb{E}}_+^{(j)}\left[\sum_{i=1}^{M}(2\hat{w}_i-1)Z_{ij}\right] &= 2\sum_{i=1}^{M}\widetilde{\mathbb{E}}_+^{(j)}\left[\hat{w}_i Z_{ij}\right] - \sum_{i=1}^{M}\widetilde{\mathbb{E}}_+^{(j)} Z_{ij} \\ &= 2\sum_{i=1}^{M}\widetilde{\mathbb{E}}_+^{(j)}[\hat{w}_i Z_{ij}] - \sum_{i=1}^{M}(2w_i-1),\end{aligned}$$

since $\widetilde{\mathbb{E}}_+^{(j)} Z_{ij} = \mathbb{E}\left[Z_{ij} \mid y_j=+1, \{y_k\}_{k\neq j}\right] = \mathbb{E}[Z_{ij}|y_j=+1] = 2w_i - 1$.

Note that for any $i \in [M]$ and $j \in [N]$, given y_j, $\{Z_{ij}\}_{i=1}^M$ will be independent of $\{Z_{lk}\}_{k\neq j}$ and $\{y_k\}_{k\neq j}$. We will use this property for dropping all the irrelevant conditioned y_k's.

$$\begin{aligned}\widetilde{\mathbb{E}}_+^{(j)}[\hat{w}_i Z_{ij}] &= \widetilde{\mathbb{E}}_+^{(j)}\left[Z_{ij}\cdot\frac{1}{N}\sum_{k=1}^{L}\mathrm{I}\left(Z_{ik}=\hat{y}_k^{mv}\right)\right] = \frac{1}{N}\sum_{k=1}^{L}\widetilde{\mathbb{E}}_+^{(j)}\left[Z_{ij}\mathrm{I}\left(Z_{ik}=\hat{y}_k^{mv}\right)\right] \\ &= \frac{1}{N}\left[\sum_{k\neq j}\widetilde{\mathbb{E}}_+^{(j)}\left[Z_{ij}\mathrm{I}\left(Z_{ik}=\hat{y}_k^{mv}\right)\right] + \widetilde{\mathbb{E}}_+^{(j)}\left[Z_{ij}\mathrm{I}\left(Z_{ij}=\hat{y}_j^{mv}\right)\right]\right] \qquad \text{(B.28)}\end{aligned}$$

When $k \neq j$, Z_{ik} and $\hat{y}_k^{mv}$ are independent of Z_{ij} given y_j.

$$
\begin{aligned}
&\widetilde{\mathbb{E}}_+^{(j)}\left[Z_{ij}\mathrm{I}\left(Z_{ik}=\hat{y}_k^{mv}\right)\right] = \quad \widetilde{\mathbb{E}}_+^{(j)} Z_{ij}\cdot\widetilde{\mathbb{E}}_+^{(j)}\mathrm{I}\left(Z_{ik}=\hat{y}_k^{mv}\right)\\
=\;& (2w_i-1)\mathbb{P}(Z_{ik}=\hat{y}_k^{mv} \mid y_k)\\
\geq\;& \mathrm{I}\left(2w_i-1\geq 0\right)\cdot(2w_i-1)w_i\left[1-\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}+\frac{1-w_i}{M})^2}{M-1}\right)\right]\\
&+\mathrm{I}\left(2w_i-1<0\right)\cdot(2w_i-1)w_i\left[1+\frac{1-w_i}{w_i}\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}-\frac{w_i}{M})^2}{M-1}\right)\right] \qquad \text{(By Lemma B.3)}\\
\geq\;& (2w_i-1)w_i\left[\mathrm{I}\left(w_i\geq\frac{1}{2}\right)-\mathrm{I}\left(w_i\geq\frac{1}{2}\right)\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}+\frac{1-w_i}{M})^2}{M-1}\right)\right]\\
&+(2w_i-1)w_i\left[\mathrm{I}\left(w_i<\frac{1}{2}\right)+\mathrm{I}\left(w_i<\frac{1}{2}\right)\left(\frac{1-w_i}{w_i}\right)\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}-\frac{w_i}{M})^2}{M-1}\right)\right]\\
=\;& (2w_i-1)w_i\left[1+\frac{1-2w_i}{w_i}\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}-\frac{w_i}{M})^2}{M-1}\right)\right] \qquad \left(\text{Because } \bar{w}>\frac{1}{2}+\frac{1}{M}\right)\\
=\;& (2w_i-1)w_i\left[1+\frac{1-2w_i}{2w_i}\eta_i\right],
\end{aligned}
$$

where $\eta_i = 2\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}-\frac{w_i}{M})^2}{M-1}\right)$.

Furthermore,

$$
\begin{aligned}
&\widetilde{\mathbb{E}}_+^{(j)}\left[Z_{ij}\mathrm{I}\left(Z_{ij}=\hat{y}_j^{mv}\right)\right] = \quad \mathbb{E}_+^{(j)}\left[Z_{ij}\widetilde{\mathbb{E}}_+^{(j)}\left[\mathrm{I}\left(Z_{ij}=\hat{y}_j^{mv} \mid Z_{ij}\right)\right]\right]\\
=\;& w_i\mathbb{P}(\hat{y}_j^{mv}=+1 \mid y_j=+1, Z_{ij}=+1)-(1-w_i)\mathbb{P}(\hat{y}_j^{mv}=-1 \mid y_j=+1, Z_{ij}=-1)\\
=\;& w_i\left[1-\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}+\frac{1-w_i}{M})^2}{M-1}\right)\right]-(1-w_i)\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}-\frac{w_i}{M})^2}{M-1}\right) \quad \text{(By Lemma B.2)}\\
\geq\;& w_i\left[1-\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}-\frac{w_i}{M})^2}{M-1}\right)\right]-(1-w_i)\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}-\frac{w_i}{M})^2}{M-1}\right) \qquad \left(\text{As } \bar{w}>\frac{1}{2}+\frac{1}{M}\right)\\
=\;& w_i-\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}-\frac{w_i}{M})^2}{M-1}\right)\\
\geq\;& w_i-\eta_i/2
\end{aligned}
$$

Combining the two bounds above, we obtained

$$\begin{aligned}
\widetilde{\mathbb{E}}_+^{(j)}[\hat{w}_i Z_{ij}] &= \frac{1}{N}\left[\sum_{k\neq j}\widetilde{\mathbb{E}}_+^{(j)}\left[Z_{ij}\mathrm{I}\left(Z_{ik}=\hat{y}_k^{mv}\right)\right]+\widetilde{\mathbb{E}}_+^{(j)}\left[Z_{ij}\mathrm{I}\left(Z_{ij}=\hat{y}_j^{mv}\right)\right]\right] \\
&\geq \frac{1}{N}\left[(N-1)(2w_i-1)w_i(1+\frac{1-2w_i}{2w_i}\eta_i)+w_i-\frac{\eta_i}{2}\right] \\
&= \frac{1}{2N}\left[\left((N-1)(2w_i-1)^2+1\right)(1-\eta_i)+N(2w_i-1)\right] \\
&\geq \frac{1}{2N}\left[N(2w_i-1)^2(1-\eta_i)+N(2w_i-1)\right] \qquad \text{(Because } 1\geq(2w_i-1)^2\text{)} \\
&= \frac{1}{2}(2w_i-1)^2(1-\eta_i)+\frac{1}{2}(2w_i-1)
\end{aligned}$$

Let $\eta = 2\exp\left(-\frac{2M^2(\bar{w}-\frac{1}{2}-\frac{1}{M})^2}{M-1}\right)$, so $\eta \leq \eta_i \ \forall i \in [M]$

$$\begin{aligned}
\widetilde{\mathbb{E}}_+^{(j)} g_j(Z) &= 2\sum_{i=1}^{M}\widetilde{\mathbb{E}}_+^{(j)}\left[\hat{w}_i Z_{ij}\right]-\sum_{i=1}^{M}(2\hat{w}_i-1) \\
&\geq \sum_{i=1}^{M}(2w_i-1)^2(1-\eta_i)+\sum_{i=1}^{M}(2\hat{w}_i-1)-\sum_{i=1}^{M}(2\hat{w}_i-1) \\
&\geq (1-\eta)\sum_{i=1}^{M}(2w_i-1)^2 \\
&= 4M\tilde{\sigma}^2(1-\eta), \qquad \text{(B.29)}
\end{aligned}$$

where $\tilde{\sigma} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(2w_i-1)^2}$.

Since $\bar{w} \geq \frac{1}{2}+\frac{1}{M}+\sqrt{\frac{(M-1)\ln 2}{2M^2}}$, so $\eta \leq 1$, which implies $\widetilde{\mathbb{E}}_+^{(j)} g_j(Z) \geq 0$.

Then by (B.27) and (B.29) we have

$$\begin{aligned}
\widetilde{\mathbb{P}}_+^{(j)} g_j(Z) < 0 &\leq \exp\left(-\frac{N^2\left(\widetilde{\mathbb{E}}_+^{(j)}[g_j(Z)]\right)^2}{2M\left[M^2N+(M+N)^2\right]}\right) \\
&\leq \exp\left(-\frac{8MN^2\tilde{\sigma}^4(1-\eta)^2}{M^2N+(M+N)^2}\right). \qquad \text{(B.30)}
\end{aligned}$$

Step 2. With the same argument and following the same logic, we can obtain the same upper bound for $\widetilde{\mathbb{P}}_-^{(j)} g_j(Z) > 0$.

Step 3. Combining the results we obtained from Step 1 and Step 2.

Since

$$\mathbb{P}\left(\hat{y}_j^{wmv} = y_j \mid y_1, \cdots, y_N\right) = \mathrm{I}\left(y_j=+1\right)\cdot\widetilde{\mathbb{P}}_+^{(j)}\left(g_j(Z)<0\right)+\mathrm{I}\left(y_j=-1\right)\cdot\widetilde{\mathbb{P}}_-^{(j)}\left(g_j(Z)>0\right)$$

and both $\widetilde{\mathbb{P}}_{+}^{(j)}\left(g_j(Z)<0\right)$ and $\widetilde{\mathbb{P}}_{-}^{(j)}\left(g_j(Z)>0\right)$ have the same upper bound, we have that

$$\mathbb{P}\left(\hat{y}_j^{wmv}=y_j \mid y_1,\cdots,y_N\right)\leq \exp\left(-\frac{8MN^2\tilde{\sigma}^4(1-\eta)^2}{M^2N+(M+N)^2}\right).$$

The upper bound above does not depend on the value of $\{y_k\}_{k=1}^N$. By what we have discussed in the very beginning of the proof,

$$\mathbb{P}(\hat{y}_j^{mv}\neq y_j)\leq \exp\left(-\frac{8MN^2\tilde{\sigma}^4(1-\eta)^2}{M^2N+(M+N)^2}\right).$$

□

Now, we can directly prove Theorem 3.3 as follows:

Proof. **(Proof of Theorem 3.3)**
Since the upper bound of $\mathbb{P}(\hat{y}_j^{mv}\neq y_j)$ doesn't depend on j, it can directly imply that

$$\frac{1}{N}\sum_{j=1}^{N}\mathbb{P}(\hat{y}_j^{mv}\neq y_j)\leq \exp\left(-\frac{8MN^2\tilde{\sigma}^4(1-\eta)^2}{M^2N+(M+N)^2}\right),$$

which is the desired result in Theorem 3.3 .

□

Appendix C

Proofs for Chapter 4

C.1 Proof of Theorem 4.1

Proof. Without loss of generality, we denote by π the prevalence of true labels, i.e., $\mathbb{P}(y_j = k) = \pi_k, \forall j \in [N], k \in [L]$, where $\mathbb{P}$ denotes the probability measure. Note that even in the scenario that y_j is assumed as fixed instead of random, our analysis and results will still hold with $\pi_k = \mathrm{I}(y_j = k)$. Furthermore, we assume the group of workers are S with $|S| = M$.

For WMV-linear, the weights $\{\hat{\nu}_i\}_{i=1}^M$ are independent of the data matrix $\widetilde{Z}$. The associated weighted majority voting is

$$\hat{y}_j = \underset{k\in[L]}{\operatorname{argmax}} \sum_{i=1}^{M} \hat{\nu}_i \mathrm{I}(Z_{ij} = k),$$

where $\hat{\nu}_i = L\hat{w}_i - 1$ and $\mathbb{E}[\hat{w}_i] = w_i$. Thus, we have $\mathbb{E}\hat{\nu}_i = \nu_i = Lw_i - 1$ and $-1 \leq \hat{\nu}_i \leq L - 1$.

Let

$$s_k^{(j)} \doteq \sum_{i=1}^{M} \hat{\nu}_i \mathrm{I}(Z_{ij} = k), \quad \forall k \in [L], j \in [N] \tag{C.1}$$

be the aggregated score of jth item that on potential label class k. Thus the general aggregation rule can be written as $\hat{y}_j = \operatorname{argmax}_{k\in[L]} s_k^{(j)}$.

We will frequently discuss condition probability, expectation and variance conditioned on the event $\{y_j = k\}$. Without introducing ambiguity in the context, we define:

$$\mathbb{P}_k(\ \cdot\ ) \doteq \mathbb{P}(\ \cdot\ | y_j = k) \tag{C.2}$$

$$\mathbb{E}_k[\ \cdot\ ] \doteq \mathbb{E}[\ \cdot\ | y_j = k] \tag{C.3}$$

Note that

$$\mathbb{E}_k\left[s_l^{(j)}\right] = \sum_{i=1}^{M} \nu_i \left(w_i \mathrm{I}(l = k) + \left(\frac{1 - w_i}{L - 1}\right) \mathrm{I}(l \neq k) \right), \quad \forall l, k \in [L]. \tag{C.4}$$

First of all, we expand the error probability of labeling the j-th item wrong in terms of the conditional probabilities:

$$\mathbb{P}(\hat{y}_j \neq y_j) \;=\; \sum_{k\in[L]} \mathbb{P}(y_j = k)\mathbb{P}(\hat{y}_j \neq k | y_j = k) = \sum_{k\in[L]} \pi_k \mathbb{P}_k\left(\hat{y}_j \neq k\right). \tag{C.5}$$

Our major focus in this proof is to bound the term $\mathbb{P}_k\left(\hat{y}_j \neq k\right)$. Our approach will be based on the fact of the following events relations:

$$\bigcup_{l\in[L],l\neq k} \left\{ s_l^{(j)} > s_k^{(j)} \right\} \quad\subseteq\quad \{\hat{y}_j \neq k\} \quad\subseteq\quad \bigcup_{l\in[L],l\neq k} \left\{ s_l^{(j)} \geq s_k^{(j)} \right\}. \tag{C.6}$$

We want to provide an upper bound for $\mathbb{P}(\hat{y}_j \neq y_j)$. Note that

$$\mathbb{P}_k\left(\hat{y}_j \neq k\right) \;\leq\; \mathbb{P}_k\left(\bigcup_{l\in[L],l\neq k} \left\{ s_l^{(j)} \geq s_k^{(j)} \right\} \right) \leq \sum_{l\in[L],l\neq k} \mathbb{P}_k\left(s_l^{(j)} \geq s_k^{(j)} \right). \tag{C.7}$$

With $s_l^{(j)}$ defined as in (C.1), and when $l \neq k$, we define

$$\xi_{kl}^{(i)} \doteq s_l^{(j)} - s_k^{(j)} = \hat{\nu}_i \left(\mathrm{I}\left(Z_{ij} = l\right) - \mathrm{I}\left(Z_{ij} = k\right) \right), \tag{C.8}$$

$$\mathbb{E}_k\left[\xi_{kl}^{(i)}\right] = \mathbb{E}\hat{\nu}_i \cdot \left(\frac{1 - Lw_i}{L-1} \right) = -\frac{1}{L-1}(Lw_i - 1)^2, \tag{C.9}$$

$$\Lambda_{kl}^{(j)} \doteq \sum_{i=1}^{M} \mathbb{E}_k\left[s_k^{(j)} - s_l^{(j)} \right] = -\sum_{i=1}^{M} \mathbb{E}_k\left[\xi_{kl}^{(i)}\right] = \frac{1}{L-1}\sum_{i=1}^{M}(Lw_i - 1)^2. \tag{C.10}$$

We have

$$\begin{aligned}
\mathbb{P}_k\left(s_l^{(j)} \geq s_k^{(j)} \right) \;&=\; \mathbb{P}_k\left(\sum_{i=1}^{M} \hat{\nu}_i \left(\mathrm{I}\left(Z_{ij} = l\right) - \mathrm{I}\left(Z_{ij} = k\right) \right) \geq 0 \right) \\
&=\; \mathbb{P}_k\left(\sum_{i=1}^{M} \xi_{kl}^{(i)} - \sum_{i=1}^{M} \mathbb{E}_k\left[\xi_{kl}^{(i)}\right] \geq -\sum_{i=1}^{M} \mathbb{E}_k\left[\xi_{kl}^{(i)}\right] \right), \\
&=\; \mathbb{P}_k\left(\sum_{i=1}^{M} \xi_{kl}^{(i)} - \sum_{i=1}^{M} \mathbb{E}_k\left[\xi_{kl}^{(i)}\right] \geq \Lambda_{kl}^{(j)} \right)
\end{aligned} \tag{C.11}$$

Note that $\left\{\xi_{kl}^{(i)}\right\}_{i\in[M]}$ are conditional independent when given $\{y_j = k\}$, and they are bounded given the voting weight $\{\hat{\nu}_i\}_{i\in[M]}$ are bounded. Therefore, we could apply Hoeffding concentration inequality to further bound $\mathbb{P}_k\left(s_l^{(j)} \geq s_k^{(j)} \right)$.

Apparently, $-1 \leq \xi_{kl}^{(i)} \leq (L-1)$. Note $\Lambda_{kl}^{(j)} \geq 0$, by appling Hoeffding inequality to (C.11),

$$\begin{aligned}
\mathbb{P}_k\left(s_l^{(j)} \geq s_k^{(j)}\right) &\leq \mathbb{P}_k\left(\sum_{i=1}^{M} \xi_{kl}^{(i)} - \sum_{i=1}^{M} \mathbb{E}_k\left[\xi_{kl}^{(i)}\right] \geq \Lambda_{kl}^{(j)}\right) \\
&\leq \exp\left(-\frac{2{\Lambda_{kl}^{(j)}}^2}{\sum_{i=1}^{M}\left[(L-1)-(-1)\right]^2}\right) \\
&\leq \exp\left(-\frac{2{\Lambda_{kl}^{(j)}}^2}{\left(L\sqrt{M}\right)^2}\right) \\
&\leq e^{-2t^2},
\end{aligned}$$

where $t = \frac{1}{L(L-1)\sqrt{M}} \sum_{i=1}^{M} (Lw_i - 1)^2$.

The right hand side of last ineiquality does not depend on k, l or i, straightforwardly,

$$\mathbb{P}_k\left(\hat{y}_j \neq k\right) \leq \sum_{l \in [L], l \neq k} \mathbb{P}_k\left(s_l^{(j)} \geq s_k^{(j)}\right) \leq (L-1)e^{-2t^2}. \tag{C.12}$$

Furthermore, we have

$$\begin{aligned}
\mathbb{P}(\hat{y}_j \neq y_j) &= \sum_{k \in [L]} \pi_k \mathbb{P}_k\left(\hat{y}_j \neq k\right) \\
&\leq (L-1)e^{-2t^2}\left(\sum_{k \in [L]} \pi_k\right) \\
&= e^{-2t^2 + \ln(L-1)}
\end{aligned} \tag{C.13}$$

The bound $e^{-2t^2+\ln(L-1)}$ does not depend on j, thus it is also a valid bound for the mean error rate. That is to say

$$\frac{1}{N}\sum_{j=1}^{N} \mathbb{P}\left(\hat{y}_j \neq y_j\right) \leq \frac{1}{N}\sum_{j=1}^{N} e^{-2t^2+\ln(L-1)} = e^{-2t^2+\ln(L-1)}.$$

Note that $t = \frac{F(S)}{L(L-1)}$, thus we have proved the desired result.

□

C.2 Proof of Lemma 4.2

Proof. Assume $|S| = k$. Let $\hat{F}_{\text{plug}}(S)$ be the pluggin estimator for $F(S)$, i.e.,

$$\hat{F}_{\text{plug}}(S) = \frac{1}{\sqrt{k}} \sum_{i \in S} (L\hat{w}_i - 1)^2.$$

First we show that $\hat{F}_{\text{plug}}(S)$ is a biased estimate of $F(S)$.

$$\begin{aligned}
\mathbb{E}[\hat{F}_{\text{plug}}(S)] &= \frac{1}{\sqrt{k}} \sum_{i \in S} \left(L^2 \mathbb{E}[\hat{w}_i^2] - 2L\mathbb{E}[\hat{w}_i] + 1 \right) \\
&= \frac{1}{\sqrt{k}} \sum_{i \in S} \left(L^2 \left(\text{var}(\hat{w}_i) + w_i^2 \right) - 2Lw_i + 1 \right) \\
&= \frac{1}{\sqrt{k}} \sum_{i \in S} \left((Lw_i - 1)^2 + L^2 \text{var}(\hat{w}_i) \right) \\
&= F(S) + \frac{1}{\sqrt{k}} \sum_{i \in S} L^2 \text{var}(\hat{w}_i).
\end{aligned}$$

Note that $\mathbb{E}\hat{w}_i = w_i$ and $\mathbb{E}[\hat{\text{var}}(\hat{w}_i)] = \text{var}(\hat{w}_i)$, thus we can move terms around to construct an unbaised estimate of $F(S)$ based on $\hat{F}_{\text{plug}}(S)$:

$$F(S) = \mathbb{E}\left[\hat{F}_{\text{plug}}(S) - \frac{1}{\sqrt{k}} \sum_{i \in S} L^2 \hat{\text{var}}(\hat{w}_i) \right],$$

which leads to a unbaised estimate of $F(S)$ as follows.

$$\begin{aligned}
\hat{F}(S) &= \hat{F}_{\text{plug}}(S) - \frac{1}{\sqrt{k}} \sum_{i \in S} L^2 \hat{\text{var}}(\hat{w}_i) \\
&= \frac{1}{\sqrt{k}} \sum_{i \in S} \left((L\hat{w}_i - 1)^2 - L^2 \hat{\text{var}}(\hat{w}_i) \right),
\end{aligned}$$

which is the same form as (4.7) and $\mathbb{E}[\hat{F}(S)] = F(S)$.

□

C.3 Proof of Theorem 4.3

Proof. Similar to the proof of Lemma 4.2, we assume $|S| = k$. With $\hat{w}_i$ and $\hat{\text{var}}(\hat{w}_i)$ defined as in (4.10), it is straightforwardly to show that $\mathbb{E}\hat{w}_i = w_i$ and $\mathbb{E}[\hat{\text{var}}(\hat{w}_i)] = \frac{w_i(1-w_i)}{n} = \text{var}(\hat{w}_i)$, then by Lemma 4.2 the corresponding unbaised estimator of $F(S)$ is

$$\begin{aligned}
\hat{F}(S) &= \frac{1}{\sqrt{k}} \sum_{i \in S} \left[(L\hat{w}_i - 1)^2 - \frac{L^2 \hat{w}_i (1 - \hat{w}_i)}{n - 1} \right] \\
&= \frac{n}{(n-1)\sqrt{k}} \sum_{i \in S} \left[\left(L\hat{w}_i - 1 - \frac{L-2}{2n} \right)^2 - \left(\frac{(L-2)^2}{4n^2} + \frac{L-1}{n} \right) \right],
\end{aligned}$$

therefore G has two equivalent forms:

$$G = (L\hat{w}_i - 1)^2 - \frac{L^2\hat{w}_i(1-\hat{w}_i)}{n-1} \tag{C.14}$$

and

$$G = \frac{n}{n-1}\left[\left(L\hat{w}_i - 1 - \frac{L-2}{2n}\right)^2 - \left(\frac{(L-2)^2}{4n^2} + \frac{L-1}{n}\right)\right] \tag{C.15}$$

Next, we prove the confidence interval of $F(S)$ based on the form (C.15) of G. We define random variables $\{X_i\}_{i\in S}$ as

$$X_i \doteq \left(L\hat{w}_i - 1 - \frac{L-2}{2n}\right)^2 - \lambda,$$

where $\lambda = \left(\frac{(L-2)^2}{4n^2} + \frac{L-1}{n}\right)$. Then

$$\hat{F}(S) = \frac{n}{(n-1)\sqrt{k}}\sum_{i\in S} X_i.$$

Note that $\{X_i\}_{i\in S}$ are a collection of independent random variables, $-\lambda \le X_i \le \left(L-1-\frac{L-2}{2n}\right)^2 - \lambda$, and $\mathbb{E}\hat{F}(S) = F(S)$. We can apply Hoeffding Inequality to bound the following probability,

$$\begin{aligned}
&\mathbb{P}\left(|\hat{F}(S) - F(S)| \le \frac{n}{(n-1)\sqrt{k}}\cdot\beta\right)\\
=\ &\mathbb{P}\left(\frac{n}{(n-1)\sqrt{k}}|\sum_{i\in S} X_i - \mathbb{E}\left[\sum_{i\in S} X_i\right]| \le \frac{n}{(n-1)\sqrt{k}}\cdot\beta\right)\\
=\ &\mathbb{P}\left(|\sum_{i\in S}(X_i - \mathbb{E}X_i)| \le \beta\right)\\
\ge\ &1 - 2\exp\left(-\frac{2\beta^2}{k(L-1-\frac{L-2}{2n})^4}\right)\\
=\ &1 - 2e^{-2\alpha^2},
\end{aligned} \tag{C.16}$$

where $\alpha = \frac{\beta}{(L-1-\frac{L-2}{2n})^2\sqrt{k}}$ and the inequality is due to Hoeffding bound. Meanwhile,

$$\begin{aligned}
&\mathbb{P}\left(|\hat{F}(S) - F(S)| \le \frac{n}{(n-1)\sqrt{k}}\cdot\beta\right)\\
=\ &\mathbb{P}\left(|\hat{F}(S) - F(S)| \le \frac{n\left(L-1-\frac{L-2}{2n}\right)^2}{(n-1)}\alpha\right)\\
\le\ &\mathbb{P}\left(|\hat{F}(S) - F(S)| \le \frac{n(L-1)^2}{n-1}\alpha\right),
\end{aligned} \tag{C.17}$$

which implies that $[\hat{F}(S) - \frac{n(L-1)^2}{n-1}\alpha, \hat{F}(S) + \frac{n(L-1)^2}{n-1}\alpha]$ covers $F(S)$ with probability at least $1 - 2e^{-2\alpha^2}$.

□

C.4 Proof of Theorem 4.4

Proof. Let $x_i = (L\hat{w}_i - 1)^2 - L^2\hat{\text{var}}(\hat{w}_i)$, then the optimization problem (4.11) can be written as

$$\underset{S\subseteq\Omega}{\text{argmax}}\, \hat{F}(S) \qquad s.t. \quad |S| \leq K \tag{C.18}$$

where

$$\hat{F}(S) = \frac{1}{\sqrt{|S|}} \sum_{i\in S} x_i.$$

Note that $\{\hat{w}_i\}_{i\in\Omega}$ are given in these optimization problems, thus we do not treat x_i as random. The problems (4.11) (i.e., (C.18)) are deterministic combinatorial problems. In this proof, we show that the output from Algorithm 3 achieves the global maximum of problem (C.18).

The worker selection problem is to select a worker set denote by $S^\star$ such that $\hat{F}(S^\star) \geq \hat{F}(S)$ for any set of workers $S \subseteq \Omega$. Let σ be a permutation of $\Omega = \{1, 2, \cdots, M\}$ such that $x_{\sigma(1)} \geq x_{\sigma(2)} \geq \ldots \geq x_{\sigma(M)}$.

We want to show that given any globally optimal solution of problem (C.18) $S^\star$, which has cardinality $|S^\star| = k^\star$, we have $\hat{F}(S^\star) = \hat{F}(\{\sigma(1), \sigma(2), \cdots, \sigma(k^\star)\})$.

To see this, let $S' = \{\sigma(1), \sigma(2), \cdots, \sigma(k^\star)\}$, and we assume $\hat{F}(S^\star) > \hat{F}(S')$. Since the value of function $\hat{F}(S)$ only depdends on cardinality of S and $\{x_i\}_{i\in S}$, the configuration [1] of values $\{x_i\}_{i\in S^\star}$ is not equal to $\{x_i\}_{i\in S'}$. This further implies that there exist $i \in S^\star \backslash S'$ and $j \in S' \backslash S^\star$ such that $x_i \neq x_j$. Since $i \notin S'$ and S' is the top $k^\star$ x-values, then $x_i < x_j$. Therefore, if we replace i in $S^\star$ with j will increase the value of $\hat{F}$, i.e., $\hat{F}((S^\star \backslash \{i\}) \cup \{j\}) > \hat{F}(S^\star)$. This contradicts with the fact that $\hat{F}(S^\star)$ is global optimum. Thus we conclude that $\hat{F}(S^\star) = \hat{F}(\{\sigma(1), \sigma(2), \cdots, \sigma(k^\star)\})$.

The analysis above implies that if we know the cardinality of the global optimal solution $k^\star$, then the top $k^\star$ workers in terms of x-values will be the global optimum in problem (C.18), although it might not be the unique one. Based on the fact that the cardinality of $S^\star$ has to be one of the values in $\{1, 2, \cdots, \min(K, M)\}$, we can compute the value of $\hat{F}(\{\sigma(1), \sigma(2), \cdots, \sigma(k)\})$ with k from 1 to $\min(K, M)$. Then the maximum of the yielded $\hat{F}$ function values has to be a global optimum of problem (C.18), and thus the corresponding worker set is global optimum of problem (4.11). Algorithm 3 follows exactly the procedure described above, therefore it output a globally optimal worker set.

□

[1] Here, we use *configuration* to denote sets that allow duplicates of values such as $\{1, 1, 1, 2, 3, 3\}$.

As mentioned in the remark of Theorem 4.4, we can show that S^* also solves the following multi-objective optimization problem that simultaneously maximizes the score $\hat{F}(S)$ and minimizes the number $|S|$ of workers actually deployed.

Theorem C.1. *Consider a multiple-objective optimization problem,*

$$\underset{S\subseteq\Omega}{\operatorname{argmax}}(\hat{F}(S), \ -|S|), \qquad s.t. \quad |S| \leq K,$$

then $S^\star$ is its Pareto optimal solution.

Proof. By Theorem 4.4, suppose $S^\star$ is the global optimum of problem (4.11) and $|S^\star| \leq K$, then there is no other set S such that $S \leq K$ and $\hat{F}(S) > \hat{F}(S^\star)$. This implies that within the sets with cardinality no more than K, there is no other set could improve $\hat{F}(S)$. Thus $S^\star$ is *Pareto optimal*[2] according to its definition in the context of multiple objective optimization. □

[2] http://en.wikipedia.org/wiki/Multi-objective_optimization