Learning-based mechanism design for microtask crowdsourcing

Hu, Zehong

2018

Nanyang Technological University

Learning-Based Mechanism Design for Microtask Crowdsourcing

Ph.D Thesis

By

Zehong Hu

Supervisor: Prof. Jie Zhang

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University in partial fulfillment of the requirement for the degree of Doctor of Philosophy

2018

Abstract

Microtask crowdsourcing, as an efficient and economical method for a requester to outsource tasks to online workers, is becoming increasingly popular in many domains, especially collecting labels for large-scale datasets. In microtask crowdsourcing, a requester usually needs to accomplish three steps: firstly, recruit as many as possible workers from the market; then, assign tasks to the workers based on their performance; lastly, reward good workers and meanwhile punish bad workers. For these three steps, various mechanisms have been proposed. Under certain assumptions about workers' responses to the rewards, these mechanisms can theoretically ensure workers to follow the strategies desired by the requester and thus maximize the revenue of the requester. However, these assumptions may be violated in practice, which causes the failure of these theoretically elegant mechanisms. Thereby, recent studies move their focus to the learning-based mechanisms which learn workers' models in an online fashion rather than simply assuming one. In this thesis, we propose three novel learning-based mechanisms, each for one step, to push forward the studies in this direction.

More specifically, when recruiting workers from the microtask crowdsourcing market, an important factor that can be controlled by the requester is the base reward guaranteed for each task, termed the price of tasks. Deciding a proper price is very challenging because overpricing causes inefficient use of the budget, whereas underpricing may lead to an insufficient number of participating workers. To solve this problem, researchers propose posted-price mechanisms which use the classic multi-armed bandit algorithms to learn the worker model online and accordingly adjust the price to be optimal. In this thesis, we propose a novel posted-price mechanism which not only outperforms existing mechanisms on improving the utilities of the requester but also avoids their need for a finite price range as the prior knowledge. The advantages are achieved by designing an optimal multi-armed bandit algorithm to exploit the unique features of microtask crowdsourcing. We theoretically show the optimality of our algorithm and prove that the performance upper bound can be achieved without the need for a prior price range. We also conduct extensive experiments using real price data to verify the advantages and practicability of our mechanism.

Improper task assignment in microtask crowdsourcing may significantly increase the number of needed labels. This is because crowdsourcing workers are often non-experts and redundant labels are collected for each task. Hence, researchers propose active task assignment mechanisms which use variational inference and active learning strategies to learn worker models and decide the optimal task assignment, respectively. However, active learning assumes labels are fully reliable while microtask crowdsourcing may generate many wrong labels. These active learning strategies, which evaluate each possible assignment at first and then greedily select the optimal one, thus may require prohibitive computation time but still leave some room for improvement. In this thesis, we develop a novel active task assignment mechanism by firstly deriving an efficient algorithm for assignment evaluation. Then, to overcome the uncertainty of labels, we modulate the scope of the greedy task assignment with the label uncertainty and keep the evaluation being optimistic. The experiments on four popular worker models and four MTurk datasets show that our mechanism needs the least workers and has the highest computation efficiency.

The rewards for workers consist of the rewards guaranteed when recruiting them and the bonus. Through adjusting the bounds according to workers' contributions, reward mechanisms aim to induce high-quality labels. To this end, the existing reward mechanisms assume workers are fully rational and evaluate their contributions by comparing the labels of two workers. In this thesis, we build a novel sequential reward mechanism by firstly developing a Bayesian inference algorithm which uses all the collected labels to evaluate workers' contributions. Then, we propose a reinforcement learning algorithm, relying on the above evaluation, to uncover how workers respond to different levels of rewards. Our mechanism determines the rewards for workers based on workers' current contributions and their patterns of responding to the rewards. We theoretically prove that our mechanism is able to incentivize workers to provide high-quality labels. We also empirically show that, compared with the simple comparison in the existing reward mechanisms, our Bayesian inference algorithm can improve the robustness and lower the variance of rewards. Besides, our reinforcement learning algorithm can adapt our mechanism to different kinds of worker models, which avoids the impractical assumption for rational workers.

To summarize, we propose three novel learning-based mechanisms for the three core steps of microtask crowdsourcing. Putting these mechanisms together, we actually expect to construct a novel microtask crowdsourcing system which is more intelligent and more practical in managing a large crowd of workers.

Acknowledgments

First and foremost, I would like to take this chance to express my sincere gratitude to my supervisor, **Dr. Jie Zhang**. He always encourages me by his passion and enthusiasm on research. He also inspire me by his incredibly sharp perceptions on various research problems. Without his support and insightful suggestions, it would be impossible for me to come to the final stage. In addition to research, his way of treating work, friends and students will always be a good guidance for me.

I would also like to thank Rolls-Royce and National Research Funding for providing me financial support. Special thanks to Dr. Hui Xi, the manager of our research project, for her kind help throughout the whole four years.

Many thanks are given to all my coauthors: Dr. Meng Sha, Dr. Moath Jarrah, Dr. Yang Liu, Yitao Liang and Yukun Ma. I also have greatly benefited from the friendship with all the staffs and students in Rolls-Royce@NTU Cooperate Lab, especially Wen Song, Sa Gao, and Zhenchao Bing.

Finally, but most importantly, I would like to dedicate this thesis to my parents and my fiancée, Zhihui Song, for their persistent love and endless support. I do feel sorry for the lack of time spending with my families.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Crowdsourcing is an economical and efficient method to outsource thousands of tasks to a crowd of individuals on the Web via an open call for contributions [31, 80, 14]. It has been used to solve problems in various domains, from design innovation to software development [89]. Early applications of crowdsourcing usually incentivize the worker crowd by the fun or community belonging incentives. A famous example of gamification-based crowdsourcing systems is the ESP game [87]. In this game, all players cannot interact with each other. Two players will win the game only when they generate the same tag for one picture. Later on, researchers propose an extension of the ESP game, Peekaboom, where the players need to annotate specific objects in the required image [88]. Another famous crowdsourcing project is Wikipedia, the largest and most popular general reference project on the Internet[1]. In spite of these successful applications, the use of monetary rewards become more and more popular in recent years because they can attract workers to make persistent and stable contributions [17]. Thus, in this thesis, we focus on the cutting-edge studies on crowdsourcing systems which use monetary rewards to induce the active participation and high efforts from a large crowd of self-interested online workers.

Depending on the types of tasks, crowdsourcing takes different forms. In this thesis, we focus on microtask crowdsourcing, one of the most widely-adopted crowdsourcing forms. In microtask crowdsourcing, one group of people, who are called the requesters, post tasks on a microtask crowdsourcing platform. Another group of people (so called workers) contribute their efforts to solve these tasks and report their

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

solutions to a requester. For each report provided by the workers, the requester will pay a small monetary reward in return. The tasks in microtask crowdsourcing are usually repetitive and easy for an individual to perform but very tedious. Popular examples of such tasks include: image tagging, audio transcription, survey filling, rewriting product reviews, and transcription of scanned shopping receipts [22].

The most common and successful applications of microtask crowdsourcing is to provide data for machine learning algorithms [17, 77, 91]. For example, in the field of computer vision, the famous ImageNet dataset contains $14,197,122$ images that are annotated with the help of the crowdsourcing workforce [42]. In the field of natural language processing, the well-known RTE dataset employs crowdsourcing workers to check whether a hypothesis sentence can be inferred from the provided sentence [81]. Furthermore, some researchers even propose hybrid human-machine systems which deeply integrate microtask crowdsourcing and machine-running algorithms [87]. This hybrid system can combine both the scalability of machine-running algorithms over large amounts of data as well as the ability of human workers in analyzing and understanding data. For example, crowd-powered databases leverage crowdsourcing to match, rank, and aggregate data based on fuzzy criteria [20]. Due to the strong connection between crowdsourcing and data collection, in this thesis, we call workers' reports about tasks as ***labels*** without considering which kind of tasks they are.

According to the report of World Bank in 2015 [43], the past several years have seen a fast growth of the microtask crowdsourcing markets. To support this rapidly growing market, various platforms have been developed. Prominent microtask crowdsourcing platforms include Amazon Mechanical Turk (AMTurk), Microworkers, and CrowdFlower. Now, there are thousands of active requesters and workers all around the world on these platforms every day. This makes crowdsourcing a promising alternative to the traditional employment which requires workers to stay in the office. However, in the way of managing this large number of workers, these platforms still heavily rely on the requesters to decide everything. In this thesis, we wish to develop an intelligent microtask crowdsourcing system to remove requesters' hands from the wheel.

Generally speaking, to leverage the power of crowd workers on these microtask crowdsourcing platforms, a requester needs to accomplish the following three steps:

1) **Worker Recruitment:** After the tasks are posted, all workers are able to see the tasks. To finish the tasks, the requester has to recruit a large enough number of workers. There are usually hundreds of data requesters on a platform. Workers are free to work on tasks that they think are the most interesting, those they like to complete or the best paid ones. Thus, in worker recruitment, a requester actually needs to compete with other requesters on the platform.

2) **Task Assignment:** After recruiting a certain number of workers from the microtask crowdsourcing market, the requester needs to decide which task to be assigned to which worker. In microtask crowdsourcing, a requester usually needs to handle hundreds of tasks and workers with varying difficulty and quality levels. Improper task assignment may signifcantly increase the number of labels needed for high accuracy, and the requester needs to pay for each collected label. Thus, finding a proper method to assign tasks is very important.

3) **Reward Payment:** After collecting labels from the workers, the requester needs to pay monetary rewards to the workers. It is very natural to expect that those workers who provide high-quality labels should get high rewards as the incentives, while those who are not so responsible should be given low rewards as the punishment. By doing so, the workers can be incentivized to be more responsible for the assigned tasks, and thus the label accuracy can be improved.

To help the requester to gain the maximal revenue in the above three steps, various mechanisms have been developed. These mechanisms try to affect workers' strategies by adjusting the monetary rewards paid to workers. However, they usually need to make some impractical assumptions about workers. For example, workers are fully rational—i.e. workers have full knowledge about the mechanisms and only follow the utility-maximizing strategy. Recently, to avoid these assumptions, researchers proposed learning-based mechanisms which learn workers' models in an online fashion rather simply assuming one. Nevertheless, the existing learning-based mechanisms often simply grab some classic machine learning techniques and plug them into the settings of microtask crowdsourcing. This way of design overlooks the unique features or problems of microtask crowdsourcing, which certainly cannot improve the revenue of the requester to the utmost. Thereby, in this thesis, we firstly aim to improve the existing learning-based mechanisms by developing machine learning algorithms specifically for the worker recruitment and task assignment steps of microtask crowdsourcing. Then, for the step of reward payment, since no learning-based mechanism has been developed, we design a completely new one based on Bayesian inference and reinforcement learning.

1.1 Problems and Challenges

1.1.1 Worker Recruitment

In microtask crowdsourcing markets, workers are free to work on tasks that they like. For a requester, to attract more workers, one commonly-adopted way is to promise more rewards for each task. Different from the final rewards paid in the third step (reward payment), the promised rewards here mean the base rewards which are only a part of the final payments. For the sake of distinction, we call this base reward as the price of a task. Since the budget owned by the requester is always limited and the number of tasks is very large, determining the price of tasks properly is always a key challenge of recruiting workers in microtask crowdsourcing. Overpricing causes inefficient use of the budget, whereas underpricing may lead to an insufficient number of participating workers [2]. To solve this problem, researchers propose an incentive mechanism which requires all workers to report the prices they are willing to accept at first [78]. It then decides the optimal price accordingly. The idea behind this mechanism is that it can theoretically ensure that workers can only get the maximal expected rewards by truthfully reporting their willing-to-accept prices. However, in practice, workers may not be able to figure out an explicit willing price or always follow the reward-maximizing strategy. We say that workers are bounded rational in this case. Due to the existence of bounded rationality, researcher propose another very promising alternative mechanism, the posted-price mechanism [79]. This mechanism only requires workers to make reject-or-accept decisions for the price offered to them. With these binary decisions, the mechanism can learn the worker model online and accordingly adjusts the price to be optimal. This way of design not only avoids the aforementioned strong assumption about workers but also lowers the communication burden between workers and the mechanism, which greatly facilitates the practical usage.

Nevertheless, existing posted-price mechanisms are still inadequate in two aspects. First, existing posted-price mechanisms directly apply the classic bandit algorithms which need to blindly try all the possible prices in the beginning [4]. They overlook the unique features of microtask crowdsourcing: the number of workers willing to accept the task is unknown and increases monotonically with the increasing price, whereas the number of workers allowed by the limited budget is accurately known and decreases monotonically. These features can be employed to guide the price adjustment of posted-price mechanisms. Second, the performance of existing posted-price mechanisms significantly degrades if the number of possible prices is very large. This is because all the possible prices need to be tried in their mechanisms. Hence, the requester is required to input a proper range of prices in advance, which causes much inconvenience. Thus, in this thesis, we focus on overcoming the above two drawbacks by exploiting the unique features of the worker recruitment process in microtask crowdsourcing. We develop a novel posted-price mechanism which not only reaches the theoretical optimum but also avoids setting a prior price range.

1.1.2 Task Assignment

Since tasks in microtask crowdsourcing are usually tedious and workers are often non-experts, the resulting labels can be very noisy [37, 12]. As a remedy, many microtask crowdsourcing markets follow the round-robin repeated labeling strategy which randomly assigns multiple workers with a same task and all tasks are assigned with a same number of workers [71]. In principle, as long as a sufficient number of labels is collected, this repeated labeling strategy can generate reports with high label accuracy. However, workers should be paid for each label they provide. The cost of having a great amount of redundant labels is non-trivial. In addition, the round-robin strategy actually overlooks the difference between different tasks and workers, and thus will waste lots of budget on easy tasks and low-quality workers. Thereby, it is very crucial to look into a better way of allocating budget among tasks and workers so that the accuracy growth rate can be maximized. By doing so, we can use few reports to generate high-accuracy labels.

To achieve this objective, Welinder and Perona [92] propose an active task assignment mechanism which applies the classic active learning strategy, uncertainty sampling [45]. They choose a high quality worker to handle the most uncertain task at each step. The uncertainties of tasks and the quality of workers are evaluated on-the-fly via running the variational inference algorithm online. Later on, Simpson and Roberts [76] further improve the label accuracy by introducing another more advanced active learning strategy, expected error reduction [70]. However, this strategy requires to predict the expected benefit of each possible task assignment through running the relatively slow variational inference algorithm for every possible future report, which is computationally very expensive in microtask crowdsourcing. Besides, all the data in active learning are assumed to be fully reliable, while a large number of labels in microtask crowdsourcing may be wrong. This difference causes these classic active learning strategies to be unable to handle the inference uncertainties in microtask crowdsouring, which inevitably lowers the label accuracy. Furthermore, their variational inference algorithms are developed for a specific worker model. We need to reinvent the wheel from scratch if changing to another worker model. Thus, in this thesis, we solve these unique problems by combining active learning and variational inference in microtask crowdsourcing. We formulate a novel active task assignment mechanism which not only improves the label accuracy but also boosts the computation efficiency. Besides, our mechanism can flexibly incorporate different worker models.

1.1.3 Reward Payment

As discussed in the last section, notwithstanding the distinctive advantages of microtask crowdsourcing, one salient concern about crowdsourcing is the quality of the collected labels, as workers are often non-experts and tasks are very tedious. How to improve the quality of the collected labels is always an important research topic of microtask crowdsourcing. In addition to assigning more tasks to more responsible workers, another popular way of improving the label quality is to adjust the rewards for workers according to their contributions. Our objective is to reward the workers who provide high-quality labels and punish those who only report low-quality labels. However, in microtask crowdsourcing, it is very difficult to verify workers' contributions because the ground-truth labels are either unavailable or too costly to obtain. Thus, researchers call this problem information elicitation without verification [90]. To solve this problem, a class of incentive mechanisms, collectively called peer prediction, has been developed [57, 39, 95, 94, 64, 97, 49]. The core idea of peer prediction is quite simple and elegant – the mechanism designer financially incentivizes workers according to the score of their contributions in comparison with their peers'. The payment rules are designed so that each worker reporting truthfully or exerting high-efforts to generate high-quality labels is a strict Bayesian Nash Equilibrium for all workers.

Nonetheless, we note two undesirable properties of existing peer prediction mechanisms. Firstly, from the machine learning studies on the collected labels[2] [99], we know that the collected labels contain a wealth of information about the true labels and workers' labeling accuracy. However, existing peer prediction mechanisms often rely on comparing the labels of the targeted worker with those of a small set of reference workers, which only exploits a limited share of the overall collected information. This way of design will inevitably lower the robustness, and meanwhile increase the

[2] In microtask crowdsourcing, there is another branch of research which develops machine learning algorithms to infer true labels from the collected noisy labels [66, 48, 101, 99]. These studies often work as one-shot, post-processing procedures facing a static set of workers, whose labeling accuracy is fixed and informative. They completely ignore the effects of rewards on workers' strategies.

variance of rewards, which is unfavorable in practice. Secondly, existing peer prediction mechanisms simplify workers' responses to the incentive mechanism by assuming that workers are all fully rational. In other words, they assume that workers have full knowledge about the incentive mechanism and will only follow the utility-maximizing strategy we design for them. However, evidence reveals that human agents may follow bounded rational models, or may improve their responding strategies gradually in practice [75, 11, 21, 52, 53, 54]. To solve the above two drawbacks, we develop a novel inference aided reinforcement reward mechanism by replacing the simple label comparison with the combination of Bayesian inference and reinforcement learning algorithms. Those machine learning algorithms can not only fully exploit the information contained in the collected labels but also analyze workers' responses to the rewards and adjust the rewards accordingly. By doing so, our mechanism can incentivize workers in a better way and avoid the impractical rationality assumption.

1.2 The Proposed Approaches

Focusing on the aforementioned deficiencies of the existing incentive mechanisms, we propose three novel mechanisms which are summarized as follows:

- **The optimal posted-price mechanism for microtask crowdsourcing** [35, 34]: For worker recruitment, we propose a novel posted-price mechanism to fully exploit the unique features of microtask crowdsourcing. More specifically, we first convert the pricing problem into an equivalent multi-armed bandit (MAB) problem. Then, we develop an algorithm that offers each coming worker the minimum price at which the anticipated number of workers willing to accept the task approximately equals the number of workers allowed by the budget. In microtask crowdsourcing, the number of workers willing to accept the task is unknown and increases monotonically with the increasing price, whereas the number of workers allowed by the limited budget is accurately known and decreases monotonically. Due to the monotonicity features, our algorithm is proven to be optimal. In addition, these features ensure that our algorithm will never explore overly high prices and thus does not need to set a price range in advance. To empirically validate the advantages of our mechanism over existing ones, we conduct extensive experiments using three popular worker models as well as the real-world price data collected from MTurk, a widely-adopted microtask crowdsourcing platform. Experimental results confirm that our mechanism achieves almost the same performance as the idealized case where the accurate worker model is known in advance (i.e., the optimal price is used from the very beginning). We also carry out robustness tests to ensure the practicability of our mechanism.

- **An active task assignment mechanism for microtask crowdsourcing** [36]: To improve the task assignment of microtask crowdsourcing, we propose a novel active task assignment mechanism. It utilizes the variational inference algorithm to learn the estimates of true labels and worker models in an online fashion. Different from the existing studies that assume one specific worker model, we provide a unified formulation of different worker models so that our inference algorithm can flexibly work with any of the worker models. In our mechanism, we also develop a novel prediction-based task assignment strategy. To improve computation efficiency, we derive the first-order approximation equations of our variational inference algorithm that can be solved by Newton's method efficiently. To suppress the mutual reinforcement between the uncertainties of true labels and worker models, we develop a heuristic rule to modulate the scope of task assignment based on the uncertainty measurement of the current inference results. Besides, we replace all the expected values with the upper confidence bounds to achieve an optimistic prediction. We conduct extensive experiments based on four popular worker models and four MTurk datasets. The empirical results show that our mechanism not only requires the least number of labels for high label accuracy but also achieves the highest computation efficiency among all existing prediction-based task assignment approaches.

- **An inference aided reinforcement learning mechanism for microtask crowdsourcing** [33, 32]: To overcome the drawbacks of peer prediction mechanisms which are used to decide the rewards for workers in microtask crowdsourcing, we combine traditional mechanism design with Bayesian inference and reinforcement learning techniques. The high level idea is as follows: we divide the large to-be-collected dataset into relatively small task packages. At each step, we employ workers to handle one task package and estimate the true labels and workers' strategies from their reports via Bayesian inference. Relying on the above estimates, a reinforcement learning algorithm is used to uncover how workers respond to different levels of offered payments. We determine the rewards based on workers' current strategies and the output of the reinforcement learning algorithm. We theoretically prove that our mechanism is able to incentivize workers to provide high-quality labels at equilibrium, which is the objective of peer prediction mechanisms. We also empirically show that our Bayesian inference algorithm can improve the robustness and lower the variance of payment, and our reinforcement learning algorithm performs consistently well under different worker models, such as bounded rational and self-learning workers.

To summarize, we improve the existing incentive mechanisms of microtask crowdsourcing in the following two aspects. Firstly, if there have been learning-based mechanisms proposed, we improve the mechanisms by analyzing the unique features or problems of microtask crowdsourcing. Secondly, if there is no learning-based mechanism, we design a new one which can not only correctly incentivize rational workers but also be adapted to bounded rational workers. Putting these two aspects together, we actually expect to build a more intelligent and more practical microtask crowdsourcing platform which incentivizes workers via learning workers' models.

1.3 Thesis Outline

The thesis consists of 6 chapters. The first two chapters introduce the background knowledge, research problems, and related studies. Chapters 3-5 present the three novel learning-based mechanisms we develop to help the requesters to handle the three core steps of microtask crowdsourcing: worker recruitment, task assignment and reward payment. Finally, Chapter 6 concludes this thesis and proposes future work. Below is a summary of each of the rest chapters in this thesis:

- Chapter 2. **Literature review:** It surveys the existing mechanisms developed for microtask crowdsourcing systems and analyzes their deficiencies.
- Chapter 3. **Optimal Posted-Price Mechanism:** In this chapter, the optimal posted-price mechanism for microtask crowdsourcing is proposed. It can improve the revenue of the requester to the utmost by fully exploiting the unique features of microtask crowdsourcing. It also avoids the need for prior knowledge on the price range. Theoretical analysis and experimental results confirm that, compared with existing pricing mechanisms, our posted-pricing mechanism can provide the optimal performance with the least prior knowledge.
- Chapter 4. **Active Task Assignment Mechanism:** In this chapter, we propose a novel task assignment mechanism based on active learning techniques. It improves existing mechanisms by solving the problems of insufficient label accuracy, low computation efficiency and the lack of flexibility in incorporating different worker models. The extensive experiments on four popular worker models and four real-world datasets have confirmed our advantages.
- Chapter 5. **Inference Aided Reinforcement Learning Reward Mechanism:** In this chapter, we propose the inference aided reinforcement learning mechanism to fully exploit the information contained in the collected labels and relax the assumption for fully rational workers. Theoretical analysis ensures that, for fully rational workers, our mechanism can correctly incentivize them to elicit high efforts and report truthfully. Experimental results then show that our mechanism can improve the robustness and lower the variance of rewards. Meanwhile, our mechanism can perform consistently well under different worker models, such as bounded rational and self-learning workers.

- Chapter 6. **Conclusions and future work:** It concludes the thesis and indicates potential future research directions.

Chapter 2

Literature Review

Nowadays, microtask crowdsourcing has developed as an important tool for requesters to efficiently and economically handle thousands of small tasks [80, 89]. To leverage the power of microtask crowdsourcing, a requester often needs to go through three steps: worker recruitment, task assignment and reward payment. Current microtask crowdsourcing platforms still rely on the requesters themselves to decide everything in the above three steps [79]. This is undesirable because a requester usually faces thousands of workers at the same time. To remove the requesters' hands from the wheel, in the literature, researchers have proposed various mechanisms to accomplish these three steps and improve the revenue of the requesters. In this way, we can not only facilitate the usage of microtask crowdsourcing in practice but also further enhance the advantages of microtask crowdsourcing.

In this chapter, we focus on reviewing the existing mechanisms used in the three steps of microtask crowdsourcing. More specifically, we first survey the pricing mechanisms used in the worker recruitment of microtask crowdsourcing and point out their shortcomings in exploiting the unique features of microtask crowdsourcing and requiring prior knowledge about the price range. Second, we review the existing task assignment mechanisms and discuss their deficiencies in label accuracy, computation efficiency and the flexibility to incorporate different worker models. At the third step, we move our focus to the reward payment step and analyze the problems of the traditional reward mechanisms on exploiting the information of collected labels and requiring the impractical assumption about fully rational workers.

2.1 Pricing in Worker Recruitment

Current crowdsourcing platforms, such as Amazon Mechanical Turk and Microworkers, offer limited capability to the requester in designing the pricing policies, mostly limiting them to a single fixed price. In this case, the only way to set prices is to estimate workers' willing-to-accept prices via a sufficient market analysis and then compute an optimal fixed price which would maximize the utility of the requester. However, this kind of market surveys usually is not cheap, which severely weakens the advantages of microtask crowdsourcing [79]. Besides, the nature of the crowdsourcing market is online and dynamic, and it is difficult to conduct market surveys for every task package posted by data requesters because the value of the task package is often not very high. In this case, the static and costly offline survey is not that meaningful. Thus, previous researchers propose different mechanisms to dynamically adjust the price of tasks, such as bargaining between requesters and workers to minimize work [30] and workers bidding their willing-to-accept prices for tasks [78, 98, 9]. The idea behind these mechanisms is that truthfully reporting the willing-to-accept prices is the utility-maximizing strategy for workers. However, in microtask crowdsourcing, due to the large number of workers, the above architecture will significantly increase the communication burden. Besides, the worker may not trust the requester and understand the mechanism to reveal their willing-to-accept prices or they do not have an explicit value for their willing-to-accept prices, which is a typical type of bounded rationality [67].

Instead of requiring workers to report their willing-to-accept prices, recently, researchers propose a more natural mechanism where workers come in a sequential manner and are offered a take-it-or-leave-it price offer. The pricing mechanism adjusts the offered price according to the accept-it-or-reject-it decisions of workers. We call this kind of pricing mechanisms posted-price mechanism. In [5] and [79], posted-price mechanisms are proven to have competitive and even better performance than the aforementioned bidding mechanisms. Besides, in posted-price mechanisms, it is naturally ensured that truthfully reporting the decisions is the utility-maximizing strategy for all workers. To achieve the design, existing posted-price mechanisms assume that a worker accepts a task if the offered price is higher than the cost. Without prior knowledge about workers' costs, these mechanisms learn the cost distribution online by counting the acceptance frequency. To maximize requesters' revenue under the uncertainty about workers' costs, different price selection algorithms have been employed. Badanidiyuru *et al.* [5] discretize the range of workers' costs into a geometric progression, and use a heuristic policy to pick the price from the progression. Long *et al.* [85] first use a part of the budget to randomly explore workers' responses and then greedily follow the optimal strategy learned based on workers' responses in the exploration strategy. Singla and Krause [79] then propose an improved policy which selects the price with the highest upper confidence bound of the expected revenue under the budget constraint. By doing so, Singla and Krause avoid the undirected random exploration stage, which lowers the regret from $O(B^{2/3})$ to $O(\log(B))$.[1] Furthermore, Badanidiyuru *et al.* [6] and Agrawal and Devanur [1] propose to employ linear programming (LP) to choose the optimal price. The LP problem is formulated with the upper and lower confidence bounds of the expected revenue and the budget cost of all possible prices.

To summarize, when a worker comes, the existing mechanisms estimate the expected revenue for all possible prices using the learned worker model, to choose the optimal price with respect to that expectation. However, the learned worker model may be inaccurate, causing the inaccuracy of the estimated revenue. Besides, an essential condition of this approach is that the range of possible prices has to be finite. Its accuracy will become even lower for a larger range of possible prices. Thereby, a finite prior price range must be provided. In contrast, in the mechanism proposed in Chapter 3, we only compare the estimated number of workers willing to accept the task with the accurately known number of workers allowed by the limited budget [35, 34]. Since one variable in our comparison is known in advance, we can avoid the complex comparison in the existing mechanisms which compares the expected

[1] The $O(B)$ item in [79] exists because the price is discrete and cannot reach the real optimal price. This is not considered in other studies. For fair comparison, we should not include this item.

revenue estimations corresponding to different prices. This feature makes the optimal price computed by our mechanism more accurate. Meanwhile, since our mechanism only cares about the minimum price at which the estimated number of workers willing to accept the task equals the number of workers allowed by the limited budget, there is no need to have a proper price range given in advance, which greatly facilitates the practical use.

2.2 Task Assignment

In microtask crowdsourcing, the tasks are often tedious and workers are non-experts. Thus, the resulting labels can be very noisy and how to improve the label accuracy is always a key topic of microtask crowdsourcing studies [37, 12]. To improve the label accuracy, many microtask crowdsourcing platforms follow the round-robin repeated labeling strategy which randomly assigns multiple workers with a same task and all tasks are assigned with a same number of workers [71]. In principle, as long as a sufficient number of labels is collected, this repeated labeling strategy can generate labels with high accuracy. However, we need to pay for each label we collect from workers. The cost of having a great number of redundant labels is non-trivial. In addition, the round-robin strategy overlooks the difference between different tasks and workers, and thus will waste lots of budget on easy tasks andB low-quality workers.

To achieve high label accuracy with fewer labels and thus lower the labeling costs, in many previous studies, researchers propose to actively select tasks and workers at each step based on the online inference of true labels and worker models. By doing so, more budget can be allocated to more difficult tasks and more reliable workers, and then the accuracy growth rate gets boosted. In this thesis, we call this approach active task assignment. The study on active task assignment starts from Sheng *et al.* [71]. They assume workers to be homogeneous and infer the true labels of tasks using majority voting. Their task assignment relies on the uncertainty sampling strategy which randomly selects a worker to label the most uncertain task at each step [45]. Then, Welinder and Perona [92] consider heterogeneous workers. They use the variational inference, a classic machine learning technique [86], to estimate true labels and worker models. Since variational inference can learn the estimates about the quality of workers' labels, they extend uncertainty sampling by excluding those workers whose quality measurements are very low at each step.

Later on, Simpson and Roberts [76] and Muhammadi *et al.* [59] employ another more advanced strategy: expected error reduction [70]. Compared with uncertainty sampling which purely relies on the current inference results, the expected error reduction predicts the benefits of all possible next-step assignments. At each step, the most beneficial assignment is selected. In principle, this prediction-based strategy can achieve higher accuracy growth rate than uncertainty sampling. However, to compute all predictions, it needs to traverse all possible future labels corresponding to each possible assignment. For each possible label, it then needs to run the inference algorithm which also needs to repeatedly traverse all tasks and workers. Thus, for a market with hundreds of tasks and workers, the computation cost of this strategy will become prohibitive. To alleviate the computation cost, Zheng *et al.* [100] propose a more efficient way for prediction. They force the estimates of worker models to be fixed and approximately update the estimates of true labels using Bayes' theorem. Nevertheless, this approximation may bring large errors since it neglects the correlation between the estimates of true labels and worker models.

Both uncertainty sampling and expected error reduction come from the active learning studies which assume there is one fully reliable annotator [70]. By contrast, microtask crowdsourcing only has many possibly unreliable workers. Considering the uncertainty of labels in microtask crowdsourcing, Chen *et al.* [12] propose to use the optimistic upper confidence bounds to replace the expected values for prediction. However, their strategy is developed for the market where workers are homogeneous and only the uncertainty of true label estimates needs to be considered. In a practical market, workers are heterogeneous. Both the estimates of true labels and worker models have a certain level of uncertainty. Even worse, the uncertainties in two aspects might be mutually reinforced. The incorrect estimates of true labels will cause the worker models to be wrong. The wrong worker models will in turn drive the inference to trust the incorrect estimates of true labels more. This iteration repeats, which will cause all estimates to be completely wrong. This problem is very severe when there are just few labels at the beginning stage. Due to the unique problems of microtask crowdsourcing on uncertainty, the existing task assignment approaches cannot improve the label accuracy to the utmost. Besides, the existing studies on microtask crowdsourcing inference often develop their algorithms by assuming one specific worker model [99]. Thus, if we directly apply these algorithms, our task assignment mechanism will be bound with a certain worker model. When changing to another model, we will always have to redevelop the whole framework, which is very inconvenient. In other words, existing task assignment approaches lack the flexibility to incorporate different types of worker models. Thus, in Chapter 4, we develop a novel active task assignment mechanism to improve label accuracy, computation efficiency and model flexibility [36].

Note that, in addition to the above research line, there are many other studies on task assignment in microtask crowdsourcing. For example, Lin *et al.* [47], Kamar *et al.* [40] and Fan *et al.* [19] focus on how to improve task assignment by incorporating the features and machine learning classifier of tasks. In fact, our research in Chapter 4 can be beneficial to them because the benchmark active learning strategies in our study are widely-adopted in their studies. It will be one of our future studies to incorporate the contextual information of tasks in task assignment. Ho *et al.* [29], Parameswaran *et al.* [62], and Tarable *et al.* [84] learn worker models using golden tasks of which the true labels are known in advance. The golden tasks can also be used to improve the learning of worker models.

2.3 Reward Payment

In addition to assigning more tasks to more reliable workers, another direction to improve the label accuracy in microtask crowdsourcing is to adjust the rewards for workers based on the analysis on the collected labels. The objective is to pay more rewards to workers who provide higher-quality labels. By doing so, we expect to truthfully elicit high-quality labels from strategic workers. However, in microtask crowdsourcing, the ground-truth labels often are either either unavailable or too costly to obtain, which makes the evaluation of workers' contributions very difficult. To solve this problem, a class of mechanisms, collectively called peer prediction [63, 26, 39, 94, 64, 15], has been proposed in the literature.

Peer prediction was first proposed by Miller *et al.* [57]. They prove that any strictly proper scoring rule [26] can be used to decide the rewards in peer prediction mechanisms. In their peer prediction mechanism, they assume that the requester knows the joint distribution of the observed labels and this distribution to be the common knowledge to all workers. Under these assumptions, the scoring rule is calculated by comparing the labels of the targeted worker and another randomly selected reference worker. Their mechanism ensures that any worker can only get the maximal rewards by truthfully reporting the observed label given all the other workers truthfully report their labels. However, a salient drawback of their mechanism is that, if all workers randomly report the labels without observing the true labels, the workers can even get higher rewards. To relax the assumption that the requester knows the joint distribution of the observed labels, Prelec [63] propose another elegant mechanism, Bayesian Truth Serum, which asks each worker to report not only the observed labels but also a prediction on the distribution of other workers' reports. The Bayesian Truth Serum mechanism compares not only the labels between two workers but also the distribution predictions. Even though the assumption for the requester is relaxed, this mechanism still requires the joint distribution of the observed labels is common knowledge for all workers. Besides, the uninformative equilibrium, where all workers randomly report the labels and predictions, is still an unsolved problem. Thus, quite a few follow up peer prediction mechanisms have been proposed to further relax the assumptions [64, 94] and to get rid of the uninformative equilibrium [39].

In the above studies, the only factor that affects workers' reporting strategies is the rewards. More recently, Witkowski *et al.* [93] and Dasgupta and Ghosh [15] formally studied more practical situations where workers need to exert higher efforts to generate higher quality labels and the higher efforts will lead to higher costs of workers. In other words, workers' reporting strategies are affected by not only the rewards but also their own costs. In the peer prediction mechanism proposed by Dasgupta and Ghosh [15], the equilibrium, where every agent first exerts maximum efforts to obtain high-quality observations and then truthfully reports them, brings the highest utility to all workers. Thus, it is not sensible for workers to all randomly report the labels, which gets rid of the uninformative equilibrium. Based on their mechanism, Shnayder *et al.* [73] and Kong and Schoenebeck [41] further extend the results from binary labels to labels with more than two possible values.

Despite the attractive property of the recently proposed peer prediction mechanisms, they still have two undesirable deficiencies. Firstly, no mater how complex the formulations of these peer prediction mechanisms are, they still decide the rewards by comparing the labels of the targeted worker with the labels of another randomly selected reference worker. However, from the machine learning studies on microtask crowdsourcing [99], we know that the collected labels contain a wealth of information about the true labels and worker models. To be more specific, the "cross validation" between the labels provided by different workers for the same task can help us to find the real true label. Meanwhile, the real true label will also be able to help us to evaluate the performance of each worker. These mechanisms cannot fully exploit the information contained in the collected labels, which will increase the variance and lower the robustness of rewards. To solve this problem, Liu and Chen [50] propose to train a supervised learning model (e.g. the neural network) with the collected labels and then use the outputs of the trained model as the reference labels. However, the supervised learning models rely on the contextual information of tasks which is usually not easy to get. Besides, training the supervised learning model for thousands of tasks is often very time-consuming, and we cannot let workers wait for 10 hours to get the rewards. Thus, in Chapter 5, we combine the peer prediction mechanism with Bayesian inference, an unsupervised learning algorithm, which is computationally very efficient and only uses the collected labels as inputs [33, 32]. By doing so, we can not only achieve all the attracting properties of the state-of-the-art peer prediction mechanisms but also efficiently compute the rewards with lower variance and higher robustness. Secondly, these peer prediction mechanisms simplify workers' responses to the incentive mechanism by assuming that workers are all fully rational and only follow the utility-maximizing strategy. However, some evidence exists that human agents may follow bounded-rationality model, and may improve their responding strategies gradually in practice [75, 11, 21]. To adapt peer prediction mechanisms to workers who may not be fully rational, Liu and Chen [51] propose to adjust the mechanism in an online fashion with a bandit algorithm. However, the bandit algorithm, which requires a static environment, will fail to learn the optimal settings when workers adjust their strategies according to the interaction with the mechanism. Thus, in Chapter 5, our another contribution is to develop a novel reinforcement algorithm to adjust our reward mechanism. In this way, we can analyze workers' response patterns to the offered rewards and then adapt our mechanism to different types of workers.

2.4 Summary

To summarize, this chapter extensively reviews the existing mechanisms in the task pricing, task assignment and reward payment of microtask crowdsourcing systems. We point out the challenging problems in the existing learning-based mechanisms is that they simply apply the classic machine learning techniques, which overlooks the unique features or problems of microtask crowdsourcing. For the reward mechanisms, they not only still require the impractical assumption that workers are fully rational but also cannot fully exploit the information contained in the collected labels. Thus, in the following three chapters, we propose three novel mechanisms, each for one step, by developing machine learning algorithms specifically for the mechanism design in microtask crowdsourcing. Putting these mechanisms together, we expect to build a novel microtask crowdsourcing system which is more intelligent and more practical in managing the large crowd of workers.

Chapter 3

Optimal Posted-Price Mechanism

This chapter focuses on developing the optimal posted-price mechanism for the worker recruitment process of microtask crowdsourcing. We propose a novel posted-price mechanism to exploit the unique features of microtask crowdsourcing—higher task prices can attract more workers from the market, but on the other hand, higher prices allow fewer workers to be recruited due to the limited budget [35, 34]. More specifically, we first convert the pricing problem into an equivalent multi-armed bandit (MAB) problem. Then, we develop an algorithm that offers each coming worker the minimum price at which the anticipated number of workers willing to accept the task approximately equals the number of workers allowed by the budget. Due to the monotonicity features of the numbers of workers that can be attracted and are allowed to be recruited, our algorithm is theoretically proven to be optimal. In addition, these features ensure that our algorithm will never explore overly high prices and thus does not need to set a price range in advance.

The following sections of this chapter are arranged as follows. In section 3.1, we formally describe the model of the worker recruitment process and the unique features of microtask crowdsourcing. Then, in Section 3.2, we present the equivalent MAB problem and the proposed posted-price mechanism. We put all the theoretical analysis in Section 3.3, including the general performance upper bound and the performance of our mechanism. In Section 3.4, a set of experiments on real-world price data is conducted to further verify the advantages and practicability of our mechanism. Finally, Section 3.5 summarizes this chapter. In addition, we list the key notations used in this chapter as Table 3.1.

Table 3.1: Key notations used in Chapter 3

B	Recruitment budget	N	Total number of workers
p	Price offered to a worker	p^*	Optimal price
S	Set of possible prices	p_i	The i-th price in S
$F(p)$	The probability that a worker accepts price p	U	Objective function (the number of recruited workers)
k^*	Subscript of the optimal price	μ_i	Average acceptance rate of p_i
$\hat{k}$	Minimum POP based on μ_i	b	Upper confidence bound
R^π	Regret of any algorithm π	$\tilde{R}$	Regret of our algorithm
δ_p	Minimum gap between prices	K_r	Subscript of the highest price
A_1	Set of steps that $\hat{k} < k^*$	A_2	Set of steps that $\hat{k} > k^*$
A_3	Set of steps that chooses p_{k^*-1} but satisfies $\hat{k} = k^*$		
C_i	The number of workers allowed by the budget ($= B/(N \cdot p_i)$)		

3.1 Modeling of Worker Recruitment

In the worker recruitment process of microtask crowdsourcing, a requester posts tasks along with the price for each task. Workers can accept or reject the task. If they accept and finish one task, they will be paid the offered price. In this section, we formulate the requester and worker models, and study the unique features, optimal price and performance metric of microtask crowdsourcing.

3.1.1 Requester and Worker Models

The requester gets benefits from the completed tasks. In this chapter, we assume that each task has unit value. Thus, the requester wishes to maximize the number of completed tasks within the given budget B. Assume there are N workers in the market and workers will finish the task if accepting it.

The worker model depicts how workers decide to accept or reject the task. Here, we give three typical worker models which will be utilized in our experiments:

- Private Cost Model [79] assumes that worker w accepts a task only if the offered price p_i is not lower than the cost c_w for performing the task.

- Discrete Choice Model [23] describes human's preference for tasks with higher utility. It assumes worker w accepts price p_i with the probability

$$\mathbb{P} \propto \exp[u(p_i)]/\{\exp[u(p_i)] + r_w\}$$

where r_w denotes the effects of other tasks. $u(p_i) = \alpha_w p_i + \beta_w$ denotes the utility of performing the offered task. The detailed discussion about how to learn parameters α_w and β_w from the real-world data can be found in [23].

- Reference Payment Model [96] depicts human's habit to maintain a reference payment level r_w. It assumes worker w accepts price p_i with the probability

$$\mathbb{P} \propto \{1 + \exp[-\alpha_w - \beta_w(p_i - r_w)]\}^{-1}$$

where α_w and β_w denote worker i's interest and activeness, respectively. Yin and Chen detailedly discussed the three parameters, α_w, β_w and r_w, in [96].

Note that we list three worker models here just to show the possible ways that workers make decisions, and in the literature, we cannot find more worker models. The mechanism we propose later actually does not make any assumption about workers' decision-making process. In other words, our mechanism is independent of the worker model, and how to set the model parameters does not affect our mechanism.

In this chapter, we assume the stochastic and sequential arrival of workers. This setting is widely-adopted in the research on the pricing mechanisms of microtask crowdsourcing [79]. Under this assumption, workers arrive one at a time, and the decision parameters of different workers (e.g. the cost c_i) are the same or independently and identically sampled from a same distribution[1]. Thus, the input-output relationship of the worker models mentioned above can be described by a probability function $F(p)$. It denotes the probability at which the coming worker accepts the offered price p. Since prices in practice must be positive, we require the support of $F(p)$ to be $(0, +\infty)$ (i.e. $F(p) = 0$ for $p \leq 0$). Note that the stochastic arrival assumption

[1] Our mechanism is based on without any prior knowledge about the distribution. Thus, we do not limit the specific form of the distribution. In other words, any distribution can be applied to generate the parameters in worker' models, and it will not affect our mechanism.

may be violated in real crowdsourcing markets, and the number of workers N is also always changing. Therefore, in our empirical experiments, we evaluate the robustness of our mechanism in more practical settings where these factors are all considered.

3.1.2 Unique Features, Optimal Price and Metric

In the worker recruitment process, our objective is to use the limited budget to recruit as many workers as possible. This objective is important because, in microtask crowdsourcing, we need to let one task be labeled by multiple workers and improve label accuracy by comparing the labels of different workers. In this case, recruiting sufficient workers is the foundation of high-quality labels. On the other hand, the budget of microtask crowdsourcing is usually very limited. Thus, how to use the budget with the highest efficiency is very important. Formally, if we choose a price p for all tasks, the requester's objective will be

$$U(p) = \min\{N \cdot F(p), B/p\}. \tag{3.1}$$

The first item denotes the expected number of workers accepting the price, and the second item represents the budget constraint. Then, we can summarize the features of the pricing problem in microtask crowdsourcing:

- Higher prices attract more workers — i.e. the acceptance probability function $F(p)$ is monotonically increasing;
- Higher prices allow fewer workers to be recruited — i.e. the budget constraint B/p is monotonically decreasing;
- The optimal price $p^* = \arg\max_p U(p)$ is the point where the budget equals the costs to recruit all workers willing to accept the price, i.e. $N \cdot F(p^*) = B/p^*$.

Besides, the available prices in practice should be discrete (e.g. 1, 2, ... cents), and the optimal price p^* may be unavailable. Thus, we write the possible prices as an increasing sequence ($p_1 < p_2 < \ldots$), and denote the available optimal price with the optimal subscript $k^* = \arg\max_k U(p_k)$.

We also introduce a performance metric for posted-price mechanisms, termed regret. It denotes the gap between the mechanism and taking the optimal price p_{k^*}:

$$R(N, B) = U(p_{k^*}, B, N) - U(B, N) \quad (3.2)$$

Maximizing the objective function $U(B, N)$ is thus equivalent to minimizing the regret. Note that the stochastic combination of different prices may generate higher values for the objective function than the optimal price p^* [6]. Nevertheless, finding the optimal combination requires much more accurate $F(p)$ than identifying the single optimal price. Thus, the mechanisms targeting the optimal combination need to explore sub-optimal prices more, and their performance turns out to be worse than ours in experimental evaluation.

3.2 Indirect Design of Posted-Price Mechanism

When selecting prices, we face a dilemma between exploring possible better prices and exploiting the current best price. A general idea to resolve this dilemma is to compute and select the current optimal price while considering the uncertainties of the learned worker model. However, it is not easy to properly measure the uncertainties. In addition, budget constraint is another factor that needs attention. The requester accumulates benefits by recruiting workers, and the recruitment stops when the budget is exhausted. The stopping point is in fact not fixed due to the change of selected prices, making it difficult to decide the current optimal price. Thus, designing a mechanism that can also exploit unique features of microtask crowdsourcing becomes even more challenging.

We adopt an indirect approach to design our mechanism. More specifically, instead of directly solving the pricing problem of microtask crowdsourcing, we define an equivalent multi-armed bandit (MAB) problem at first. It not only has the same optimal solution as microtask crowdsourcing but also inherits all the unique features. The only difference is that this equivalent problem has a fixed stopping point which simplifies computing the current optimal. For this equivalent problem, we then develop an algorithm which optimally exploits the unique features of microtask crowdsourcing and use it as the core of our mechanism. For clarity, all the theoretical analysis, including the optimality of our MAB algorithm, the linkage between the two problems on regret and the regret under the infinite price range, will be provided in Section 5.

3.2.1 Equivalent MAB Problem

We first define the equivalent MAB problem using the notations explained in microtask crowdsourcing. Suppose that we need to repeatedly select a price $p_i \in \{p_1, \ldots, p_K\}$ for N times. After selecting the price p_i in round n, we can observe a stochastic signal X_n which follows the Bernoulli distribution. The mean of X_n equals to $F(p_i)$. Meanwhile, the reward for choosing p_i is $\tilde{U}_i = \min\{F_i, C_i\}$, where $F_i = F(p_i)$ and $C_i = B/(N \cdot p_i)$. The rewards are accumulated in background, so we can only know the rewards after finishing all the N selections. In fact, this new problem just equally divides the objective function in microtask crowdsourcing into N rounds, i.e. $\tilde{U}_i = U(p_i)/N$. This conversion helps us bypass the difficulty to predict the stopping point that the budget is exhausted, when trying to exploit the unique features of microtask crowdsourcing. It is also easy to conclude that this equivalent problem inherits all the unique features of microtask crowdsourcing because F_i and C_i directly come from the acceptance probability function and the budget constraint, and are monotonically increasing and deceasing, respectively. In this case, the equivalent MAB problem also satisfies the third unique feature of the pricing problem discussed above—i.e. the optimal price should be the intersection point. Thus, the optimal arm of this MAB problem should also be p_{k^*}, namely, the optimal price in the pricing problem.

Based on the three unique features, we introduce the following two kinds of possible optimal prices (POPs):

- POP-1 is the price p_k that satisfies $C_k > F_k \geq C_{k+1}$;
- POP-2 is the price p_k that satisfies $F_k \geq C_k > F_{k-1}$.

Here, $k \in \{1, \ldots, K\}$. For completeness, we add the conventions that $p_0 = 0$ and $p_{K+1} = +\infty$. The rationale behind introducing POPs lies in the following theorem:

Theorem 3.1 *There is always one and only one POP-1 or POP-2. The optimal price p_{k^*} must be POP-1 or POP-2.*

[Proof of Existence:] Due to the monotonicity of $F(p)$ and B/p, $F_i - C_i$ is monotonically increasing when i changes from 1 to K. Meanwhile, the conventions, $p_0 = 0$ and $p_{K+1} = +\infty$, lead to $F_0 - C_0 < 0$ and $F_{K+1} - C_{K+1} > 0$, respectively. Therefore, there must exist k satisfying $F_k - C_k < 0$ and $F_{k+1} - C_{k+1} \geq 0$. In this case, if $F_k \geq C_{k+1}$, then p_k is POP-1; otherwise, p_{k+1} is POP-2.

[Proof of Uniqueness:] Let p_k be POP-1. Then, for $\forall i < k$, $F_i \leq F_k < C_k \leq C_{i+1} < C_i$. For $\forall i > k$, $F_i \geq F_{i-1} \geq F_k \geq C_{k+1} \geq C_i$. Thus, $p_{i \neq k}$ cannot be POP-1 or POP-2. Besides, let p_k be POP-2. Then, for $\forall i < k$, $F_i \leq F_{k-1} < C_k \leq C_{i+1} < C_i$. For $\forall i > k$, $F_i \geq F_k \geq C_k \geq C_{i-1} > C_i$. Thus, $p_{i \neq k}$ also cannot be POP-1 or POP-2.

[Proof of Optimality:] Let p_k be POP-1 or POP-2. From the uniqueness proof, we can have $F_i \leq \min\{C_k, F_k\}$ for $\forall i < k$ and $C_i \leq \min\{C_k, F_k\}$ for $\forall i > k$. Therefore, $\tilde{U}_k \geq \tilde{U}_{i \neq k}$, and p_k equals to the optimal price p_{k^*}.

3.2.2 Optimal MAB Algorithm

Algorithm 1: Optimal MAB Algorithm (OA-MAB)

Input: n, $S = \{p_1, \ldots, p_K\}$, $\mu_j(n)$ for $i = 1 \ldots K$

Search for the minimum POP in S based on $\mu_i(n)$;

Output the current optimal price $p_{k(n)}$ at which:

$$k(n) = \begin{cases} \hat{k} & \text{if } p_{\hat{k}} \text{ is POP-1,} \\ \hat{k} & \text{if } p_{\hat{k}} \text{ is POP-2 and } \frac{l_{\hat{k}}-1}{2} \in \mathbb{N} \\ \hat{k} & \text{if } p_{\hat{k}} \text{ is POP-2, } \frac{l_{\hat{k}}-1}{2} \notin \mathbb{N} \text{ and } b_{\hat{k}-1} < C_{\hat{k}} \\ \hat{k}-1 & \text{if } p_{\hat{k}} \text{ is POP-2, } \frac{l_{\hat{k}}-1}{2} \notin \mathbb{N} \text{ and } b_{\hat{k}-1} \geq C_{\hat{k}} \end{cases}$$

where $\hat{k}$ is the subscript of the minimum POP, and the upper confidence bound b is defined in Equation 3.4.

Here we describe the optimal algorithm for the MAB problem, a novel price selection algorithm with the regret proven to match the Lai-Robbins regret lower bound in Section 3.3.1. To formally describe the algorithm, we need the following notations.

Let $N_i(n)$ denote the number of times that price p_i has been selected up to round n. The empirical estimate of the acceptance probability F_i in round n is

$$\mu_i(n) = \frac{1}{N_i(n)} \sum_{t=1}^{n} 1\{k(t) = i\} X_t \tag{3.3}$$

where $k(t)$ denotes the subscript of the selected price in round t, and we set $\mu_i(n) = 1$ if $N_i(n) = 0$. Besides, we employ the KL-divergence to compute the upper confidence bound of the estimate as:

$$b_i(n, \mu_i, N_i) = \sup\{q \geq \mu_i(n) : N_i(n) \cdot KL[\mu_i(n), q] \leq \log(n) + 3\log(\log(n))\} \tag{3.4}$$

with the convention that $b_i(n, \mu_i, 0) = 1$ and $b_i(n, 1, N_i) = 1$. Here,

$$KL[x, y] = x \log(\frac{x}{y}) + (1 - x) \log(\frac{1 - x}{1 - y})$$

denotes the KL-divergence between two Bernoulli distributions whose means are x and y respectively. We adopt this upper confidence bound formulation because [24] prove it to outperform other formations in Bernoulli distributions. Furthermore, we define $l_i(n)$ as the number of times that p_i has been identified as not only the minimum possible optimal price (being POP-1 or POP-2) but also POP-2. The pseudo-code of the algorithm is presented in Algorithm 1.

3.2.3 Posted-Price Mechanism

Our posted-price mechanism is presented in Algorithm 2. The input parameters include the budget B, the number of workers N and the minimum price gap δ_p. Here, δ_p is a parameter of the microtask crowdsourcing platform and denotes the minimum payment unit (e.g. 1 cent). Due to the search of the minimum possible optimal price in our MAB algorithm, the probability of exploring overly large prices exponentially decreases. We call this property probability decay and will further explain it in Section 3.3.3. This special property ensures the performance of our MAB algorithm to be unaffected when $K \to +\infty$. Thus, our mechanism does not need the input of a finite price range, which is required by other existing mechanisms. Furthermore, to strictly ensure the offered price is not higher than the remaining budget B_n, we use B_n to update the available price set S in our mechanism.

Algorithm 2: Optimal Posted-Price Mechanism (OPPM)

Parameters: B, N, δ_p
Initialize: worker counter: $n = 1$; available budget: $B_n = B$;
the number of recruited workers: $U = 0$;
the number of available prices: $K = \lfloor B/\delta_p \rfloor$;
the estimate of F: $\mu_i = 1$ for $i = 1 \ldots K$;
begin
  while $B_n > \delta_p$ *and* $n \le N$ **do**
    $K \leftarrow \lfloor B_n/\delta_p \rfloor$, $S \leftarrow \{\delta_p, 2\delta_p, \ldots, K\delta_p\}$;
    $p_k \leftarrow$ OA-MAB$(n, S, \mu_1, \ldots, \mu_K)$;
    Offer price $p_{k(n)}$ to the coming worker;
    Observe the decision: $X_n = 0$ (reject) or 1 (accept);
    Update: $U \leftarrow U + X_n$; $B_{n+1} \leftarrow B_n - p_k X_n$;
      $\mu_k \leftarrow (N_k \mu_k + X_n)/(N_k + 1)$
      $N_{k(n)} \leftarrow N_{k(n)} + 1$;$n \leftarrow n + 1$

3.3 Theoretical Analysis

In this section, we provide extensive theoretical analysis to support our indirect approach to designing the posted-price mechanism.

3.3.1 Optimality of the MAB Algorithm

To prove the optimality of Algorithm 1, we firstly derive a general bound for the MAB problem defined in Section 3.2.1, the Lai-Robbins regret lower bound [44]. Since we expect for smaller regret, this classic lower bound represents the possible best performance that can be achieved by any uniformly good algorithm. We say an algorithm π is uniformly good if its regret is at most $O(\log N)$ for all possible $F(p)$. Besides, we keep $B/N = const$ when $N \to +\infty$. This setting ensures the optimal price p_{k^*} to be unchanged and can greatly facilitate the asymptotic analysis.

Theorem 3.2 *In the MAB problem defined in Section 3.2.1, any uniformly good algorithm π satisfies:*

$$\liminf_{N\to\infty} \frac{R^\pi(N)}{\log(N)} = \begin{cases} 0 & p_{k^*} \text{ is POP-1} \\ \frac{C_{k^*} - F_{k^*-1}}{KL[F_{k^*-1}, C_{k^*}]} & p_{k^*} \text{ is POP-2} \end{cases} \tag{3.5}$$

Proof: *To prove this theorem, we firstly employ the Theorem 1 in the classic studies of Graves and Lai [28][2] and can get*

$$\lim_{N\to\infty} \inf\left[R^{\pi}(N)/\log(N)\right] \geq w(F)$$

where $w(F)$ equals the output of the following LP problem:

$$\begin{aligned} \min \quad & \textstyle\sum_{j=1}^{K} w_j \cdot [\tilde{U}_{k^*}(F) - \tilde{U}_j(F)] \\ s.t. \quad & \inf_{\check{F}\in Z(F)} \textstyle\sum_{j\neq k^*} w_j \cdot KL[F_j, \check{F}_j] \geq 1 \end{aligned}$$

where $w_j \geq 0$ and $\check{F}$ denotes the bad distribution that has the same value as F at k^* but provides the largest rewards at another price $\check{k} \neq k^*$. 1 in the right-hand side of the constraint ensures the regret $R^{\pi}(N)$ to be $[w(F) + O(1)] \cdot \log(N)$, for which the detailed explanation can be found in the Equation 2.5 of [28]. Even starting from k^*, any algorithm still needs to explore other prices to distinguish F and $\check{F}$. Otherwise, it may miss the real optimal price $\check{k}$ if we substitute F with $\check{F}$. All the bad distributions form the set $Z(F)$:

$$Z(F) = \{\check{F} | \check{F}_{k^*} = F_{k^*}, \tilde{U}_{k^*}(\check{F}) < \max_k \tilde{U}_k(\check{F})\}.$$

Secondly, we solve the LP with the monotonicity of $F(p)$ and $C(p)$ considered. The detailed deduction is as follows:

- *If p_{k^*} is POP-1, $Z(F) = \varnothing$ because*

  $$\begin{aligned} \tilde{U}_{k^*}(\check{F}) = \check{F}_{k^*} \geq \check{F}_{j<k^*} \geq \tilde{U}_{j<k^*}(\check{F}) \\ \tilde{U}_{k^*}(\check{F}) = \check{F}_{k^*} \geq C_{j>k^*} \geq \tilde{U}_{j>k^*}(\check{F}). \end{aligned}$$

  Thus, the minimal value is $w(F) = 0$.

- *If p_{k^*} is POP-2,*

  $$\inf_{\check{F}\in Z(F)} \sum_{j\neq k^*} w_j KL[F_j, \check{F}_j] = w_{k^*-1} \cdot KL[F_{k^*-1}, C_{k^*}]$$

  because:

  $$\check{F}_{k^*-1} \geq \max_j \tilde{U}_j(\check{F}) > C_{k^*},$$

[2] The Condition (2.14) for Theorem 1 in [Graves and Lai 1997] is equivalent to $F(p_(k^* - 1)) > 0$, which is satisfied in all cases.

$$\sum_{j\neq k^*} w_j KL[F_j, \check{F}_j] \geq w_{k^*-1} KL[F_{k^*-1}, \check{F}_{k^*-1}] > w_{k^*-1} KL[F_{k^*-1}, C_{k^*}],$$

and, for $\epsilon > 0$, constructing $\check{F}^{\epsilon}_j$ as $\check{F}^{\epsilon}_j = F_j$ for $j \neq k^* - 1$ and $\check{F}^{\epsilon}_{k^*-1} = C_{k^*} + \epsilon$ leads to $\check{F}^{\epsilon} \in Z(F)$ and

$$\lim_{\epsilon\to 0} \sum_{j\neq k^*} w_j KL[F_j, \check{F}^{\epsilon}_j] = w_{k^*-1} KL[F_{k^*-1}, C_{k^*}].$$

Thus, the minimal value is obtained at

$$w_{k^*-1} = 1/KL[F_{k^*-1}, C_{k^*}] \quad \text{and} \quad w_{j\neq k^*-1} = 0.$$

Combining the above two cases, we can conclude Theorem 3.2.

Then, let $\tilde{R}$ be the regret of Algorithm 1. The optimality of our algorithm is guaranteed by the following theorem:

Theorem 3.3 *The regret upper bound of Algorithm 1 equals the Lai-Robbins regret lower bound, namely*

$$\limsup_{N\to\infty} \tilde{R}/\log(N) = \liminf_{N\to\infty} R^{\pi}/\log(N). \tag{3.6}$$

To prove this theorem, we firstly decompose the rounds where sub-optimal prices are selected into the following sets:

$$\{n \leq N | k(n) \neq k^*\} \subset A_1 \cup A_2 \cup A_3, \tag{3.7}$$

where $A_1 = \{n|\hat{k}(n) < k^*\}$ and $A_2 = \{n|\hat{k}(n) > k^*\}$ denote the cases where POP is wrongly identified. $A_3 = \{n| \ \hat{k}(n) = k^*, k(n) = k^* - 1\}$ represents the cases where POP is correct but the price is wrongly selected. Next, based on this decomposition, we compute the regret of Algorithm 1 as:

Theorem 3.4

$$\tilde{R} \leq \mathbb{E}|A_1| + \mathbb{E}|A_2| + (C_{k^*} - F_{k^*-1}) \cdot \mathbb{E}|A_3| \tag{3.8}$$

where $|\cdot|$ denotes the size of a set.

Proof: *Considering the fact that $\tilde{U}_{k^*} \leq C_{k^*}$, $\tilde{U}_{k^*-1} = F_{k^*-1}$ and $0 < \tilde{U}_j \leq \tilde{U}_{k^*} \leq 1$ hold for $\forall j$, we can have*

$$\begin{aligned} R(B,N) &= U(p_{k^*}, B, N) - \textstyle\sum_{n=1}^{N} \tilde{U}_{k(n)} \\ &= (\textstyle\sum_{j<k^*-1} + \sum_{j>k^*} + \sum_{j=k^*-1})[\tilde{U}_{k^*} - \tilde{U}_j]\mathbb{E}N_j \\ &\leq \mathbb{E}|A_1| + \mathbb{E}|A_2| + (C_{k^*} - F_{k^*-1}) \cdot \mathbb{E}|A_3| \end{aligned}$$

where, N_j denotes the total times that p_j is selected.

Furthermore, we can bound $\mathbb{E}|A_1|$, $\mathbb{E}|A_2|$ and $\mathbb{E}|A_3|$ with:

Theorem 3.5 $\mathbb{E}|A_1| < +\infty$.

Proof: *According to the settings of Algorithm 1, we derive*

$$\left.\begin{array}{r} Q_j \stackrel{def}{=} \{n|\hat{k} = j\} \\ W_j \stackrel{def}{=} \{n|\hat{k} = j, k(n) = j\} \end{array}\right\} \Rightarrow |W_j| \geq |Q_j|/2$$

using the fact that p_j *is selected at least when* $\frac{l_j - 1}{2} \in \mathbb{N}$*. For* $j < k^*$*, to achieve* $\hat{k} = j$*,* μ_j *should satisfy* $\mu_j \geq C_{j+1}$*. Meanwhile,* $F_j < C_{j+1}$*. Thus,* $\mu_j > F_j$ *holds for all elements in* W_j*. According to Theorem 4.1 of [65],* $\mathbb{E}|W_j| < \infty$ *and* $\mathbb{E}|A_1| = \sum_{j<k^*} \mathbb{E}|Q_j| < +\infty$*, which concludes Theorem 3.5.*

Theorem 3.6 $\mathbb{E}|A_2| = O(\log\log(N))$.

Proof: *For any* $j > k^*$*, we can define the following sets:*

$$Q_j \stackrel{def}{=} \{n|p_j \textit{ is POP-1}\} \quad W_j \stackrel{def}{=} \{n|p_j \textit{ is POP-2}\}$$

Furthermore, we can decompose the set W_j *as:*

$$X_j \stackrel{def}{=} \{n \in W_j | N_{j-1}(n) \geq l_j(n)/4\}$$
$$Y_j \stackrel{def}{=} \{n \in W_j | N_{j-1}(n) < l_j(n)/4\}$$

where $l_j(n)$ *denotes the times that* p_j *is identified as the minimum POP and POP-2. Similar as Theorem 3.5, we can prove* $\mathbb{E}|Q_j| < +\infty$ *and* $\mathbb{E}|X_j| < +\infty$*. Then, we can derive*

$$Z_j \stackrel{def}{=} \{n \in W_j | b_{j-1}(n) < C_j\} \Rightarrow |Y_j| \leq 4|Z_j|$$

using the fact that p_j *is selected at least* $l_j(n)/4$ *times under the condition that* $(l_j - 1)/2 \notin \mathbb{N}$ *and* $b_{j-1}(n) < C_j$*. Since* $C_j < F_{j-1}$*, according to Theorem 2 of [24],* $\mathbb{E}|Z_j| \leq \gamma \log\log(N)$*, where* γ *is a constant. Thus,* $\mathbb{E}|A_2| = \sum_{j>k^*}(\mathbb{E}|Q_j| + \mathbb{E}|W_j|) = O(\log\log(N))$*, which concludes Theorem 3.6.*

Theorem 3.7 *If* p_{k^*} *is POP-1,* $\mathbb{E}|A_3| < +\infty$ *. If* p_{k^*} *is POP-2,* $\mathbb{E}|A_3| = O((1 + \epsilon)\log(N)/KL[F_{k^*-1}, C_{k^*}])$*, where* ϵ *can be any positive number [24].*

Proof: A_3 *denotes the case where* p_{k^*} *is identified as POP-2 in Algorithm 1 and*

k^*-1 is selected because $b_{k^*-1} \geq C_{k^*}$. If k^* is actually POP-1, we can know that $C_{k^*} > F_{k^*}$. However, to enter set A_3, our algorithm must wrong identify p_{k^*} as POP-2, which requires $\mu_{k^*} \geq C_{k^*}$. In other words, our algorithm must maintain a wrong estimation of F_{k^*}. Moreover, according to the settings of Algorithm 1, we can know $|A_3| \leq |H|/2$, where the set $H = n|\mu_{k^*} \geq C_{k^*}$. Similar as Theorem 3.5, we can drive $|H| < +\infty$ based on Theorem 4.1 of [65], which concludes $\mathbb{E}|A_3| < +\infty$. If k^* is POP-2, we can get $F_{k^*-1} < C_{k^*} \leq b_{k^*-1}$. According to Theorem 2 of [24], $\mathbb{E}|A_3| = O((1+\epsilon)\log(N)/KL[F_{k^*-1}, C_{k^*}])$.

Using Theorems 3.4∼3.7, we can finally conclude Theorem 3.3.

3.3.2 Regret of Our Mechanism

Since the stopping point of posted-price mechanisms is not fixed, the regret of our mechanism has a little difference with the regret of the proposed MAB algorithm (Theorem 3.4). Thus, we here derive the regret upper bound of our mechanism as the following Theorem 3.8.

Theorem 3.8 *The expected regret of our mechanism satisfies:*

$$\begin{aligned} R(N,B) \leq & 1 + \mathbb{E}|A_1| + (F_{k^*} - F_{k^*-1})\,\mathbb{E}|A_3| \\ & [K_r\delta_p - F_{k^*}p_{k^*}] \cdot \mathbb{E}|A_2|/p_{k^*} \end{aligned} \tag{3.9}$$

where $K_r > k^*$ denotes the subscript of the highest price explored by our mechanism.

Proof: *The expected utility of our posted-price mechanism satisfies*

$$U(M,B,N) = \sum_{i=1}^{K_r} \mathbb{E}[N_i] \cdot F_i.$$

Let T be the round when p_{k^*} can not be selected in $T+1$. Then, $T = N$ or $B_n(T+1) < p_{k^*}$. The former denotes there are no workers any more. The latter denotes there is not enough budget, where we have

$$\begin{aligned} B - p_{k^*} &< \sum_{i=1}^{K_r} \mathbb{E}[N_i(T)] \cdot F_i p_i \\ &\leq \sum_{i>k^*} \mathbb{E}[N_i(T)] \cdot [F_i p_i - F_{k^*}p_{k^*}] + \sum_{i=1}^{K_r} \mathbb{E}[N_i(T)] \cdot F_{k^*}p_{k^*}. \end{aligned}$$

Thus,

$$\mathbb{E}[T] = \sum_{i=1}^{K_r} \mathbb{E}[N_i(T)] > B/(F_{k^*}p_{k^*}) - 1/F_{k^*} - \sum_{i>k^*} \mathbb{E}[N_i(T)] \cdot \delta_i$$

where $\delta_i = (F_i p_i - F_{k^*}p_{k^*})/(F_{k^*}p_{k^*})$. *Furthermore,*

$$\begin{aligned} R &= U(p_{k^*}) - \sum_{i=1}^{K_r} \mathbb{E}[N_i] \cdot F_i \\ &\le \min\{N \cdot F_{k^*}, B/p_{k^*}\} + \sum_{i<k^*} \mathbb{E}[N_i(T)] \cdot [F_{k^*} - F_i] - \mathbb{E}[T] \cdot F_{k^*}. \end{aligned}$$

When $T = N$,

$$R \le \sum_{i<k^*} \mathbb{E}[N_i(T)] \cdot [F_{k^*} - F_i].$$

When $B_n(T+1) < p_{k^*}$,

$$R < 1 + \sum_{i<k^*} \mathbb{E}[N_i(T)] \cdot [F_{k^*} - F_i] + \sum_{i>k^*} \mathbb{E}[N_i(T)] \cdot \delta_i F_{k^*}.$$

Since $\delta_i \le \delta_{K_r}$, *we can get*

$$R < 1 + \sum_{i<k^*} \mathbb{E}[N_i(T)] \cdot [F_{k^*} - F_i] + \sum_{i>k^*} \mathbb{E}[N_i(T)] \cdot \frac{K_r \delta_p - F_{k^*}p_{k^*}}{p_{k^*}}.$$

Recall the definitions of A_1, A_2 *and* A_3*:*

$$A_1 = \{n|\hat{k}(n) < k^*\}\ ,\ A_2 = \{n|\hat{k}(n) > k^*\}\ ,\ A_3 = \{n|\hat{k}(n) = k^*, k(n) = k^* - 1\}.$$

Thus,

$$\sum_{i<k^*} \mathbb{E}[N_i(T)] \le \mathbb{E}|A_1| + \mathbb{E}|A_3| \quad , \quad \sum_{i>k^*} \mathbb{E}[N_i(T)] = \mathbb{E}|A_2|.$$

Correspondingly, we can get

$$\sum_{i<k^*} \mathbb{E}[N_i(T)] \cdot [F_{k^*} - F_i] \le \mathbb{E}|A_1| + (F_{k^*} - F_{k^*-1})\, \mathbb{E}|A_3|$$

because $F_{k^*} - F_i \le 1$. *Meanwhile, we can have*

$$\sum_{i>k^*} \mathbb{E}[N_i(T)] \cdot \frac{K_r \delta_p - F_{k^*}p_{k^*}}{p_{k^*}} = [K_r \delta_p - F_{k^*}p_{k^*}] \cdot \mathbb{E}|A_2|/p_{k^*}.$$

Thereby, we can conclude Theorem 3.8.

Theorems 3.4 and 3.8 show that the regrets of both the algorithm and the posted-price mechanism are linearly proportional to $\mathbb{E}|A_1|$, $\mathbb{E}|A_2|$ and $\mathbb{E}|A_3|$. This linkage explains the rationale behind our indirect approach — i.e. we can first develop the optimal MAB algorithm to minimize $\mathbb{E}|A_1|$, $\mathbb{E}|A_2|$ and $\mathbb{E}|A_3|$, and then design our mechanism based on the optimal MAB algorithm, which will also minimize the regret of our mechanism. In addition, due to the probability decay that will be explained in Section 5.3, K_r may be far smaller than $\lfloor B/\delta_p \rfloor$, which will further reduce the regret and is favorable in practical use.

3.3.3 The Regret under Infinite Price Range

In our mechanism, the price range is set as the budget B. However, in real markets, B is usually very large, leading to a large number of possible prices. To avoid the low efficiency of exploring a large price space, existing mechanisms all require inputting a prior price range. By contrast, the regret of our mechanism is not affected by the infinitely increasing price range. To demonstrate, we analyze the probability distribution of the largest price explored in our mechanism:

Theorem 3.9 (Probability Decay) *There exists a probability $\mathbb{P}_d < 1$ which ensures $\mathbb{P}(K_r = k^* + n) \leq \mathbb{P}_d^{n-1}$.*

Proof: *If Algorithm 1 outputs K_r, p_{K_r} or p_{K_r+1} must be identified as the minimum POP. Thus, $\mu_k < C_k$ must hold for $k^* < \forall k < K_r$. In this case, we can have*

$$\mathbb{P}(K_r = k^* + n) \leq \prod_{k=k^*+1}^{k^*+n-1} \mathbb{P}(\mu_k \leq C_k) \tag{3.10}$$

Meanwhile, since the initial value of μ_i is set as 1 in Algorithm 2, to satisfy $\mu_k < C_k$, p_k must be tried for at least one time. Considering the fact that $F_k > C_k$ and the Chernoff-Hoeffding bound [4], we can get

$$\mathbb{P}(\mu_k \leq C_k) \leq e^{-2N_k(F_k - C_k)^2} \leq e^{-2\Delta^2} \tag{3.11}$$

where $\Delta = F_{k^*+1} - C_{k^*+1}$. Combining Equations 3.10 and 3.11, we can conclude Theorem 3.9 by setting $\mathbb{P}_d$ as $\exp(-2\Delta^2)$.

Then, we conduct asymptotic analysis of the regret as:

Theorem 3.10 *When $N \to \infty$ and $B/N = const$,*

$$R(N,B) \leq \begin{cases} O(\log\log N) & p_{k^*} \text{ is POP-1} \\ O\left(\frac{F_{k^*}-F_{k^*-1}}{KL[F_{k^*-1},C_{k^*}]}\log N\right) & p_{k^*} \text{ is POP-2} \end{cases}$$

Proof: *Considering the probability decay, we can bound the expected value of the right-hand side of Equation* 3.9 *using*

$$\sum\nolimits_{n=1}^{\infty} K_r \delta_p \mathbb{P}(K_r = k^* + n) \leq \sum\nolimits_{n=1}^{\infty} (k^* + n)\delta_p \mathbb{P}_d^n \tag{3.12}$$

The other items in Equation 3.9 *are not affected by K_r. Furthermore, we can compute the above infinite series as*

$$\sum\nolimits_{i=1}^{\infty} (k^* + n)\mathbb{P}_d^n = (k^* + 1)Z(\mathbb{P}_d) + Z^2(\mathbb{P}_d) < +\infty \tag{3.13}$$

where $Z(\mathbb{P}_d) = \mathbb{P}_d/(1 - \mathbb{P}_d)$. Thus, the effects of the infinite price range ($B \to \infty$) is bounded by a finite constant. Hence, using Theorems 3.5∼3.7, we can conclude Theorem 3.10.

Theorem 3.10 shows that our mechanism not only outperforms the state-of-the-art mechanism, BP-UCB [79], but also classic MAB algorithms, such as UCB-1 [4] and OSUB [13]. In addition, even when the number of possible prices is infinite, the regret upper bound we get in Theorem 3.10 still matches the Lai-Robbins regret lower bound in Theorem 3.2. In other words, the optimality of our mechanism is not affected by the number of possible prices, which is attractive for practical usage but unreachable by existing posted-price mechanisms and MAB algorithms.

3.4 Experimental Evaluation

In this section, we empirically compare our mechanism with state-of-the-art mechanisms including BP-UCB [79], PD-BwK [6] and UCB-BwK [1]. The idealized case with the optimal price p_{k^*} known in advance and always selected is also employed for comparison. The testbeds are built based on the three worker models mentioned in

Table 3.2: Experimental settings for Posted-Price Mechanisms

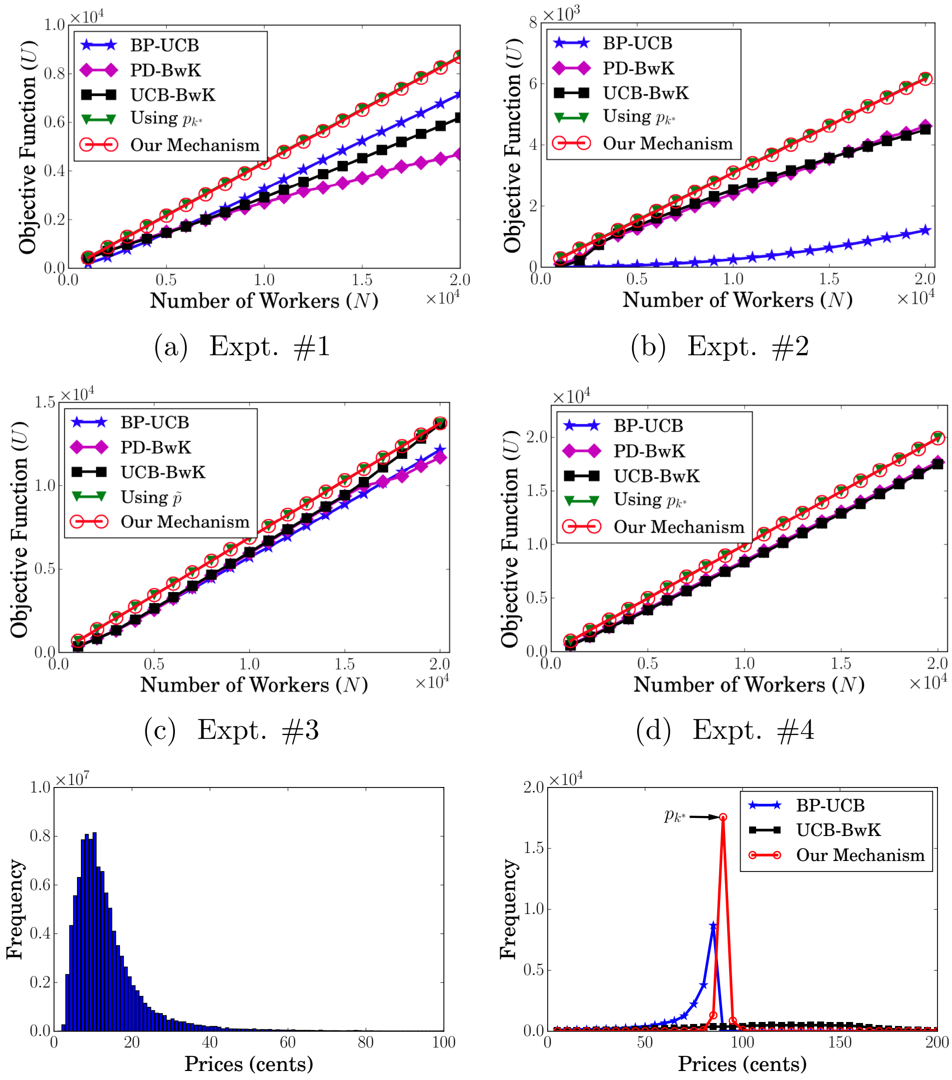
Expt.	Worker Model	Price Range[1]	B/N
#1	Private Cost Model: $c_i \sim U[5, 200]^2$	$[5, 200]$	40
#2	Discrete Choice Model: $\alpha_i = 1/15$, $\beta_i = 0.39$, $M_i = 2000$	$[1, 200]$	30
#3	Reference Payment Model: $\alpha_i \in \{0, 1, 3\}$, $\beta_i \in \{0, 1, 3\}$, $r_i \in \{20, 60, 120\}$, Evenly Distributed	$[1, 200]$	70
#4	The data (Fig. 3.1e) collected using MTurk-Tracker [17]	[1, 100]	10

[1] The price range is set for the benchmark mechanisms.
[2] Here, U denotes the uniform distribution.

Section 3.1 and the real-world price data collected from MTurk. In our experiments, workers come sequentially. The mechanism offers a price for each worker. The worker decides to accept or reject the price according to the worker model. After observing worker's decision, the mechanism updates the price offered to the next worker. The settings of our experiments are shown in Table 3.2. The price unit in all experiments is the cent. The objective function is estimated with the mean of 100 runs. Besides, we keep the ratio between the budget B and the number of workers N fixed so as to ensure the optimal price p_{k^*} to be unchanged and thus a fair comparison. Meanwhile, for different set of experiments, we keep testing different values of the ratio to show that our mechanism can work very well in different parameter settings.

The results are shown in Fig. 1(a-d), respectively. We can conclude that our mechanism remarkably outperforms state-of-the-art mechanisms.[3] It even achieves the same performance as the idealized case where p_{k^*} is known and always selected. To explain the reason for the optimal performance of our mechanism, we compare the selected price distributions in Fig. 3.1f. We can observe an overstaying of BP-UCB at the current optimal price, which is caused by an over-exploitation on the current estimate of $F(p)$. UCB-BwK, on the other hand, shows an over-exploration on the

[3] The t-test also supports our conclusion. For example, in Figure 3.1c, if $N = 1.0 \times 10^4$, the p-values of the t tests between our mechanism and other algorithms except for using p_{k^*} are all below 1×10^{-3}.



(a) Expt. #1
(b) Expt. #2
(c) Expt. #3
(d) Expt. #4
(e) Prices in MTurk(2011-2016)
(f) Selected Prices in Expt. #1

Figure 3.1: Experiment results for posted-price mechanisms on different worker models

possible better prices. By contrast, our mechanism performs much better in balancing exploitation and exploration. It can not only move quickly to p_{k^*}, but also accurately stop exploration around p_{k^*}. Note that PD-BwK behaves similarly as UCB-BwK and thus is not included for comparison in Fig. 3.1f.

To further verify the practical usability of our mechanism, we conduct robustness analysis by considering two abnormal cases where the assumptions used for design are violated. Here, we use the private cost model as the testbed, and the settings are the same as Expt. #1. In Fig. 3.2a, we compare the performance of different mechanisms when the number of workers N is not accurately given. The real number of workers

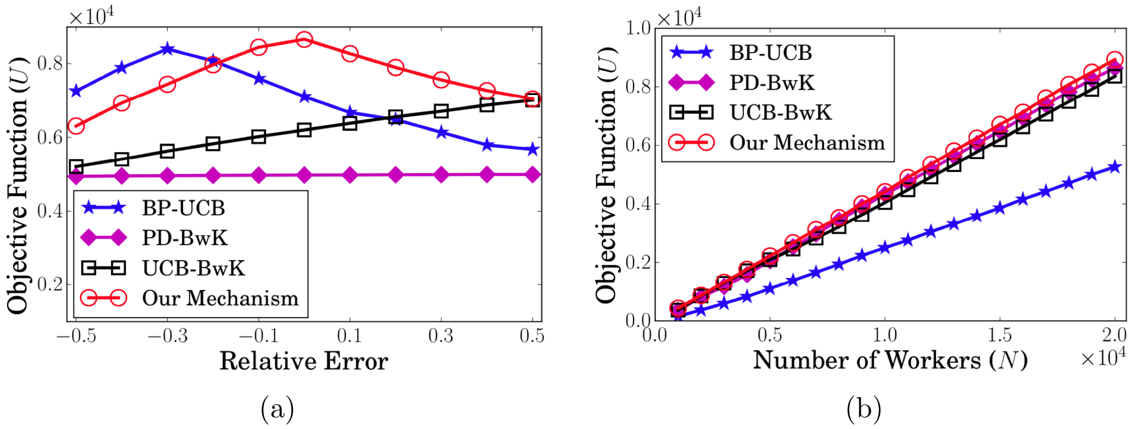


Figure 3.2: Robustness tests of posted-price mechanisms against: (a) the inaccurate worker number; (b) the non-stochastic arrival of workers

N is 20000, and the relative error is calculated as $(\bar{N} - N)/N$, where $\bar{N}$ is the value offered to the mechanism. The comparison shows that our mechanism has distinct advantages over state-of-the-art mechanisms in this abnormal case. Fig. 3.2b presents the comparison of different mechanisms in a non-stochastic setting where two groups with the private cost uniformly distributed in $[5, 100)$ and $[100, 200]$ respectively arrive one after another. This setting can explain the phenomenon that working at night may cost workers more than at day. The results show that the performance advantage of our mechanism is kept in this non-stochastic setting.

3.5 Summary

In this chapter, we propose an optimal posted-price mechanism for microtask crowdsourcing. Compared with existing mechanisms, our mechanism not only has better performance but also requires fewer inputs. To demonstrate the advantages, we firstly prove the optimality of our algorithm that its regret matches the Lai-Robbins regret lower bound. This lower bound applies to any possible algorithms and denotes the best performance that can be achieved. Then, we prove that the regret of our mechanism is not affected by the infinite price range. Besides, the empirical results on various worker models and the real price data collected from MTurk also verify the advantages of our mechanism.

Chapter 4

Active Task Assignment Mechanism

This chapter focuses on the task assignment process of microtask crowdsourcing. We propose a novel active crowd labeling framework to improve task assignment [36]. It utilizes the variational inference algorithm to learn the estimates of true labels and worker models in an online fashion. Different from the previous inference studies that assume one specific worker model, we provide a unified formulation of different worker models so that our inference algorithm can work with any of the worker models. In our mechanism, we also propose a novel prediction-based task assignment strategy. To improve computation efficiency, we derive the first-order approximation equations of our variational inference algorithm that can be solved by Newton's method efficiently. To suppress the mutual reinforcement between the uncertainties of true labels and worker models, we develop a heuristic rule to modulate the scope of task assignment based on the uncertainty measurement of the current inference results. Besides, we replace all the expected values with the upper confidence bounds to achieve an optimistic prediction. We conduct extensive experiments based on four popular worker models and four MTurk datasets. The empirical results show that our framework not only requires the least number of labels for high label accuracy but also achieves the highest computation efficiency.

The following sections of this chapter are arranged as follows. In Section 4.1, we formally describe the task assignment process in microtask crowdsourcing and provide a general description of active task assignment mechanism. Then, in Section 4.2, we

Table 4.1: Key notations used in Chapter 4

M	Number of tasks	N	Number of workers
$L_i^t(j)$	Worker i's label for worker j at step t		
$L^t(j)$	Worker i's label for task j at step t		
$L^t(j)$	The real true label of task j		
S	Label matrix, $[s_i(j)]$	$s_i(j)$	Worker i's label for task j
$\mathcal{L}^t(j)$	Inferred true label for task j	$\mathcal{L}_j^t$	Simplified notation of $\mathcal{L}^t(j)$
C_i	Confusion matrix of worker i	c_{ikg}	Element of C_i
E	Set of constraints on C_i, for example, $\{c_{i11} = c_{i22}\}$	$\mathcal{B}$	Basis of workers' labeling behaviour space
B_{wl}	Element of $\mathcal{B}$ (basis matrix)	θ_{iwl}	Probability on basis B_{wl}
δ_{ijg}	Indicator function, $1(s_{ij}^t = g)$	$A(t)$	Label accuracy at step t
q_{jk}	Parameter of the categorical distribution, $\mathbb{P}(\mathcal{L}_j = k) = q_{jk}$		
$\langle i_t, j_t \rangle$	Assign task j_t to worker i_t at step t		
$I(t, x, z, \lambda)$	Accuracy increment when worker x label task z as λ at step $t+1$		
$\tilde{I}$	Estimation of $I(t, x, z, \lambda)$	$Q(t)$	Scope of task assignment
$\tilde{G}(t, x, z)$	Conditional value-at-risk of $\mathbb{E}\tilde{I}(t, x, z, \lambda)$, where $\lambda \sim [q_{z\lambda}^t]$		

analyze the unique accuracy and efficiency problems existing in the task assignment process of microtask crowdsourcing and propose a novel task assignment strategy to solve these problems. In Section 4.3, we conduct extensive experiments on four popular worker models and four MTurk datasets to validate the advantages of our mechanism. Finally, Section 4.4 summarizes this section. In addition, we list the key notations used in this chapter as Table 4.1. Note that, in this chapter, we use the subscript t to denote the case that the variable is computed at step t.

4.1 Task Assignment in Microtask Crowdsourcing

Suppose there are M tasks and N workers in a push market of microtask crowdsourcing. Tasks may belong to different classes numbered from 1 to K. $\mathcal{L}(j) \in \{1, \ldots, K\}$ denotes the true label of task j, and vector $\mathcal{L} = [\mathcal{L}(j)]_M$ represents the true labels of all tasks, and we assume all tasks are independent. The labels collected up to step t can be written as a matrix $S^t = [s_i(j)]_{N \times M}$, where $s_i(j) \in \{0, 1, \ldots, K\}$ and 0 means task j has not been labeled by worker i. At each step t, our mechanism decides the assignment $\langle i^t, j^t \rangle$ according to a certain task assignment strategy. After obtaining labels, it firstly updates the label matrix S^t and then the true label estimates $\mathcal{L}^t(j)$ and worker models based on the label matrix S^t. So, the accuracy $A(t)$ of our mechanism is calculated as

$$A(t) = \frac{1}{M}\sum\nolimits_{i=1}^{M} 1[\mathcal{L}^t(j) = \mathcal{L}(j)]. \tag{4.1}$$

In this chapter, we adopt a common assumption for the task assignment process of microtask crowdsourcing that acquiring a label incurs a fixed unit cost [47]. Thus, the objective of our active crowd labeling mechanism is to maximize the growth rate of accuracy $A(t)$ as the number of labels t.

To achieve this objective, our mechanism employs the variational inference to guide the assignment of tasks.[1] However, when applying this classic technique, it requires to assume a specific worker model. If the worker model is changed, the variational inference algorithm will need to be redeveloped. Thus, in this section, we propose a unified formulation of different worker models, based on which we derive an online variational inference algorithm that can take any worker model as the input. In this way, our mechanism can flexibly incorporate different worker models.

4.1.1 Unified Formulation of Worker Models

Previous studies [102, 99] usually used the confusion matrix $C_i = [c_{ikg}]_{K\times K}$ to depict workers' labeling behaviors. Here, c_{ikg} denotes the probability that worker i labels a task in class k as class g. Since the number of free parameters in C_i increases sharply as K increases, researchers have also introduced some simpler worker models. For example, the widely-adopted one-coin worker model assumes that worker i has a fixed probability to provide true labels for all tasks [99]. Welinder and Perona [92] assume that, given true label k of a task, worker i has the same probability to choose a wrong label $g \neq k$. In fact, these simplified models can be regarded as adding some

[1]We choose the variational inference technique because it can provide very stable performance in online settings where labels are incrementally added, while other methods (e.g. the Dawid-Skene [16] and Minimax [101] estimators) cannot. The reason behind is that the variational inference contains Laplacian smoothing [48] in computation. This advantage has also been confirmed by our empirical investigation.

Algorithm 3: Basis of Workers' Labeling Behavior Space

Input: the constraint set $E = \{c_{ikg} = c_{ist}\}$

Output: the basis of worker' labeling behavior space $\mathcal{B}$

$w \leftarrow 0, l \leftarrow 0, \mathcal{B} \leftarrow \varnothing, T \leftarrow \varnothing$

for $k = 1$ **to** K **do**

  for $g = 1$ **to** K **do**

    if $g = 1$ ***and*** $c_{ikg} = c_{ist}$ *does not hold for any* $c_{ist} \in T$ **then**

      $w \leftarrow w + 1, l \leftarrow 1$, Append B_{wl} with $B_{wl}(k, g) = 1$ to $\mathcal{B}$

    else if $c_{ikg} = c_{ist}$ *does not hold for any* $c_{ist} \in T$ **then**

      $l \leftarrow l + 1$, Append B_{wl} with $B_{wl}(k, g) = 1$ to $\mathcal{B}$

    else

      If $c_{ikg} = c_{ist} \in T$ and there exist m and n satisfying $B_{mn}(s, t) = 1$, set $B_{mn}(k, g) = 1$

    Append c_{ikg} to the set of traversed elements T

extra equality constraints to the confusion matrix. For example, the one-coin worker model is equivalent to requiring $c_{iks} = c_{igt}$ and $c_{ikk} = c_{igg}$ for all $k \neq s$ and $g \neq t$. We can formulate these constraints as $c_{ikg} = c_{ist}$, and they form a constraint set E. Since variational inference cannot incorporate these equality constraints, we have to redevelop the mechanism when changing to a different worker model. Thus, we build a unified representation of worker models so that our inference algorithm can work with different worker models without modification.

Mathematically, these equality constraints limit the values of the confusion matrix to a subspace of the matrix space $R^{K \times K}$. We call this subspace as workers' labeling behavior space, which provides an equivalent representation of the worker model. To uniformly depict this subspace, we formally define its basis as:

Definition 4.1 *The basis of workers' labeling behavior space $\mathcal{B}$ is a set of $K \times K$ matrices B_{wl}, each element of which is 0 or 1. Besides, $B_{wl}(k, g) = B_{wl}(s, t) = 1$ is equivalent to requiring $c_{ikg} = c_{ist}$.*[2] *Note that, to uniformly represent different worker models, we arrange all the basis matrices as a two dimensional table. Thus, the subscripts of $B_{wl}(k, g)$, w and l, denotes the row and column numbers in the*

[2] All workers share the same behavior space. Thus, j does not denote a specific value.

table. Meanwhile, all basis matrices are two dimensional. Thus, k and g in $B_{wl}(k,g)$ represent a specific position in the basis matrix.

In Definition 4.1, $w \in [W]$ and $l \in [L(w)]$, where W and $L(w)$ denote the degree of freedom of rows in the confusion matrix and the elements in row w, respectively. We do not need to know the values of W and $L(w)$ in advance because they will be the by-products of building the basis $\mathcal{B}$. In Algorithm 3, we denote all the constraints added to the confusion matrix (e.g. $c_{i11} = c_{i22}$ and $c_{i12} = c_{i21}$ in the aforementioned one-coin model) by the constraint set E, and present the pseudo code to build the basis $\mathcal{B}$ from E via traversing all elements in the confusion matrix by row.[3] For each element c_{ikg}, if it is not required to be the same as any previous element, we will add a new basis matrix B_{wl} (line 7); otherwise, we just represent the equality constraint $c_{ikg} = c_{ist}$ by adding a new element 1 to the basis matrix corresponding to c_{ist} (line 9). If the unconstrained c_{ikg} is the first element of a row, we add a new row to the basis matrix set (line 5). In this way, we can map a row of the confusion matrix to a row of the basis matrices (Theorem 4.2), which can facilitate the computation of the normalization condition $\sum_g c_{ikg} = 1$. Based on Algorithm 3, we can derive the following three properties of $\mathcal{B}$:

Theorem 4.1 *Given a pair of $k \in [K]$ and $g \in [K]$, there must exist one and only one basis matrix B_{wl} satisfying $B_{wl}(k,g) = 1$.*

Proof: *For each pair of k and g, Algorithm 3 either adds a new basis matrix or sets $B_{mn}(k,g) = 1$, which naturally ensures the existence of $B_{wl}(k,g) = 1$. On the other hand, if there are two different basis matrices that satisfy $B_{m_1n_1}(k,g) = B_{m_2n_2}(k,g) = 1$, the element 1 can only be set in line 9 of Algorithm 3. Thus, there exist (s_1,t_1) and (s_2,t_2) satisfying $B_{m_1n_1}(s_1,t_1) = 1$ and $B_{m_2n_2}(s_2,t_2) = 1$. Without loss of generality, we assume $c_{js_1t_1}$ locates before $c_{js_2t_2}$. To add the basis matrix $B_{m_2n_2}$, $c_{js_1t_1} = c_{js_2t_2}$ does not hold. However, both $c_{is_1t_1}$ and $c_{is_2t_2}$ equal c_{ikg}. These two aspects contradict, which proves the uniqueness and thus concludes Theorem 4.1.*

[3] Algorithm 3 will create a basis for a given family of worker models. Since our inference and task assignment algorithms are designed without assuming a specific worker model, any models can be used in our mechanism after transferring to the basis formation.

Theorem 4.2 *For the k-th row of C_j, if there exists $k \in [K]$ satisfying $B_{wl}(k,g) = 1$, $B_{mn}(k,q) = 0$ for all $m \neq w$ and $q \in [K]$.*

Proof: *Suppose the first $k-1$ rows of C_i satisfy Theorem 4.2. Assume that, given $B_{wl}(k,g) = 1$, there exist $q \in [K]$, $m \neq w$, and $n \in [L(w)]$ satisfying $B_{mn}(k,q) = 1$. According to the uniqueness proved in Theorem 4.1, we can know that $B_{mn}(k,g) = 0$ and $B_{wl}(k,q) = 0$. Without loss of generality, we assume $m < w$. In Algorithm 3, $B_{mn}(k,q) = 1$ can only be set through the third case, namely line 9. In other words, there exists $s < k$ and $t \in [K]$ satisfying $c_{ikq} = c_{ist}$. However, due to the uniqueness, $B_{md}(k,g) = 0$ holds for all $d \in [K]$, which means $c_{ikg} \neq c_{isx}$ for all $x \in [K]$ because our assumption at the beginning of this proof requires elements in row s of C_i share the same w. Meanwhile, the normalization condition of probability requires $\sum_{x\in[K]} c_{ikx} = \sum_{x\in[K]} c_{isx} = 1$. Here, x is introduced to let c_{ikx} and c_{isx} denote any element in rows k and s, respectively. Since $c_{ikg} \neq c_{isx}$ for all $x \in [K]$ while $c_{ikq} = c_{ist}$, we can obtain a non-cancellable constraint that $\sum_{x\neq q} c_{ikx} = \sum_{x\neq t} c_{isx}$, which violates our requirement that all equality constraints are in set E. Thereby, elements in row k must share the same w. Besides, in Algorithm 3, all elements in the first row of C_i naturally satisfy Theorem 4.2 because m in line 9 can only be 1. In this case, our recursive proof holds for all rows of C_i, which concludes Theorem 4.2.*

Theorem 4.3 $\sum_{g\in[K]} B_{wl}(k,g) = 0$ *or* $||B_{wl}||$*, where* $||\cdot||$ *denotes the* ∞*-norm, namely* $||B_{wl}|| = \max_{k\in[K]} \sum_{g\in[K]} B_{wl}(k,g)$.

Proof: *Let $\sum_g B_{wl}(k,g) = a_{kl}$ and $\sum_g B_{wl}(s,g) = a_{sl}$. Assume there exits $q \in [K]$ satisfying $a_{kq} \neq a_{sq}$, $a_{kq} > 0$ and $a_{sq} > 0$. According to Theorems 4.1 and 4.2, for any element in rows k and s of C_i, for example, c_{ikg}, there exists $x \in [K]$ staisfying $B_{wx}(k,g) = 1$. Considering the normalization condition, we can obtain an equation that $\sum_l a_{kl} c_{ikg_{kl}} = \sum_l a_{sl} c_{isg_{sl}} = 1$, where the subscript g_{kl} satisfies $B_{wl}(k, g_{kl}) = 1$. The existence of $a_{kq} \neq a_{sq}$ will cause this equation to be a non-cancellable constraint which violates our requirement that all constraints are in set E. Thus, the assumption does not hold and $a_{kq} = a_{sq} > 0$. In this case, the ∞-norm satisfies $a_{kq} = ||B_{wl}||$, which concludes Theorem 4.3.*

We say that a labeling behavior (for example, labeling a task in class k as class g) is captured by B_{wl} when $B_{wl}(k,g) = 1$. Then, we use $\theta_{iwl} = \sum_g c_{ikg} B_{wl}(k,g)$ to denote the probability that the labeling behaviors captured by B_{wl} happen when the true label is k. Suppose there exists another $s \neq k$ and $t \in [K]$ satisfying $B_{wl}(s,t) = 1$. In other words, there is another labeling behavior captured by B_{wl}. Due to Theorem 4.3, we can get $\sum_g c_{ikg} B_{wl}(k,g) = \sum_g c_{isg} B_{wl}(s,g)$. Thus, we can overlook the labeling behaviors in other rows and use θ_{iwl} without considering which specific k is conditioned on. Taking the uniqueness of element 1 (Theorem 4.1) into consideration, we can further calculate c_{ikg} as

$$c_{ikg} = \sum\nolimits_{w \in [W]} \sum\nolimits_{l \in [L(w)]} \frac{\theta_{iwl}}{||B_{wl}||} B_{wl}(k,g). \tag{4.2}$$

Meanwhile, Theorem 4.2 ensures the equivalence between the normalization conditions $\sum_g c_{ikg} = 1$ and $\sum_l \theta_{iwl} = 1$. In this way, we convert different worker models into a set of standard $K \times K$ basis matrices and uniformly denote the variables of different worker models by the conditional probability vector $\boldsymbol{\theta}_i = [\theta_{iwl}]$. These basis matrices can be further included in our inference algorithm and task assignment strategy, which enables our mechanism to incorporate different worker models without modification.

4.1.2 Online Variational Inference

To apply the variational inference technique, we first need to analyze the joint probability distribution of the observed label matrix S^t, the hidden true labels $\mathcal{L}$, and the conditional probability vector $\Theta = [\boldsymbol{\theta}_i]$. From [102], we can know that the joint probability distribution of $\mathcal{L}$, Θ and the confusion matrix $\boldsymbol{C} = [C_i]$ satisfying

$$\mathbb{P}(\mathcal{L}, S^t, \boldsymbol{C}) = \prod_{i \in [N], j \in [M], g \in [K]} \left(c_{i\mathcal{L}_j g}\right)^{\delta^t_{ijg}} \tag{4.3}$$

where $\delta^t_{ijg} = \mathbb{1}(s^t_{ij} = g)$ is 1 only when worker i's label for task j is g and 0 otherwise. For the simplicity of notations, we use $\mathcal{L}_j$ to denote $\mathcal{L}(j)$. Substituting Equation 4.2 into the Equation 4.3, we can compute the joint probability distribution of $\mathcal{L}$, Θ and S^t as

$$\mathbb{P}(\mathcal{L}, \Theta, S^t) = \prod_{\substack{i \in [N], j \in [M], g \in [K] \\ w \in [W], l \in [L(w)]}} \left(\frac{\theta_{iwl}}{||B_{wl}||}\right)^{\delta^t_{ijg} B_{wl}(\mathcal{L}_j, g)} \tag{4.4}$$

Then, we need to analyze the probability distribution of hidden variables, namely the true labels $\mathcal{L}$ and the conditional probability vector Θ. According to the mean field theory [86], we can factorize the distribution of $\mathcal{L}$ and Θ as

$$\mathbb{P}(\mathcal{L}, \Theta) = \prod_{j=1}^{M} \mathbb{P}(\mathcal{L}_j) \cdot \prod_{i=1}^{N} \prod_{w=1}^{W} \mathbb{P}(\boldsymbol{\theta}_{iw}) \tag{4.5}$$

where $\boldsymbol{\theta}_{iw} = [\theta_{iw1}, \ldots, \theta_{iwL}]$. The assumption behind the mean field theory is that all tasks and workers are independent. Furthermore, to ensure the normalization condition of conditional probabilities $\sum_l \theta_{iwl} = 1$, we approximate the probability distribution $\mathbb{P}(\boldsymbol{\theta}_{iw})$ with the Dirichlet distribution $\mathrm{Dir}(\boldsymbol{\alpha}_{iw})$, where the parameter vector $\boldsymbol{\alpha}_{iw} = [\alpha_{iw1}, \ldots, \alpha_{iwL}]$. Meanwhile, for the simplicity of notations, we use q_{jk} to denote $\mathbb{P}(\mathcal{L}_j = k)$. According to the basic theory of variational inference [86], the optimal estimates of variational variables q_{jk} and α_{iwl} should minimize the following KL divergence

$$\begin{aligned} KL[\mathbb{P}(\mathcal{L}, \Theta) || \mathbb{P}(\mathcal{L}, \Theta, S^t)] = & -\mathbb{E}_{\mathbb{P}(\mathcal{L},\Theta)}[\log \mathbb{P}(\mathcal{L}, \Theta, S^t)] \\ & + \sum_j \mathbb{E}_{\mathbb{P}(\mathcal{L}_j)}[\log \mathbb{P}(\mathcal{L}_j)] + \sum_i \sum_w \mathbb{E}_{\mathbb{P}(\boldsymbol{\theta}_{iw})}[\log \mathbb{P}(\boldsymbol{\theta}_{iw})] \end{aligned} \tag{4.6}$$

Solving this minimization problem, we can have

$$\alpha_{iwl}^t = \sum_{j=1}^{M} \sum_{k=1}^{K} \sum_{g=1}^{K} \delta_{ijg}^t B_{wl}(k, g) \cdot q_{jk}^t + \alpha_{wl}^0 \tag{4.7}$$

$$\begin{aligned} \log q_{jk}^t = & \sum_{i=1}^{N} \sum_{g=1}^{K} \sum_{w=1}^{W} \sum_{l=1}^{L} \delta_{ijg}^t \cdot B_{wl}(k, g) \cdot \Big[\psi(\alpha_{iwl}^t) \\ & - \psi(\sum_{l=1}^{L} \alpha_{iwl}^t) - \log(||B_{wl}||) \Big] + \varepsilon_j \end{aligned} \tag{4.8}$$

where $\psi(\cdot)$ represents the digamma function and ε_j denotes the normalization constant used to keep $\sum_k q_{jk} = 1$. α_{wl}^0 denotes the value of α_{jwl} at step 0, namely the prior knowledge about workers. In our mechanism, we assume all workers to share the same priors. According to the previous studies [12], we should be optimistic—i.e. believing workers have higher probability to be correct. Here, we denote the priors corresponding to the correct- and wrong-label-reporting components by $\alpha_{w^+l^+}^0$ as $\alpha_{w^-l^-}^0$, respectively. Mathematically, there exists at least one k that ensures $B_{w^+l^+}(k, k) = 1$. To be optimistic, we can set $\alpha_{w^-l^-}^0 = 1$ while keeping $\alpha_{w^+l^+}^0 > 1$. On the other hand, $\alpha_{w^+l^+}^0 \ll N$ is needed so that the dominance of the first item in the right-hand side

Algorithm 4: Active Task Assignment Mechanism

Input: the required number of labels T
the basis of workers' labeling behavior space $\mathcal{B}$

Output: the inferred true labels $\boldsymbol{y}^T$

```
1 for t = 0 to T − 1 do
2     ⟨i_t, j_t⟩ ← UMOA(q^t_{ik}, α^t_{jwl}, S^t, B)
3     S^{t+1} ← Get the label and update the label matrix S^t
4     ⟨q^{t+1}_{ik}, α^{t+1}_{jwl}⟩ ← Iterate Eq. 4.7 and 4.8 until convergence
5 y^T_i ← arg max_k q^T_{ik}   (Maximum a Posterior Probability Rule)
```

of Equation 4.7 can be ensured. Our empirical investigation shows that the vaule changes of $\alpha^0_{w^+l^+}$ in the range discussed above almost have no effects on the inference results. Thus, we keep $\alpha^0_{w^+l^+} = 4$ in this Chapter.

4.1.3 Pseudo-Code Summary

To summarize, we present the pseudo code of our active task assignment mechanism in Algorithm 4. At each step, it firstly decides the task assignment $\langle i_t, j_t \rangle$ based on the current estimates of variational variables. After collecting a new label, our mechanism updates the label matrix and solves the variational variables by iterating Equations 6 and 7 until convergence. Note that, to improve the computation efficiency, we use the values of variational variables at the last step as the starting point of iteration. When the required T labels are collected, we decide the true labels by using the classic maximum a posteriori probability rule (line 5).

The remaining question of our mechanism is how to use the inferred variational variables to guide the task assignment. The existing task assignment strategies [59, 12] encounter the efficiency bottleneck and uncertainty reinforcement when combined with variational inference in microtask crowdsourcing. This unique problem motivates us to propose a novel task assignment strategy, uncertainty modulated optimistic assignment (UMOA), which will be detailed in the next section.

4.2 Task Assignment Strategy

The objective of our active task assignment mechanism is to maximize the growth rate of accuracy $A(t)$ as the number of labels t. However, the accuracy $A(t)$ defined in Equation 4.1 cannot be used to guide our task assignment because the true labels $\mathcal{L}$ are unknown. Instead, we calculate the expectation of label accuracy based on the estimated posterior distribution of true labels as

$$\mathbb{E}A(t) = \frac{1}{M}\sum_{j=1}^{M} \mathbb{E}_{\mathcal{L}_j \sim [q_{jk}]}\left[1(\mathcal{L}_j^t = \mathcal{L}_j)\right] = \frac{1}{M}\sum_{i=1}^{M} \max_{k\in[K]} q_{jk}^t. \tag{4.9}$$

Suppose worker x provides label λ for task z at step t.[4] Then, the expected accuracy increment or error reduction brought by this new label can be calculated as

$$I(t,x,z,\lambda) = \frac{1}{M}\sum_{j=1}^{M}\left(\max_k q_{jk}^{t+1} - \max_k q_{jk}^t\right).^5 \tag{4.10}$$

The classic expected error reduction strategy greedily selects the assignment $\langle x, z\rangle$ that maximizes $\mathbb{E}_\lambda I(t,x,z,\lambda)$, where the distributions of λ is computed based on the current variational estimates q_{jk}^t and α_{iwl}^t. Nevertheless, when applying this simple greedy strategy to microtask crowdsourcing, we observe the following two drawbacks.

- **Efficiency Bottleneck**: To find the optimal task assignment, we need to traverse all possible pairs of tasks and workers. In other words, we need to compute $O(MN)$ times of $\mathbb{E}_\lambda I(t,x,z,\lambda)$. For every λ, we need to solve Equations 4.7 and 4.8 to compute q_{jk}^{t+1} in Equation 4.10. Solving Equations 4.7 and 4.8 requires to repeatedly go through all tasks and workers, and the time complexity is around $O(MN)$. Thereby, computing all the predictions requires $O(M^2N^2)$ time costs, which significantly lowers the computation efficiency because there are usually hundreds of tasks and workers in a market.
- **Uncertainty Reinforcement**: Since we are uncertain about the true labels, we cannot accurately infer worker models. The uncertainty of worker models will

[4] Here, we introduce the new notation to avoid the nested subscript i_t and j_t in our derivation.

[5] The N tasks is included because the iteration used to solve Equations 4.7 and 4.8 makes the true label estimates for all tasks correlated, even though the true labels of tasks are independent.

in turn cause us to be more uncertain about the true labels. This reinforcement process amplifies the uncertainties of the inferred true labels and worker models. Thereby, if we purely rely on Equation 4.10 to guide the task assignment, we may make severe mistakes. For example, after obtaining the first label, the inference algorithm will trust the label because of the optimistic priors. This inferred true label will increase our confidence about the worker's ability to provide correct labels. Then, we will choose this "trusted" worker to label a new task because new tasks have the minimal $\max_k q_{jk}^t$, and this worker becomes more "trusted". Finally, we use this "trusted" worker to label all tasks at first and then greedily select workers who can provide the same label as this worker. This way of task assignment is obviously not reasonable.

Due to this unique problem of microtask crowdsourcing, in this section, we first overcome the efficiency bottleneck by developing an efficient algorithm to predict q_{jk}^{t+1}. Its time complexity is independent of M and N. Then, we suppress the uncertainty reinforcement by using the uncertainty measurement of current inference results to adjust the scope of task assignment. Furthermore, we follow the optimistic rule to replace the expectations with the upper confidence bounds in prediction. After these modifications, we formally write our task assignment strategy as Equation 4.22 below, in Section 4.2.2.

4.2.1 Efficient Prediction

In Equations 4.7 and 4.8, the newly obtained label $s_{xz} = \lambda$ only causes the change that $\delta_{xz\lambda}^t = 0 \rightarrow \delta_{xz\lambda}^{t+1} = 1$. When iteratively solving Equations 4.7 and 4.8, since we use the estimates at step t as the starting values of step $t+1$, this change firstly affects the variational variables of task z and worker x. Through the second round of iteration, the variational variables of the workers who have labeled task z and the tasks that have been labeled by worker x will then be updated. This process repeats, and finally the variational variables of all workers and tasks will be updated. To achieve efficient prediction, we focus on the direct effects of s_{xz} on task z and worker

x and neglect its high-order effects on other tasks and workers. In other words, when predicting the next-step values of variational variables, we assume $q_{jk}^{t+1} \approx q_{jk}^{t}$ for $j \neq z$ and $\alpha_{iwl}^{t+1} \approx \alpha_{iwl}^{t}$ for $i \neq x$. Despite this assumption, solving Equations 4.7 and 4.8 for prediction still needs to repeatedly go through all tasks and workers. To overcome this efficiency bottleneck, we construct a new set of equations to directly solve the differences of q_{zk} and α_{xwl} between step t and $t+1$. Firstly, we define the differences as

$$\phi_{zk}^{t} = q_{zk}^{t+1} - q_{zk}^{t} \quad , \quad \varphi_{xwl}^{t} = \alpha_{xwl}^{t+1} - \alpha_{xwl}^{t}. \tag{4.11}$$

Then, substituting Equation 4.7 into Equation 4.11, we can have

$$\varphi_{xwl}^{t} = \sum\nolimits_{k=1}^{K} B_{wl}(k,\lambda) \cdot (\phi_{zk}^{t} + q_{zk}^{t}). \tag{4.12}$$

According to Theorem 4.1, there will be at most one k satisfying $\xi_{k\lambda wl} = 1$, which ensures $0 \leq \varphi_{xwl}^{t} \leq 1$. On the other hand, in Equation 4.7, α_{jwl}^{t} will become far larger than 1 as the number of labels increases. Thus, for the relatively small φ_{xwl}^{t}, we introduce the following linear approximation equation

$$\psi(\alpha_{xwl}^{t+1}) - \psi(\alpha_{xwl}^{t}) \approx \zeta(\alpha_{xwl}^{t})\varphi_{xwl}^{t} \tag{4.13}$$

where $\zeta(\cdot)$ denotes the trigamma function [56]. Then, substituting Equations 4.8, 4.12 and 4.13 into Equation 4.11, we can have

$$\log\left(1 + \frac{\phi_{zk}^{t}}{q_{zk}^{t}}\right) - \sum\nolimits_{g=1}^{K} T_{kg} \cdot \phi_{zg}^{t} + \tilde{\varepsilon}_z = H(k) \tag{4.14}$$

where

$$\begin{aligned}
&T_{kg} = \sum\nolimits_{w=1}^{W}\sum\nolimits_{l=1}^{L} \tau_{xwl} \cdot B_{wl}(g,\lambda) \\
&\tau_{xwl} = B_{wl}(k,\lambda) \cdot \zeta(\alpha_{xwl}^{t}) - \sum\nolimits_{q=1}^{K} B_{wq}(k,\lambda) \cdot \zeta(\sum\nolimits_{l=1}^{L} \alpha_{xwl}^{t}) \\
&H(k) = \sum\nolimits_{w=1}^{W}\sum\nolimits_{l=1}^{L}\sum\nolimits_{g=1}^{K} \tau_{xwl} \cdot B_{wl}(g,\lambda) \cdot q_{zg}^{t} + \\
&\sum\nolimits_{w=1}^{W}\sum\nolimits_{l=1}^{L} B_{wl}(k,\lambda) \cdot \left[\psi(\alpha_{xwl}^{t}) - \psi(\sum\nolimits_{l=1}^{L}\alpha_{xwl}^{t}) - \log||B_{wl}||\right].
\end{aligned}$$

The relaxation variable $\tilde{\varepsilon}_z$ ensures the probability sum of all label values to always be 1, namely

$$\sum\nolimits_{k=1}^{K} q_{zk}^{t+1} - \sum\nolimits_{k=1}^{K} q_{zk}^{t} = \sum\nolimits_{k=1}^{K} \phi_{zk}^{t} = 0. \tag{4.15}$$

In this way, we convert the prediction of the variational variables as solving $K+1$ logarithmic equations.

The most efficient method to solve Equations 4.14 and 4.15 is Newton's method [8]. However, the overshoot of Newton's method may cause the essential condition $\phi^t_{zk} \geq -q^t_{zk}$ in Equation 4.14 to be violated. Thus, we eliminate the non-negative condition of the logarithmic function via defining a new variable as

$$\sigma^t_{zk} = \log\left(1 + \phi^t_{zk}/q^t_{zk}\right). \tag{4.16}$$

Then, we can rewrite Equations 4.14 and 4.15 as $\boldsymbol{F} = \mathbf{0}$, where the vector $\boldsymbol{F} = [F(k)]_{K+1}$. For $k = 1, \ldots, K$,

$$\begin{aligned} F(k) &= \sigma^t_{zk} - \sum\nolimits_{g=1}^{K} T_{kg} \cdot q^t_{zg} \cdot (e^{\sigma^t_{zg}} - 1) + \tilde{\varepsilon}_z - H(k) \\ F(K+1) &= \sum\nolimits_{k=1}^{K} q^t_{zk} \cdot (e^{\sigma^t_{zk}} - 1). \end{aligned} \tag{4.17}$$

Applying Newton's method to solve the $K+1$ exponential equations $\boldsymbol{F} = \mathbf{0}$, we can efficiently solve σ^t_{zk} and then compute the estimates of the expected accuracy increment as

$$\tilde{I}(t, x, z, \lambda) = (\max_k q^t_{zk} \cdot e^{\sigma^t_{zk}} - \max_k q^t_{zk})/N. \tag{4.18}$$

4.2.2 Task Assignment

To suppress the reinforcement between the uncertainties of true labels and worker models, we heuristically set an upper bound for the uncertainty of the inferred true labels. When the number of labels is very small, the uncertainty upper bound should be very low to prevent the misjudgment of true labels and the corresponding worker models. If the upper bound is not reached, we will force the next-step task assignment to stay at the same task to ensure accurate estimates of true labels and worker models. We call this period as the exploration stage. When the number of labels is large enough, few wrong labels cannot cause us to make severe mistakes in the inference of worker models. In this stage, to boost the label accuracy to the utmost, our mechanism should be able to freely select the task assignment that can bring the largest expected accuracy increment, which requires a high uncertainty upper bound.

Thus, we introduce the logistic function and formally write the uncertainty upper bound of the inferred true labels as

$$1 - \max_{k\in[K]} q^t_{j_{t-1}k} \leq \frac{K-1}{K} \cdot \frac{1}{1+\exp\left[-a(t-b)\right]} \tag{4.19}$$

where the left-hand side denotes the uncertainty measurement of task i_{t-1}. On the right-hand side, the coefficient $\frac{K-1}{K}$ is used because the minimal value of $\max_{k\in[K]} q^t_{j_{t-1}k}$ can be theoretically proven to be $1/K$. In addition, the uncertainty parameters, a and b, adjust the length of our exploration stage, respectively. We will decide the values of these two parameters through empirical experiments in the next section. Based on this uncertainty upper bound, we can define the scope of task assignment at step t as

$$Q(t) = \begin{cases} \{\langle i,j\rangle | j = j_{t-1}, s^t_{ij} = 0\} & \text{Cond.1 \& 2} \\ \{\langle i,j\rangle | s^t_{ij} = 0\} & \text{Otherwise} \end{cases} \tag{4.20}$$

where condition 1 denotes the violation of Equation 4.19. Condition 2 denotes that the number of workers who have labeled task j_{t-1} is smaller than M. In other words, there are still some workers for task j_{t-1} to choose. In addition, at step $t=0$, we use the convention that $j_{-1} = 1$ to start the assignment from the first task.

In addition, the obtained label λ from worker x can be any value in $\{1,\dots,K\}$. Considering the uncertainty of online inference, we use the upper confidence bound to replace the expectation in the expected error reduction strategy. For the K-dimensional distribution over K possible values, the upper confidence of the expected accuracy increment, $\tilde{G}(t,x,z)$, can be computed as the conditional value-at-risk which satisfies [68]:

$$\begin{aligned} \tilde{G}(t,x,z) &= \max_{q_\lambda \geq 0, \lambda\in[K]} \tilde{I}(t,x,z,\lambda)\cdot q_\lambda \\ \text{s.t.} \quad & q_\lambda \leq \mathbb{P}(s^{t+1}_{xz} = \lambda)/\gamma(t), \quad \sum\nolimits_\lambda q_\lambda = 1 \end{aligned} \tag{4.21}$$

where $\mathbb{P}(s^{t+1}_{xz} = \lambda) = \sum_k q^t_{zk} c^t_{jk\lambda}$ denotes the probability that the obtained label is λ. Besides, $\gamma(t)$ denotes the risk level, and $1-\gamma$ equals the required confidence level. We will empirically decide the $\gamma(t)$ function in the next section. Lastly, our task assignment strategy, uncertainty modulated optimistic assignment (UMOA), decides the task assignment at step t by the following equation:

$$\langle i_t, j_t\rangle = \arg\max_{\langle x,z\rangle\in Q(t)} \tilde{G}(t,x,z). \tag{4.22}$$

To summarize, our task assignment strategy first computes the expected accuracy increment $\tilde{I}(t, x, z, \lambda)$ corresponding to each possible label λ by using Newton's method to solve Equation 4.17. This design avoids iteratively solving Equations 4.7 and 4.8, which will significantly improve the computation efficiency. Then, we optimistically calculate the expected accuracy increment $\tilde{G}(t, x, z)$ corresponding to assignment $\langle x, z\rangle$ as the upper confidence bound of $\tilde{I}(t, x, z, \lambda)$. Meanwhile, we introduce an upper bound for the uncertainty of true labels, and modulate the scope of task assignments by comparing the current uncertainty measurement and the upper bound. By doing so, we can suppress the reinforcement between the uncertainties of true labels and worker models, which avoids wrongly judging and choosing low-quality workers. Thus, the label accuracy growth rate can get boosted. Furthermore, we wish to again highlight the importance of uniformly formulating different worker models as the basis of workers' labeling behavior space $\mathcal{B}$. Without $\mathcal{B}$, we will have to re-derive Equations 4.7, 4.8, 4.14 and 4.17 as well as the Jacobi matrix required in Newton's method [8] when changing to another model of workers, which is not trivial to achieve.

4.3 Experimentation

In this section, we first present four popular worker models to demonstrate the proposed basis of workers' labeling behavior space. Then, we explain how to set the risk level function $\gamma(t)$ and the uncertainty parameters a and b. Since our task assignment strategy is very robust to parameter values, we actually use the same parameter settings in all the following comparison experiments. Next, we compare the computation efficiency of different task assignment strategies to show the importance of our efficient prediction algorithm. Lastly, we compare the label accuracy by employing the four worker models and four Mechanical Turk datasets.

4.3.1 Popular Worker Models

We firstly discuss two worker models for binary labels ($K = 2$):

- One-Coin Model [12] assumes workers to have a fixed probability to provide correct labels for all tasks, which requires $C_j(1,1) = C_j(2,2)$ and $C_j(1,2) = C_j(2,1)$. Thus, the basis matrix set $\mathcal{B}$ corresponding to this model is:

$$B_{11} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad B_{12} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

- Two-Coin Model [66] directly uses the confusion matrix to depict workers' labeling behaviors. Thus, the corresponding basis matrix set $\mathcal{B}$ satisfies:

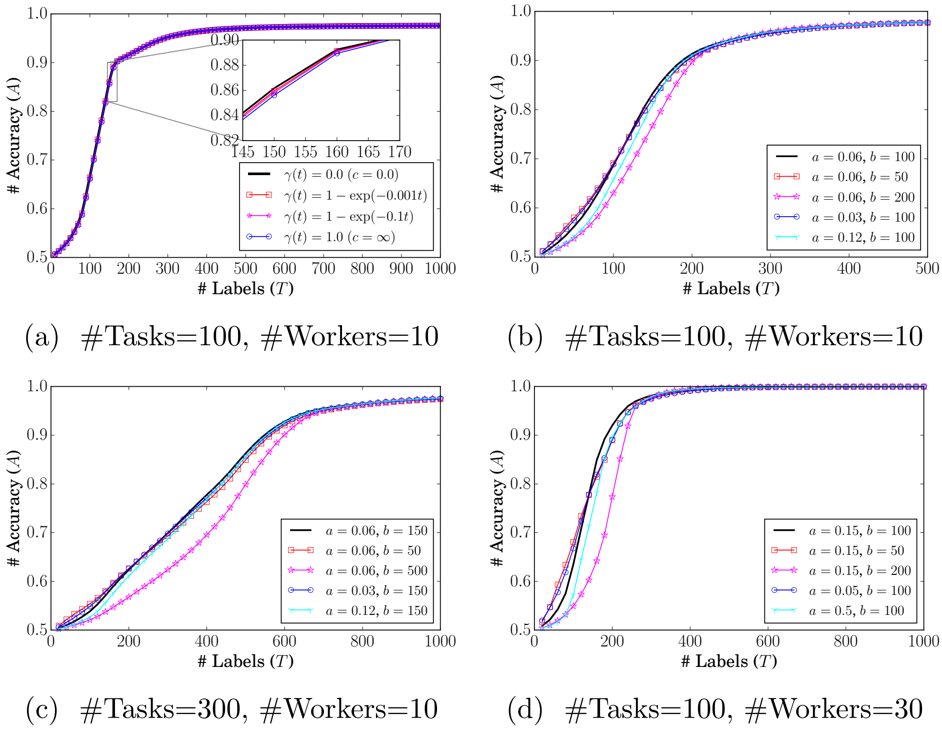
$$B_{11} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad B_{12} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$
$$B_{21} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \quad B_{22} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

Then, we consider the more complex triple-valued labels ($K = 3$):

- No-Preference Model [92] assumes that workers have no preference over any label value but workers' labeling behaviors depend on the true labels of tasks. In other words, workers will choose the two possible wrong labels with the same probability, which is equivalent to requiring $C_j(1,2) = C_j(1,3)$, $C_j(2,1) = C_j(2,3)$ and $C_j(3,1) = C_j(3,2)$. Thus, the corresponding basis matrix set $\mathcal{B}$ satisfies:

$$B_{11} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B_{12} = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B_{21} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix},$$
$$B_{22} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \quad B_{31} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad B_{32} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}.$$

- Middle-Preference Model [69] considers the psychological observation that if there are three options, human workers usually have special preference for the middle one. In this case, we can assume that workers have the same labeling behavior for the other two options, which requires $C_j(1,1) = C_j(3,3)$, $C_j(1,2) = C_j(3,2)$ and $C_j(1,3) = C_j(3,1)$. Thus, the basis matrix set $\mathcal{B}$ corresponding to



(a) #Tasks=100, #Workers=10 (b) #Tasks=100, #Workers=10

(c) #Tasks=300, #Workers=10 (d) #Tasks=100, #Workers=30

Figure 4.1: Experiments on different parameter settings of our mechanism

this model is:

$$B_{11} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad B_{12} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad B_{13} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix},$$
$$B_{21} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B_{22} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}.$$

4.3.2 Empirical Study on Parameter Settings

In Figure 4.1, we present the label accuracy comparison between different settings of $\gamma(t)$, a and b. Here, we use the popular one-coin worker model and binary labels as our testbed. Note that, in our experiments, we have also tried other worker models and get similar results. Thus, in this section, we only show the results of one-coin worker model to support our conclusions. Besides, since parameters are correlated with each other, we actually decide the parameter values by iteratively testing different combinations. In Figure 4.1, we only show the final results.

In our experiments, we firstly generate the true labels of tasks via the uniform distribution. Meanwhile, we generate workers' probabilities to provide the correct label by using the uniform distribution over $[0.5, 1.0]$. Then, we generate workers' labels based on the Bernoulli distribution. In Figure 4.1a, we employ the function family $\gamma(t) = 1 - \exp(-c \cdot t)$ to test the effects of changing risk level functions $\gamma(t)$ on label accuracy. The rationale of this function family roots in the fact that the uncertainty of online inference will decrease as more labels are collected. In this case, we can gradually increase the risk level. Besides, we calculate the label accuracy by running the simulation for 10000 rounds. We use so many rounds of simulations to support using the t-test to distinguish the slight differences between different risk level functions. The simulation results show that the effects of the risk level functions are very tiny. However, decreasing c from ∞ to 0 can slightly improve the label accuracy. Thus, we use $\gamma(t) = 0$ for our following experiments. In this case, we can simplify the linear programming in Equation 4.21 as looking for the maximum value, namely

$$\tilde{G}(t, x, z) = \max\nolimits_{\lambda} \tilde{I}(t, x, z, \lambda). \tag{4.23}$$

By doing so, we can also slightly boost the efficiency.

In Figure 4.1b, we study the effects of the uncertainty parameters a and b by running the simulation for 2000 rounds. From the figure, we can find that overly large a and b values must be avoided. They will lead to an overly long exploration stage, which has obviously adverse effects on the label accuracy. On the other hand, smaller a and b can slightly improve the label accuracy at the beginning because of a shorter exploration stage. However, an overly short exploration stage can also degrade the label accuracy in some cases, for example, when high-quality workers make some mistakes at the beginning. Due to this reason, we can observe that, when the number of labels is larger than 100, the accuracy growth rate corresponding to $a = 0.06$ and $b = 100$ is larger than the other two parameter settings with smaller a and b. Actually, when the number of tasks is 100, the number of collected labels is at least 100. Thus, the advantage when the number of labels is smaller than the number of taks is meaningless, and we will not show the accuracy comparison of this stage in following experiments.

From Figure 4.1b to 4.1c and 4.1d, we increase the numbers of tasks and workers by 2 times, respectively. The comparison between different parameter settings shows that the major threat to label accuracy is still the overly large b which will greatly lengthen the exploration stage. From Figure 4.1c, we can further obverse that as the number of tasks increases, the effects of a, which is responsible for finely adjust the length of exploration stage, will decrease. Besides, when the number of tasks increases, we should slightly increase b. This is because, after the exploration stage, our task assignment strategy will greedily label all tasks with reliable workers. In this case, the cost to correct the mistakes in the learned worker models will increase, which requires a smaller risk to wrongly judge workers and thus a longer exploration stage. From Figure 4.1d, we can find that we should synchronously increase a as the number of workers. This is because more workers require more steps to learn the correct worker models. To summarize, parameters a and b capture the effects of more workers and tasks, respectively. Note that, our conclusion is only suitable for the case when the number of tasks is far larger than the number of workers. This is usually satisfied in practice because a practical worker often needs to label hundreds of tasks in a crowd labeling market.

4.3.3 Computation Efficiency Comparison

Table 4.2 compares the average one-step time cost of our mechanism when different task assignment strategies are applied. The selected benchmarking task assignment strategies include:

- Round-Robin strategy [71] randomly assigns the same number of workers for all tasks;
- Uncertainty sampling [92] selects the most uncertain task at each step and excludes must-be-bad workers whose correction rate and the corresponding variance are lower than 0.55 and 0.05^2 respectively;
- Expected error reduction [59] greedily selects the task assignment that can bring the maximum one-step expected accuracy increment at each step;

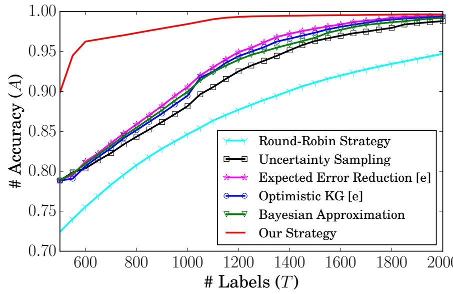
- Optimistic KG [12] calculates the optimistic knowledge gradient (KG) as the maximum expected accuracy increment of all possible next-step labels and selects the one with the highest optimistic KG at each step;

- Bayesian approximation [100] fixes the learned worker models, uses Bayes' theorem to predict the one-step expected accuracy increment, and selects the assignment with the maximal increment at each step.

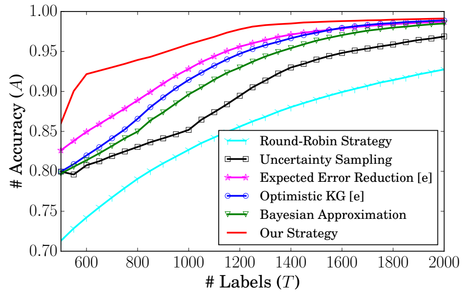
Table 4.2: Computation time of different task assignment strategies

Task Assignment Strategies	**One-Step Time Cost (ms)**		
	$(20, 10)$	$(60, 10)$	$(60, 20)$
Round-Robin Strategy	0.7	0.8	0.8
Uncertainty Sampling	0.9	1.2	0.9
Expected Error Reduction	**24**	**247**	**431**
Optimistic KG	**21**	**180**	**436**
Bayesian Approximation	**3.1**	**8.3**	**16**
Our Strategy	**1.5**	**3.2**	**5.6**

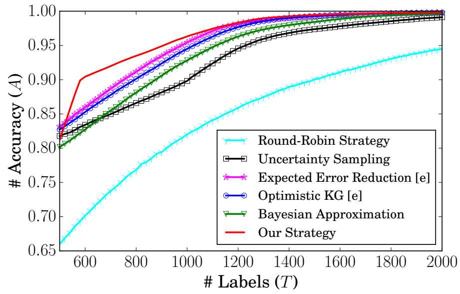
In Table 4.2, the brackets (N, M) denote that the numbers of tasks and workers are N and M, respectively. Our experiment settings here are the same as that in Figure 4.2a, which will be detailed in the next section. The time cost is estimated via running 100 rounds of experiments on Xeon CPU E5-1650 and collecting $M \cdot N$ labels in each round of experiments. From Table 4.2, we can conclude that our strategy is the most efficient among all prediction-based task assignment strategies (i.e. the last ones). The time complexity of our strategy, which uses the efficient prediction algorithm, is lower than $O(NM)$. By contrast, the time complexity of the expected error reduction and optimistic KG strategies is around $O(N^2M)$. In our following experiments with hundreds of tasks and thousands of labels to collect, the computation time of these two strategies is prohibitive. Thus, we also use our efficient prediction algorithm to improve the efficiency of these two strategies and mark them with [e] for distinction in the following sections. Besides, the computation efficiency of the Bayesian approximation strategy is acceptable because it approximately predicts



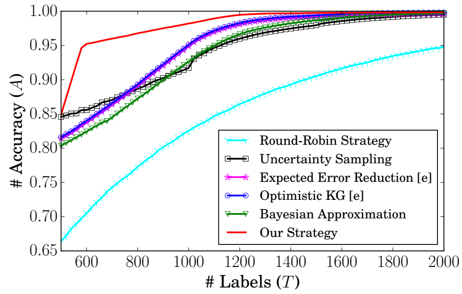
(a) One-Coin Worker Test 1



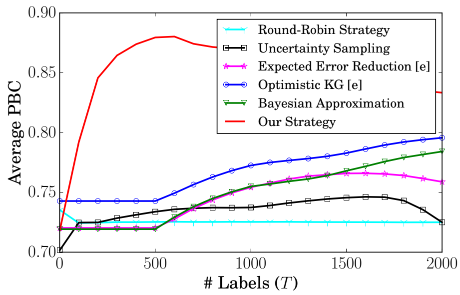
(b) Two-Coin Workers



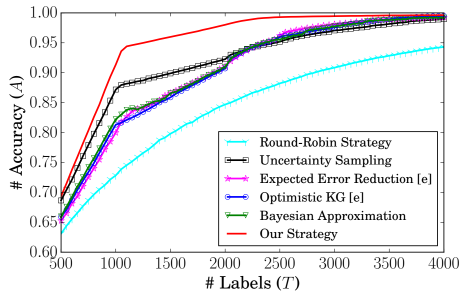
(c) No-Preference Workers



(d) Middle-Preference Workers



(e) One-Coin Worker Test 2



(f) One-Coin Worker Test 3

Figure 4.2: Label accuracy comparison on four popular worker models

the one-step expected accuracy increment by fixing workers' models. However, this approximation neglects the correlation between the estimates of worker models and true labels, which will degrade the label accuracy. We will empirically show this point in the next two sections.

4.3.4 Accuracy Comparison on Synthetic Data

To show the advantage of our strategy on accuracy growth rate, we conduct experiments on the four aforementioned worker models and present the results in Figure 4.2. Here, we set the number of experiments as 200 so that the differences between different strategies can be distinguished by t-test. For example, the p-values of the t-tests corresponding to $T = 1,000$ in Figure 4.2b between our strategy and benchmarking strategies are all below 1×10^{-10}. In our experiments, we firstly generate the true labels of tasks via the uniform distribution. Meanwhile, we generate worker models $\boldsymbol{\theta}_{jw}$ by using the Dirichlet distribution $\mathrm{Dir}(\boldsymbol{\alpha}_w)$, where $\alpha_{wl} = 4$ if there exists k satisfying $B_{wl}(k,k) = 1$ and $\alpha_{wl} = 1$ otherwise. Then, we compute workers' confusion matrices using Equation 4.2 and generate workers' labels based on the categorical distribution.

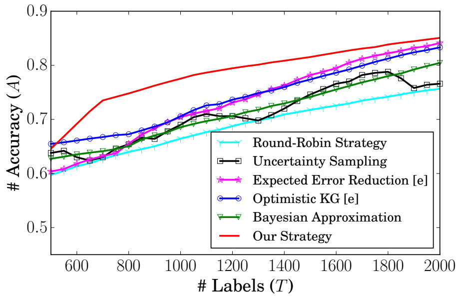
In Figures 4.2a-d, we set the numbers of tasks and workers as 500 and 10, respectively. Considering the fact that we should collect at least 500 labels for 500 tasks, we start the accuracy comparison from #Labels= 500. Actually, the accuracy of all strategies equals $1/K$ when #Labels= 0. From these four figures, we can conclude that our task assignment strategy can reach high accuracy (e.g. 0.95) with much less number of labels than all the benchmarking strategies. Meanwhile, Expected Error Reduction [e] and Optimistic KG [e], which combines the classic strategies with our efficient prediction algorithm, also perform better than the other three strategies. On the other hand, the advantage of our strategy on the single-coin model is larger than that on the other three models. The reason for this phenomenon is that, for example, the three rows of the confusion matrix corresponding to the no-preference model are independently generated. Since the true label is also unknown, it becomes difficult to judge whether a worker is better than another in this case (e.g. $c_{111} > c_{211}$ while $c_{122} < c_{222}$).

To further validate the advantage of our strategy, we conduct the other two sets of experiments based on Figure 4.2a. Firstly, in the one-coin model, θ_{j11} denotes worker j's probability of being correct (PBC). In Figure 4.2e, we set $\theta_{j11} = j * 0.05 + 0.45$ and compute the average PBC of employed workers up to step n as $\sum_{t=1}^{n} \theta_{j_t 11}/n$. The results show that our strategy has the highest average PBC, which means that our strategy successfully identifies and employs higher-quality workers. This observation explains the rationale behind the advantage of our strategy. Secondly, in Figure 4.2f, we increase the number of tasks to 1000, which requires two times of labels to be collected. Compared with expected error reduction, our strategy reduces #labels to reach 0.95 by 50% in both Figures 4.2a and f, which reveals the good robustness of our strategy to the scale of collected labels. In fact, as the number of required labels T increases, the ratio between the length of the exploration stage in our strategy and T will decrease, which is helpful to enhance our advantage over benchmarking strategies. However, running 200 experiments is quite time-consuming. Thus, we keep the number of tasks as 500 in our following experiments on MTurk datasets.

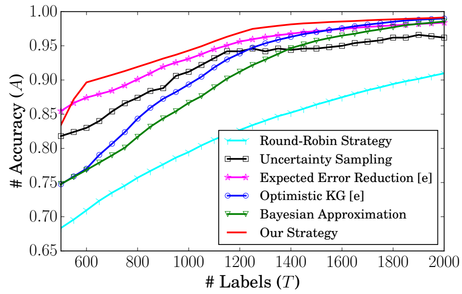
4.3.5 Accuracy Comparison on MTurk Datasets

In addition to using the synthetic data, we employ four popular MTurk datasets as our testbeds in Figure 4.3. HCB dataset contains the judgments on the relevance between Web pages and search queries [7]. RTE dataset workers need to check whether a hypothesis sentence can be inferred from the provided sentence [81]. SPE dataset consists of the positive or negative labels of movie reviews [61]. ACC dataset workers classify websites according to their adult contents [38]. A challenge of using these datasets is that they are very sparse and the label corresponding to the required assignment may not exist. Thus, we employ the famous SQUARE library (Version 2.0) to complement those non-existent labels [72]. It uses the true labels to compute workers' confusion matrices at first and then generates workers' labels accordingly. Since the SQUARE library relies on the two-coin worker model, we also use it for the computation of our mechanism. Besides, to facilitate computation, we select the first 500 tasks and 10 workers in the experiments with the first three datasets. The ACC dataset has only 333 tasks with known true labels that have been labeled by workers. Thus, we set the number of tasks as 333 in Figure 4.3d. From Figure 4.3, we can conclude that our strategy always significantly outperforms the round-robin, uncertainty sampling and Bayesian approximation strategies.[6] The advantage of our strategy over the expected error reduction and optimistic KG strategies, which needs to employ our efficient prediction algorithm to solve the efficiency bottleneck, always exists but may become small in some cases. This is because, if the differences between the two rows of the confusion matrix in the two-coin worker model are very large, it will be difficult for our strategy to distinguish which worker is the best one.

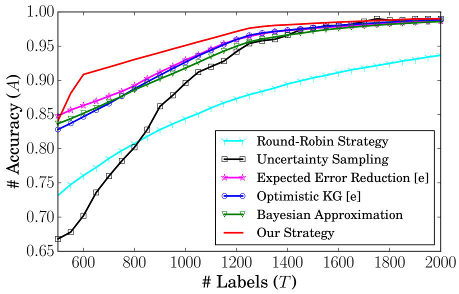
[6] The t-test also supports our conclusion. For example, in Figure 4.3b, if $T = 1,000$, the p-values of the t tests between UMOA and other strategies are all below 1×10^{-4}.



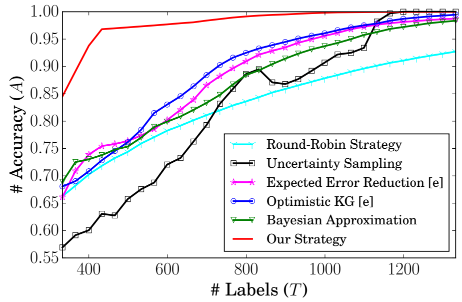
(a) HCB dataset



(b) RTE dataset



(c) SPE dataset



(d) ACC dataset

Figure 4.3: Label accuracy comparison on four popular MTurk datasets

4.4 Summary

In this chapter, we present an active task assignment mechanism which uses online variational inference to improve task assignment in microtask crowdsourcing. Our core contribution is the novel prediction-based task assignment strategy. To improve accuracy growth rate and efficiency, it keeps the prediction optimistic, uses the uncertainty measurement of inference to modulate the scope of task assignment, and predicts the future via an approximation algorithm of variational inference. Our another contribution is a unified formulation of different worker models so that our mechanism can incorporate any of the worker models. In our experiments, we first decide the parameters in our mechanism by conducting experiments on one typical worker model and then apply our mechanism to four worker models and four MTurk datasets. The comparison with various benchmarking algorithms show that our active task assignment mechanism not only requires the least labels for high accuracy but also has better efficiency than existing prediction-based task assignment approaches.

Chapter 5

Inference Aided Reinforcement Learning Reward Mechanism

This chapter focuses on the reward payment process of microtask crowdsourcing. We propose a novel inference-aided reinforcement mechanism, aiming to avoid the impractical requirement for workers' rationality and the prior knowledge about workers in existing reward mechanisms [33, 32]. The high level idea is as follows: we collect data in a sequential fashion. At each step, we assign workers a certain number of tasks and estimate the true labels and workers' strategies from their labels. Relying on the above estimates, a reinforcement learning (RL) algorithm is proposed to uncover how workers respond to different levels of offered payments. The RL algorithm determines the payments for the workers based on the collected information up-to-date. By doing so, our mechanism not only incentivizes (non-)rational workers to provide high-quality labels but also dynamically adjusts the payments according to workers' responses to maximize the data requester's cumulative utility. Applying standard RL solutions here is challenging, due to unobservable states (workers' labeling strategies) and reward (the aggregated label accuracy) which is further due to the lack of ground-truth labels. Leveraging standard inference methods seems to be a plausible solution at the first sight (for the purposes of estimating both the states and reward), but we observe that existing methods tend to over-estimate the aggregated label accuracy, which would mislead the superstructure RL algorithm.

We address the above challenges and make the following contributions: (1) We propose a Gibbs sampling augmented Bayesian inference algorithm, which estimates workers' labeling strategies and the aggregated label accuracy, as done in most existing inference algorithms, but significantly lowers the estimation bias of labeling accuracy. This lays a strong foundation for constructing correct reward signals, which are extremely important if one wants to leverage reinforcement learning techniques. (2) A reinforcement incentive learning (RIL) algorithm is developed to maximize the data requester's cumulative utility by dynamically adjusting incentive levels according to workers' responses to payments. (3) We prove that our Bayesian inference algorithm and RIL algorithm are incentive compatible (IC) at each step and in the long run, respectively. Here, we formally define incentive compatibility as follows:

Definition 5.1 *Suppose there are a group of agents and we expect those agents to follow the desired strategy s^*. For any agent i, we use $-i$ to denote all the other agents except for agent i. Then, incentive compatibility formally means*

$$u_i(s_i^*, s_{-i}^*) \geq u_i(s_i, s_{-i}^*) \tag{5.1}$$

where u_i and s_i denotes agent i's utility and strategy, respectively. In other words, given other agents follow the desired strategy, agent i can only get the maximal utility by also following the desired strategy.

In our problem, we desire workers to exert high efforts to observe labels at first and then truthfully report the observations, because the data requester can only get the high-quality labels when workers follow this strategy. (4) Experiments are conducted to test our mechanism, which shows that our mechanism performs consistently well under different worker models. Meanwhile, compared with the state-of-the-art peer prediction solutions, our Bayesian inference aided mechanism can improve the robustness and lower the variances of payments.

The following parts of this chapter are arranged as follows. In Section 5.1, we formally formulate the model of the reward payment step of microtask crowdsourcing. Then, in section 5.2, we describe our inference-aided reinforcement reward mechanism. Section 5.3 is devoted to proving the IC property of the proposed mechanism. In Section 5.4, a set of experiments are conducted to validate the advantages of our

Table 5.1: Key notations used in Chapter 5

M	Number of tasks at each step	N	Number of workers
$L_i^t(j)$	Worker i's label for task j at step t		
$\mathcal{L}^t(j)$	The real true label of task j at step t		
eft_i^t	Worker i's probability of exerting high efforts		
rpt_i^t	Worker i's probability of reporting truthfully		
$\mathbb{P}_H$	Probability to observe the correct label when exerting high efforts		
$\mathbb{P}_L$	Probability to observe the correct label when exerting low efforts		
m_i^t	Number of tasks assigned to worker i at step t		
c_H	Workers' cost to exert high efforts		
c_L	Workers' cost to exert low efforts		
u_i^t	Utility of worker i at step t	A^t	Label accuracy at step t
$P_i^t(j)$	The payment to worker i for task j at step t		
$F(\cdot)$	The function that maps label quality to data requester's utility		
r_t	Data requester's utility at step t		
η	The tunable parameter that balances label quality and costs		
$\mathsf{sc}_i^t(j)$	The probability that worker i's label for task j is correct		
a_t	The scaling factor in our payment rule		
b	The base payment in our payment rule		
$\mathcal{A}$	The set of possible values for a_t		
$\boldsymbol{L}^t$	Matrix representation of the collected labels, $= [L_i^t(j)]$		
$\mathcal{L}^t = [\mathcal{L}^t(1), \ldots, \mathcal{L}^t(M)]$, $\boldsymbol{p} = [\mathbb{P}_1, \ldots, \mathbb{P}_N]$			
$\boldsymbol{\tau}$	The distribution of real true labels, $= [\tau_1, \tau_2]$		
$\delta_{ijk}^t = \mathbb{1}(L_i^t(j) = k)$ and $\xi_{jk}^t = \mathbb{1}(\mathcal{L}^t(j) = k)$ are the indicator functions.			
α_1, α_2, β_1, β_2	Parameters of Dirichlet priors		
$\hat{\boldsymbol{\alpha}}_i$, $\hat{\boldsymbol{\beta}}$	Parameters of the posterior distribution $\mathbb{P}(\mathcal{L}\vert\boldsymbol{L})$		
W, W_0	Number of samples in Gibbs sampling and the burn-in process		
s	State in the Markov decision process		
$\hat{s}$	The augmented state representation used in our mechanism		
$Q^\pi(\hat{s}_t, a_t)$	The Q-value of choosing action a_t at state $\hat{s}_t$ under policy π		
$V^\pi(\hat{s}_t)$	The value of state $\hat{s}_t$ under policy π		
γ	Discount factor in the Markov decision process		
ϵ_t, ϵ_π	Gaussian noise	$\boldsymbol{r}$	Vector representation of r_t

mechanism on rewards and the ability to perform consistently well under different worker models. Finally, Section 5.5 summarizes this chapter. In addition, we list the key notations used in this chapter as Table 5.1. Note that, for the simplification of notations, we omit the subscript t in some subsections of this chapter.

5.1 Problem Formulation

This chapter considers the following data collection problem via microtask crowdsourcing: at each discrete time step $t = 1, 2, ...$, one requester assigns M tasks with binary answer space $\{1, 2\}$ to $N \geq 3$ candidate workers to label. Workers receive payments for submitting a label for each task. We use $L_i^t(j)$ to denote the label worker i generates for task j at time t. For simplicity of computation, we reserve $L_i^t(j) = 0$ if j is not assigned to i. Furthermore, we use $\boldsymbol{\mathcal{L}}$ and $\boldsymbol{L}$ to denote the set of ground-truth labels and the set of all collected labels respectively.

The generated label $L_i^t(j)$ depends both on the ground-truth $\mathcal{L}(j)$ and worker i's strategy, which is mainly determined by two factors, exerted effort level (high or low) and reporting strategy (truthful or deceitful). Accommodating the notation commonly used in reinforcement learning, we also refer to worker i's strategy as his/her internal state. At any given time for any task, workers at their will adopt an arbitrary combination of effort level and report strategy. Specifically, we define $\mathsf{eft}_i^t \in [0, 1]$ and $\mathsf{rpt}_i^t \in [0, 1]$ as worker i's probability of exerting high efforts and reporting truthfully for task j, respectively. The strategy space of worker i can be formally written as $(\mathsf{eft}_i^t, \mathsf{rpt}_i^t) \in [0, 1]^2$. We desire all workers to follow $(\mathsf{eft}_i^t, \mathsf{rpt}_i^t) = (1.0, 1.0)$ to generate labels. Furthermore, following existing literature [15, 51], we assume that tasks are homogeneous and workers share the same probability of generating the correct labels if they exert the same level of efforts - we denote these probabilities as $\mathbb{P}_H$ and $\mathbb{P}_L$.[1] We assume $\mathbb{P}_H > \mathbb{P}_L = 0.5$. We further assume that the cost for any worker i to exert low effort is $c_L = 0$, whereas exerting high effort incurs $c_H \geq 0$.[2] These cost parameters

[1]For simplicity we have assumed that the labeling accuracy is ground-truth label independent.

[2]We make such assumption for simplicity. Our analysis can be extended to the case where both $c_L, c_H \geq 0$, as long as $c_H \geq c_L$.

stay unknown to the data requester. Worker i's probability of being correct (PoBC) at time t for any given task is then given as

$$\begin{aligned}\mathbb{P}_i^t = \; & \mathsf{rpt}_i^t \cdot \mathsf{eft}_i^t \mathbb{P}_H + (1-\mathsf{rpt}_i^t)\cdot \mathsf{eft}_i^t (1-\mathbb{P}_H) + \\ & \mathsf{rpt}_i^t \cdot (1-\mathsf{eft}_i^t)\mathbb{P}_L + (1-\mathsf{rpt}_i^t)\cdot(1-\mathsf{eft}_i^t)(1-\mathbb{P}_L)\end{aligned} \tag{5.2}$$

Suppose we assign $m_i^t \leq M$ tasks to worker i at time step t, then his or her utility would be

$$u_i^t = \sum\nolimits_{j=1}^{M} P_i^t(j) - m_i^t \cdot c_H \cdot \mathsf{eft}_i^t \tag{5.3}$$

where $P_i^t(j)$ denotes our payment to worker i for task j at time t, for which we will provide further detail in Section 5.2.1.

At the beginning of each step, the requester and workers mutually agree to a certain rule of payment determination, which would not be changed until the next time step. The workers are self-interested and may change their strategies according to the expected utility $\mathbb{E}u_i^t$ he/she can get. It is not surprising that workers' different strategies would lead to different PoBCs and then different qualities of labels. After collecting the generated labels, the data requester infers the true labels $\tilde{\mathcal{L}}^t(j)$ by running a certain inference algorithm. The aggregate label accuracy A^t and the data requester's utility r_t are defined as follows:

$$\begin{aligned} A^t &= \frac{1}{M}\sum\nolimits_{j=1}^{M} 1\left[\tilde{\mathcal{L}}^t(j) = \mathcal{L}^t(j)\right] \\ r_t &= F(A^t) - \eta \sum\nolimits_{i=1}^{N}\sum\nolimits_{j=1}^{M} P_i^t(j)\end{aligned} \tag{5.4}$$

where $F(\cdot)$ is a non-decreasing monotonic function mapping accuracy to utility and $\eta > 0$ is a tunable parameter balancing label quality and costs. Naturally, $F(\cdot)$ function is non-deceasing as higher accuracy is preferred.[3]

5.2 Reward Mechanism for Crowdsourcing

Our mechanism mainly consists of three components: one-step payment rule, Bayesian inference and reinforcement incentive learning (RIL); see Figure 5.1 for the overall layout, where estimated values are denoted with tildes.

[3] We use r to denote the data requester's utility as it is used as the reward in our RL algorithm. See Section 5.2.3 for details.

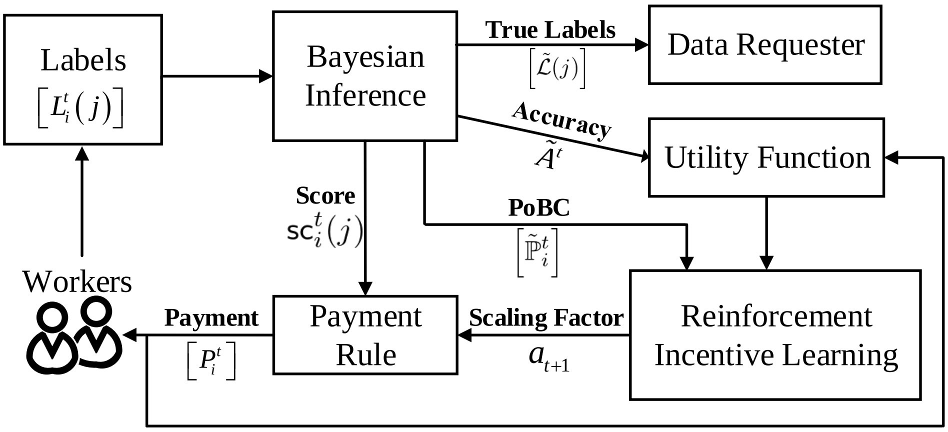


Figure 5.1: Overview of our reward mechanism.

The payment rule ensures that reporting truthfully and exerting high efforts is the utility-maximizing strategy for all workers at any given time. This property is termed as one-step IC. Furthermore, our reward mechanism as a whole guarantees that this is also the utility-maximizing strategy for all workers in the long-run, which is termed as long-term IC. We kindly refer readers to Section 5.3.2 for the theoretical proof. This property prevents strategic manipulations from workers, which brings more long-term benefits to them by sacrificing short-term gains, or the other way around, or both. The Bayesian inference algorithm is responsible for estimating the true labels, workers' PoBCs and the aggregate label accuracy from the collected labels at each time step. It utilizes soft Dirichlet priors and Gibbs sampling to prevent overestimation of accuracy when workers generate poor-quality labels. RIL adjusts the payment rule based on the historical data of rewards, workers' PoBCs and aggregate labels' accuracy, aiming to optimally balance the utility gain from high accuracy and loss from large payments, which corresponds to $F(A^t)$ and $\sum_i \sum_j P_i^t(j)$ in Equation 5.4 respectively.

5.2.1 Payment Rule

We design the payment to worker i for his/her label on task j as

$$P_i^t(j) = a_t \cdot [\mathsf{sc}_i^t(j) - 0.5] + b \tag{5.5}$$

where $\mathsf{sc}_i^t(j)$ denotes the score of worker i's label for task j—i.e. the probability that worker i's label for task j is correct, which will be computed by our Bayesian inference algorithm (details in next subsection). $b \geq 0$ is a constant representing the fixed base payment even if the worker purely genrates random labels. We use $a_t \in \mathcal{A}$ to denote the scaling factor, determined by RIL at the beginning of every step t. We assume $\mathcal{A}$ is a finite set $|\mathcal{A}| < \infty$. Note that, in practice, we cap $P_i^t(j)$ to be non-negative.

5.2.2 Bayesian Inference

For the simplicity of notations, we omit the superscript t in this subsection. The motivation for designing our own Bayesian inference algorithm is as follows. We ran several preliminary experiments using popular inference algorithms used in the literature. Our empirical studies reveal that those methods tend to heavily bias towards overestimating the accuracy when the quality of labels is very low. For example, in our experiments in Section 5.4.1, when there are 10 workers and $\mathbb{P}_i^t = 0.55$, the estimated label accuracy of the EM estimator [66] stays at around 0.9 while the real accuracy is only around 0.5. This heavy bias will lead the data requester's utility r_t to be miscalculated, causing two bad consequences. First, it induces bad incentive property, as workers with poor labeling accuracy now enjoy good estimates. Secondly, this potentially misleads RIL, as r_t is used as reward.

For the simplicity of notations, we omit the superscript t in this subsection. It is not hard to figure out the joint distribution of the collected labels $\boldsymbol{L}$ and the true labels $\boldsymbol{\mathcal{L}}$

$$\mathbb{P}(\boldsymbol{L}, \boldsymbol{\mathcal{L}} | \boldsymbol{p}, \boldsymbol{\tau}) = \prod_{j=1}^{M} \prod_{k=1}^{2} \left\{ \tau_k \prod_{i=1}^{N} \mathbb{P}_i^{\delta_{ijk}} (1 - \mathbb{P}_i)^{\delta_{ij(3-k)}} \right\}^{\xi_{jk}}$$

where $\boldsymbol{p} = [\mathbb{P}_1, \dots, \mathbb{P}_N]$ and $\boldsymbol{\tau} = [\tau_1, \tau_2]$. τ_1 and τ_2 denote the distribution of true label 1 and 2, respectively. Besides, $\delta_{ijk} = \mathbb{1}(L_i(j) = k)$ and $\xi_{jk} = \mathbb{1}(\mathcal{L}(j) = k)$. Here, we assume Dirichlet priors $\mathrm{Dir}(\cdot)$ for $\mathbb{P}_i$ and $\boldsymbol{\tau}$

$$[\mathbb{P}_i, 1 - \mathbb{P}_i] \sim \mathrm{Dir}(\alpha_1, \alpha_2) \ , \ \boldsymbol{\tau} \sim \mathrm{Dir}(\beta_1, \beta_2).$$

Then, the joint distribution of $\boldsymbol{L}$, $\boldsymbol{\mathcal{L}}$, $\boldsymbol{p}$ and $\boldsymbol{\tau}$

$$\begin{aligned}\mathbb{P}(\boldsymbol{\mathcal{L}}, \boldsymbol{L}, \boldsymbol{p}, \boldsymbol{\tau}) &= \mathbb{P}(\boldsymbol{\mathcal{L}}, \boldsymbol{L} | \boldsymbol{p}, \boldsymbol{\tau}) \cdot \prod_{i=1}^{N} \mathrm{Dir}(\alpha_1, \alpha_2) \cdot \mathrm{Dir}(\beta_1, \beta_2) \\ &= \frac{1}{B(\boldsymbol{\beta})} \prod_{k=1}^{K} \tau_k^{\hat{\beta}_k^* - 1} \cdot \prod_{i=1}^{N} \frac{1}{B(\boldsymbol{\alpha})} \mathbb{P}_i^{\hat{\alpha}_{i1}^* - 1} (1 - \mathbb{P}_i)^{\hat{\alpha}_{i2}^* - 1}\end{aligned}$$

where $B(x,y)=(x-1)!(y-1)!/(x+y-1)!$ denotes the beta function, and

$$\begin{aligned}\hat{\alpha}^*_{i1} &= \sum\nolimits_{j=1}^{M}\sum\nolimits_{k=1}^{K}\delta_{ijk}\xi_{jk}+\alpha_1\\ \hat{\alpha}^*_{i2} &= \sum\nolimits_{j=1}^{M}\sum\nolimits_{k=1}^{K}\delta_{ij(3-k)}\xi_{jk}+\alpha_2\\ \hat{\beta}^*_{k} &= \sum\nolimits_{j=1}^{M}\xi_{jk}+\beta_k.\end{aligned}$$

The convergence of our inference algorithm requires $\alpha_1>\alpha_2$. To simplify the theoretical analysis, we set $\alpha_1=1.5$ and $\alpha_2=1$ in this chapter. Note that, as long as $\alpha_1>\alpha_2$, our mechanism works well under the different settings of α_1 and α_2. Fixing the values of α_1 and α_2 is because our ability to theoretically analyse the transcendental function $H(\cdot)$ defined in Appendix B.2 (Equation 7.28) is limited. Meanwhile, we employ the uniform distribution for $\boldsymbol{\tau}$ by setting $\beta_1=\beta_2=1$. In this case, we can conduct marginalization via integrating the joint distribution $\mathbb{P}(\mathcal{L},\boldsymbol{L},\boldsymbol{p},\boldsymbol{\tau})$ over $\boldsymbol{p}$ and $\boldsymbol{\tau}$ as

$$\mathbb{P}(\mathcal{L},\boldsymbol{L}|\boldsymbol{\alpha},\boldsymbol{\beta})=\frac{B(\hat{\boldsymbol{\beta}})}{B(\boldsymbol{\beta})}\cdot\prod\nolimits_{i=1}^{N}\frac{B(\hat{\boldsymbol{\alpha}}_i)}{[B(\boldsymbol{\alpha})]^2} \tag{5.6}$$

where $\hat{\boldsymbol{\alpha}}_i=[\hat{\alpha}^*_{i1}+\alpha_1-1,\hat{\alpha}^*_{i2}+\alpha_2-1]$ and $\hat{\boldsymbol{\beta}}=[\hat{\beta}^*_1+\beta_1-1,\hat{\beta}^*_2+\beta_1-1]$. Following Bayes' theorem, we can know that

$$\mathbb{P}(\mathcal{L}|\boldsymbol{L})=\frac{\mathbb{P}(\mathcal{L},\boldsymbol{L}|\boldsymbol{\alpha},\boldsymbol{\beta})}{\mathbb{P}(\mathcal{L}|\boldsymbol{\alpha},\boldsymbol{\beta})}\propto B(\hat{\boldsymbol{\beta}})\prod_{i=1}^{N}B(\hat{\boldsymbol{\alpha}}_i). \tag{5.7}$$

Algorithm 5: Gibbs Sampling aided Bayesian Inference

```
Input   : the collected labels L, the number of samples W
Output: the sample sequence S
begin
    S ← ∅, Initialize L̃ with the uniform distribution
    for s = 1 to W do
        for j = 1 to M do
            Compute P[L(j) = k] by letting L(−j) = L̃(−j)
            L̃(j) ← Sample {1, 2} with P[L(j) = k]
        Append L̃ to the sample sequence S
```

Note that it is generally hard to derive an explicit formula for the posterior distribution of a specific task j's ground-truth from the conditional distribution $\mathbb{P}(\mathcal{L}|\boldsymbol{L})$. We thus resort to Gibbs sampling for the inference. More specifically, according to Bayes' theorem, we know that the conditional distribution of task j's ground-truth $\mathcal{L}(j)$ satisfies $\mathbb{P}[\mathcal{L}(j)|\boldsymbol{L}, \mathcal{L}(-j)] \propto \mathbb{P}(\mathcal{L}|\boldsymbol{L})$, where $-j$ denotes all tasks excluding j. Leveraging this, we generate samples of the true label vector $\mathcal{L}$ following Algorithm 5. At each step of the sampling procedure (lines 6-7), Algorithm 5 first computes $\mathbb{P}[\mathcal{L}(j)|\boldsymbol{L}, \mathcal{L}(-j)]$ and then generates a new sample of $\mathcal{L}(j)$ to replace the old one in $\tilde{\mathcal{L}}$. After traversing through all tasks, Algorithm 5 generates a new sample of the true label vector $\mathcal{L}$. Repeating this process for W times, we get W samples, which is recorded in $\mathcal{S}$. Here, we write the s-th sample as $\tilde{\mathcal{L}}^{(s)}$. Since Gibbs sampling requires a burn-in process, we discard the first W_0 samples in $\mathcal{S}$. After doing so, we calcualte worker i's score on task j as

$$\mathsf{sc}_i^t(j) = \sum\nolimits_{s=W_0+1}^{W} \mathbb{1}(\tilde{\mathcal{L}}^{(s)}(j) = \mathcal{L}_i(j))/(W - W_0) \tag{5.8}$$

where $\mathbb{1}(\tilde{\mathcal{L}}^{(s)}(j) = \mathcal{L}_i(j))$ is 1 only when the worker i's label for task j is the same as the s-th true label sample for task j. In other words, we are computing the score by calculating the ratio of true label samples that are the same as workers' labels. Similarly, we estimate worker i's PoBC $\mathbb{P}_i$ as

$$\tilde{\mathbb{P}}_i = \frac{\sum\limits_{s=W_0+1}^{W} \left[2\alpha_1 - 1 + \sum\limits_{j=1}^{M} \mathbb{1}(\tilde{\mathcal{L}}^{(s)}(j) = \mathcal{L}_i(j))\right]}{(W - W_0)\cdot(2\alpha_1 + 2\alpha_2 - 2 + m_i)} \tag{5.9}$$

and the distribution of true labels $\boldsymbol{\tau}$ as

$$\tilde{\tau}_k = \frac{\sum\limits_{s=W_0+1}^{W} \left[2\beta_1 - 1 + \sum\limits_{j=1}^{M} \mathbb{1}(\tilde{\mathcal{L}}^{(s)}(j) = k)\right]}{(W - W_0)\cdot(2\beta_1 + 2\beta_2 - 2 + M)}. \tag{5.10}$$

Furthermore, we define the log-ratio of task j as

$$\tilde{\sigma}_j = \log\frac{\mathbb{P}[\mathcal{L}(j) = 1]}{\mathbb{P}[\mathcal{L}(j) = 2]} = \log\left(\frac{\tilde{\tau}_1}{\tilde{\tau}_2}\prod_{i=1}^{N}\tilde{\lambda}_i^{\delta_{ij1}-\delta_{ij2}}\right) \tag{5.11}$$

where $\tilde{\lambda}_i = \tilde{\mathbb{P}}_i/(1 - \tilde{\mathbb{P}}_i)$. Finally, we decide the true label estimate $\tilde{\mathcal{L}}(j)$ as 1 if $\tilde{\sigma}_j > 0$ and as 2 if $\tilde{\sigma}_j < 0$. Correspondingly, the label accuracy A is estimated as

$$\tilde{A} = \mathbb{E}(A) = \frac{1}{M}\sum\nolimits_{j=1}^{M} e^{|\tilde{\sigma}_j|}\left(1 + e^{|\tilde{\sigma}_j|}\right)^{-1}. \tag{5.12}$$

For good inference accuracy, we require both W and W_0 to be large values, and in the rest of this chapter, we set $W = 1000$ and $W_0 = 100$ respectively.

5.2.3 Reinforcement Incentive Learning

In this subsection, we formally introduce our reinforcement incentive learning (RIL) algorithm, which adjusts the scaling factor a_t at each time step t. Viewed under the big picture, it serves as the glue to connect the other components in our mechanism (see the edges and parameters around RIL in Figure 5.1). In a reinforcement learning (RL) problem, an agent interacts with an unknown environment and attempts to maximize its cumulative collected reward [82, 83]. The environment is commonly formalized as a Markov Decision Process (MDP) defined as $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma \rangle$. At time t the agent is in state $s_t \in \mathcal{S}$ where it takes an action $a_t \in \mathcal{A}$ leading to the next state $s_{t+1} \in \mathcal{S}$ according to the transition probability kernel $\mathcal{P}$, which encodes $\mathbb{P}(s_{t+1} \mid s_t, a_t)$. In most RL problems, $\mathcal{P}$ is unknown to the agent. The agent's goal is to learn the optimal policy, a conditional distribution $\pi(a \mid s)$ that maximizes the sate's value function. The value function calculates the cumulative reward the agent is expected to receive given it would follow the current policy π after observing the current state s_t

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[\sum_{k=1}^{\infty} \gamma^k r_{t+k} \mid s_t = s\right].$$

Intuitively, it measures how preferable each state is given the current policy.

As a critical step towards improving a given policy, it is a standard practice for RL algorithms to learn a state-action value function (i.e. Q-function). Q-function calculates the expected cumulative reward if agent choose a in the current state and follows π thereafter

$$Q^{\pi}(s, a) = \mathbb{E}_{\pi}\left[\mathcal{R}(s_t, a_t, s_{t+1}) + \gamma V^{\pi}(s_{t+1}) \mid s_t = s, a_t = a\right].$$

In real-world problems, in order to achieve better generalization, instead of learning a value for each state-action pair, it is more common to learn an approximate value function: $Q^{\pi}(s, a; \theta) \approx Q^{\pi}(s, a)$. A standard approach is to learn a feature-based state representation $\phi(s)$ instead of using the raw state s [27, 58, 46]. Due to the popularity of Deep Reinforcement learning, it has been a trend to deploy neural networks to automatically extract high-level features [74, 58]. However, running most deep RL models is computationally very heavy. In contrast, static feature representations are usually light-weight and simple to deploy. Several studies also reveal that a carefully designed static feature representation can achieve performance as good as the most sophisticated deep RL models, even in the most challenging domains [46].

With some transformation, the crowdsourcing problem we aim to tackle in this chapter can perfectly fit into the above RL formalization. To be more specific, the data requester is the agent and it interacts with workers (i.e. the environment); scaling factors are actions; the utility of the data requester after paying workers (see Equation 5.4) is the reward r_t; how workers respond to different incentives and potentially change their internal states thereafter forms the transition kernel, which is unobservable; what scaling factor to be picked at each step t given workers' labeling constructs the policy, which needs to be learned. Since the real accuracy cannot be observed, we use the estimated accuracy $\tilde{A}$ calculated by Equation 5.12 instead to approximate the reward signal as

$$r_t \approx F(\tilde{A}^t) - \eta \sum_{i=1}^{N} P_i^t. \tag{5.13}$$

Recall that the data requester's implicit utility at time t only depends on the aggregate PoBC averaged across the whole worker body. Such observation already points out to a representation design with good generalization, namely $\phi(s_t) = \sum_{i=1}^{N} \mathbb{P}_i^t / N$. Further recall that, when deciding the current scaling factor a_t, the data requester does not observe the latest workers' PoBCs and thus cannot directly estimate the current $\phi(s_t)$. Due to this one-step delay, we have to build our state representation using the previous observation. Since most workers would only change their internal states after receiving a new incentive, there exists some imperfect mapping function $\phi(s_t) \approx f(\phi(s_{t-1}), a_{t-1})$. Utilizing this implicit function, we introduce the augmented state representation in RIL as

$$\hat{s}_t = \langle \phi(\hat{s}_{t-1}), a_{t-1} \rangle.$$

Since neither r_t nor $\hat{s}_t$ can be perfectly accurate, it would not be a surprise to observe some noise that cannot be directly learned in our Q-function. As for most crowdsourcing problems the number of tasks M is large, we leverage the central limit theorem to justify our modeling of the noise using a Gaussian process. To be more specific, we calculate the temporal difference (TD) error as

$$r_t = Q^{\pi}(\hat{s}_t, a_t) - \gamma \mathbb{E}_{\pi} Q^{\pi}(\hat{s}_{t+1}, a_{t+1}) + \epsilon_t \tag{5.14}$$

where the noise ϵ_t follows a Gaussian process $\mathcal{N}(\hat{s}_t, \hat{s}_{t+1})$, and $\pi = \mathbb{P}(a|\hat{s})$ denotes the current policy. Doing so, we gain two benefits. First, it greatly simplifies the derivation of the update equation for the Q-function. Secondly, as shown in our empirical results later, it is robust against different worker models. Besides, following [25], we approximate Q-function as

$$Q^{\pi}(\hat{s}_{t+1}, a_{t+1}) = \mathbb{E}_{\pi} Q^{\pi}(\hat{s}_{t+1}, a_{t+1}) + \epsilon_{\pi}$$

where ϵ_{π} also follows a Gaussian process.

Under the Gaussian process approximation, all the observed rewards and the corresponding Q values up to the current step t form a equation set, and it can be written as

$$\boldsymbol{r} = \boldsymbol{H}\boldsymbol{Q} + \boldsymbol{N} \tag{5.15}$$

where $\boldsymbol{r}$, $\boldsymbol{Q}$ and $\boldsymbol{N}$ denote the collection of rewards, Q values, and residuals. Following Gaussian process's assumption for residuals, $\boldsymbol{N} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\sigma}^2)$, where $\boldsymbol{\sigma}^2 = \mathrm{diag}(\sigma^2, \ldots, \sigma^2)$. The matrix $\boldsymbol{H}$ satisfies $\boldsymbol{H}(k,k) = 1$ and $\boldsymbol{H}(k, k+1) = -\gamma$ for $k = 1, \ldots, t$. Then, by using the online Gaussian process regression algorithm [18], we effectively learn the Q-function as

$$Q(\hat{s}, a) = \boldsymbol{k}(\hat{s}, a)^{\mathrm{T}} (\boldsymbol{K} + \boldsymbol{\sigma}^2)^{-1} \boldsymbol{H}^{-1} \boldsymbol{r} \tag{5.16}$$

where the vector $\boldsymbol{k}(\hat{s}, a) = [k((\hat{s}, a), (\hat{s}_1, a_1)), \ldots, k((\hat{s}, a), (s_t, a_t))]^{\mathrm{T}}$ and the matrix $\boldsymbol{K} = [\boldsymbol{k}(\hat{s}_1, a_1), \ldots, \boldsymbol{k}(\hat{s}_t, a_t)]$. Here, we use $k(\cdot, \cdot)$ to denote the Gaussian kernel. Finally, we employ the classic ϵ-greedy method to decide a_t based on the learned Q-function. To summarize, we provide a formal description about RIL in Algorithm 6. Note that, when updating $\boldsymbol{K}$, $\boldsymbol{H}$ and $\boldsymbol{r}$ in Line 6, we employ the sparse approximation to discard some data so that the size of these matrices does not increase infinitely. The details of this technique can be found in [25]. Note that, for the hyper-parameters in the Gaussian kernel, if we can have a good estimation in advance, we can set them accordingly; otherwise, we tend to set them larger for an approximately uniform prior.

Algorithm 6: Reinforcement Incentive Learning (RIL)

begin
  for *each episode* **do**
    for *each step in the episode* **do**
      Decide the scaling factor as (ϵ-greedy method)

$$a_t = \begin{cases} \arg\max_{a \in \mathcal{A}} Q(\hat{s}_t, a) & \text{Probability } 1-\epsilon \\ \text{Random } a \in \mathcal{A} & \text{Probability } \epsilon \end{cases}$$

      Assign tasks and collect labels from the workers
      Run Bayesian inference to get $\hat{s}_{t+1}$ and r_t
      Use $(\hat{s}_t, a_t, r_t)$ to update $\boldsymbol{K}$, $\boldsymbol{H}$ and $\boldsymbol{r}$ in Equation 5.16

5.3 Convergence and Incentive Results

In this section, we present the theoretic analysis of our incentive mechanism[4]. Our main results are as follows:

Theorem 5.1 (One-Step IC) *At any time step t, when $M \gg 1$, $(2\mathbb{P}_H)^{2(N-1)} \geq M$ and $a_t > \frac{c_H}{\mathbb{P}_H - 0.5}$, reporting truthfully and exerting high efforts is the utility-maximizing strategy for any worker i if other workers all follow this strategy.*

Theorem 5.2 (Long-Term IC) *Suppose the conditions in Theorem 5.1 are satisfied and the learned Q-function approaches the real $Q^{\pi}(\hat{s}, a)$. When*

$$\eta M(N-1)\mathbb{P}_H \cdot G_{\mathcal{A}} > \frac{F(1) - F(1-\psi)}{1-\gamma} \tag{5.17}$$

$$\psi = (\tau_1 \tau_2^{-1} + \tau_1^{-1}\tau_2)[4\mathbb{P}_H(1-\mathbb{P}_H)]^{\frac{N-1}{2}} \tag{5.18}$$

[4]Currently, our theoretical analysis is for the case that $m_i^t = M$. $m_i^t < M$ requires to replace the binomial distribution with the trinomial distribution when analyzing a key function involved in the proof in the supplementary file. The main idea of proof is roughly the same, and we will extend our proof in the future work.

always reporting truthfully and exerting high efforts is the utility-maximizing strategy for any worker i in the long-term if other workers all follow this strategy. Here, $G_{\mathcal{A}} = \min_{a,b\in\mathcal{A},a\neq b} |a-b|$ denotes the minimal gap between two available values of the scaling factor.

In the following two subsections, we outline the proofs of these two theorems. In our proof, if we omit the superscript t in an equation, we mean that this equation holds for all time steps. We also employ the convention that $\bar{\mathbb{P}} = 1-\mathbb{P}$, $\hat{\mathbb{P}} = \max\{\mathbb{P}, \bar{\mathbb{P}}\}$ and $\mathbb{P}_0 = \tau_1$.

5.3.1 Proof for One-Step IC

Our idea to prove the one-step IC is to derive the conditions from the conclusion. To be more specific, we focus on one individual worker at first and derive the conditions needed to ensure exerting high efforts and reporting truthfully to be the utility-maximizing strategy. Then, we move to our Bayesian inference algorithm and study how to ensure these conditions in our mechanism.

Firstly, for workers' utility defined in Equation 5.3, we can have

Lemma 5.1 *For worker i, when $M \gg 1$ and $a_t > \frac{c_H}{\mathbb{P}_H - 0.5}$, if $\tilde{\mathbb{P}}_i^t \approx \mathbb{P}_i^t$, reporting truthfully ($\mathsf{rpt}_i^t = 1$) and exerting high efforts ($\mathsf{eft}_i^t = 1$) is the utility-maximizing strategy.*

Proof: *When $M \gg 1$, we can have $\sum_j \mathsf{sc}_i(j) \approx M \cdot \tilde{\mathbb{P}}_i$. Thus, the utility of worker i can be computed as*

$$u_i^t \approx M \cdot a_t \cdot (\tilde{\mathbb{P}}_i - 0.5) + M \cdot b - M \cdot c_H \cdot \mathsf{eft}_i^t. \tag{5.19}$$

Further considering Equation 5.2 and $\mathbb{P}_L = 0.5$, if $\tilde{\mathbb{P}}_i^t \approx \mathbb{P}_i^t$, we can compute worker i's utility as

$$u_i^t \approx M \cdot [a_t(2 \cdot \mathsf{rpt}_i^t - 1)(\mathbb{P}_H - 0.5) - c_H] \cdot \mathsf{eft}_i^t + M \cdot b. \tag{5.20}$$

Thereby, if $a_t > \frac{c_H}{\mathbb{P}_H - 0.5}$, $\mathsf{rpt}_i^t = 1$ and $\mathsf{eft}_i^t = 1$ maximize u_i^t, which concludes Lemma 5.1.

In this case, we can conclude Theorem 5.1 by proving the convergence of our Bayesian inference algorithm, i.e. proving $\tilde{\mathbb{P}}_i^t \approx \mathbb{P}_i^t$. $\tilde{\mathbb{P}}_i^t$ is computed according to Equation 5.9, and if most of the samples in our Bayesian inference are correct, namely $\tilde{\mathcal{L}}^{(s)}(j) \equiv \mathcal{L}(j)$, we can prove $\tilde{\mathbb{P}}_i^t \approx \mathbb{P}_i^t$ by leveraging the law of large numbers. This observation motivates us to bound $|\tilde{\mathbb{P}}_i^t - \mathbb{P}_i^t|$ by calculating the upper bound of the ratio of wrong labels in the samples. Thereby, we prove Theorem 5.1 with the following two steps:

Step 1: In this step, we aim to derive the upper bound of the ratio of wrong samples. To achieve this objective , we introduce n and m to denote the number of tasks of which the true label sample in Equation 5.9 is correct ($\tilde{\mathcal{L}}^{(s)}(j) = \mathcal{L}(j)$) and wrong ($\tilde{\mathcal{L}}^{(s)}(j) \neq \mathcal{L}(j)$), respectively. Then, we are able to prove the following lemma:

Lemma 5.2 *When $M \gg 1$,*

$$\mathbb{E}[m/M] \lesssim (1+e^{\delta})^{-1}(\varepsilon + e^{\delta})(1+\varepsilon)^{M-1} \tag{5.21}$$

$$\mathbb{E}[m/M]^2 \lesssim (1+e^{\delta})^{-1}(\varepsilon^2 + e^{\delta})(1+\varepsilon)^{M-2} \tag{5.22}$$

where $\varepsilon^{-1} = \prod_{i=0}^{N}(2\hat{\mathbb{P}}_i)^2$, $\delta = O[\Delta \cdot \log(M)]$ and

$$\Delta = \sum_{i=1}^{N}[1(\mathbb{P}_i < 0.5) - 1(\mathbb{P}_i > 0.5)].$$

We defer the detailed proof to Appendix B.2. Our main idea is to introduce a set of counts for the collected labels at first. More specifically, among the n tasks of which the posterior true label is correct, x_i and y_i denote the number of tasks of which worker i's label is correct and wrong, respectively. Among the remaining m tasks, w_i and z_i denote the number of tasks of which worker i's label is correct and wrong, respectively. Then, we calculate the approximation of $\mathbb{P}(m)$ based on the conditional probabilities $\mathbb{P}(x_i, y_i, w_i, z_i|m)$ and $\mathbb{P}(\mathcal{L}|\boldsymbol{L})$. The upper bounds of $\mathbb{E}[m/M]$ and $\mathbb{E}[m/M]^2$ can be obtained by calculating the upper bounds of $\sum_m m\mathbb{P}(m)$ and $\sum_m m^2\mathbb{P}(m)$.

Step 2: In this step, we derive the upper bound of $|\tilde{\mathbb{P}}_i - \mathbb{P}_i|$ under the conditions of Theorem 5.1. Following the notations in Step 1, when $M \gg 1$, in Equation 5.9, we have $\tilde{\mathbb{P}}_i \approx \mathbb{E}_{\mathcal{L}}(x_i + z_i)/M$, where $\mathbb{E}_{\mathcal{L}}$ denotes the expectation with respect to $\mathbb{P}(\mathcal{L}|\boldsymbol{L})$. Meanwhile, according to the law of large numbers, $\mathbb{P}_i \approx (x_i + w_i)/M$. Thus, we have

$$|\tilde{\mathbb{P}}_i - \mathbb{P}_i| \approx \mathbb{E}_{\mathcal{L}}|w_i - z_i|/M \leq \mathbb{E}_{\mathcal{L}}\left[m/M\right]. \tag{5.23}$$

If workers, except for worker i, all report truthfully and exert high efforts, then $\Delta \leq -1$ in Theorem 5.2 because we require $N \geq 3$ in Section 5.1. Thus, $e^{\delta} \approx 0$. Since $2\hat{\mathbb{P}}_i \geq 1$, we have $\varepsilon^{-1} \geq (2\mathbb{P}_H)^{2(N-1)}$. Hence, $\varepsilon \leq M^{-1}$ when $(2\mathbb{P}_H)^{2(N-1)} \geq M$. Taking the above analysis into consideration, Equations 5.21 and 5.22 can be calculated as

$$\mathbb{E}\left[\frac{m}{M}\right] \lesssim \frac{C_1}{M \cdot C_2} \ , \ \mathbb{E}\left[\frac{m}{M}\right]^2 \lesssim \frac{C_1}{M^2 \cdot C_2^2} \tag{5.24}$$

where $C_1 = (1+M^{-1})^M \leq e$ and $C_2 = 1 + M^{-1} \geq 1$. Then, $m/M \approx 0$ because $\mathbb{E}[m/M] \approx 0$ and $\mathrm{Var}[m/M] = \mathbb{E}[m/M]^2 - (\mathbb{E}[m/M])^2 \approx 0$. In this case, $\tilde{\mathbb{P}}_i \approx \mathbb{P}_i$, and thus we can conclude Theorem 5.1.

5.3.2 Proof for Long-Term IC

Due to the one-step IC in our mechanism, we know that, to get higher long-term payments, worker i must mislead our RL algorithm into at least increasing the scaling factor from a to any $a' > a$ at a certain state $\hat{s}$. Actually, our RL algorithm will only increase the scaling factor when the state-action value function satisfies $Q^{\pi}(\hat{s}, a) \leq Q^{\pi}(\hat{s}, a')$. Equation 5.13 tells us that our objective function consists of the utility obtained from the collected labels ($F(\tilde{A}^t)$) and the utility lost in the payment ($\eta \sum_{i=1}^{N} P_i^t$). Once we increase the scaling factor, we at least need to increase the payments for the other $N-1$ workers by $M(N-1)\mathbb{P}_H \cdot G_{\mathcal{A}}$, which corresponds to the left-hand side of Equation 5.17.

On the other hand, for the obtained utility, we have the following lemma.

Lemma 5.3 *At any time step t, if all workers except for worker i report truthfully and exert high efforts,*

$$F(\tilde{A}^t) \leq F(1) \ , \ F(\tilde{A}^t) \geq F(1-\psi)$$

where ψ is defined in Equation 5.18.

We again defer the detailed proof to Appendix B.4. Our main idea is to derive the bounds of the estimated label accuracy $\tilde{A}$ by analyzing the distribution of the log-ratio $\tilde{\sigma}_j$ calculated in Equation 5.11. Our motivation comes form Equation 5.12 which ensures, $\tilde{A} \approx 1 - \mathbb{E}[1/(1 + e^{|\tilde{\sigma}_j|})]$. From Lemma 5.3, we can know that, even in the long-term, worker i can at most increase our value by $(1-\gamma)^{-1}[F(1) - F(1-\psi)]$, which corresponds to the right-hand side of Equation 5.17.

Thereby, if Equation 5.17 is satisfied, worker i will be unable to make up our value loss increment in the payments, and our RL algorithm will reject the hypothesis to increase the scaling factor. In this case, the only utility-maximizing strategy for worker i is to report truthfully and exert high efforts in all time steps, which concludes Theorem 5.2.

5.4 Empirical Experiments

In our reward mechanism, Bayesian inference extracts important and useful information from all collected labels. On the other hand, RIL empowers us with the adaptivity to different types of worker models. In this section, we demonstrate these benefits of our reward mechanism via empirical studies.

5.4.1 Performance Analysis of Bayesian Inference

In this subsection, we focus on our Bayesian inference algorithm by fixing the scaling factor $a_t = 1$ and setting the number of tasks at each step $M = 100$, the number of workers $N = 10$, the probability of being correct when exerting high efforts $\mathbb{P}_H = 0.8$, the base salary $b = 0$ and the number of tasks assigned to each worker at each step $m_i^t = M(N-1)/N$. We use the average payment to worker i as a proxy, as it is proportional to scores computed by Bayesian inference. We use DG13, the state-of-the-art peer prediction mechanism for binary labels [15], as the benchmark to conduct our comparison. To set up the experiments, we generate task j's true label $\mathcal{L}(j)$ following its distribution $\boldsymbol{\tau}$ and worker i's label for task j based on i's PoBC $\mathbb{P}_i$ and $\mathcal{L}(j)$. For each data point, we run experiments for 1000 rounds and report the mean.

In Figure 5.2a, we let all workers excluding i report truthfully and exert high efforts (i.e. $\mathbb{P}_{-i} = \mathbb{P}_H$), and increase τ_1 from 0.05 to 0.95. In Figure 5.2b, we let $\tau_1 = 0.5$, and increase other workers' PoBCs $\mathbb{P}_{-i}$ from 0.6 to 0.95. As both figures reveal, in our mechanism, the payment for worker i almost only depends on his/her own strategy.

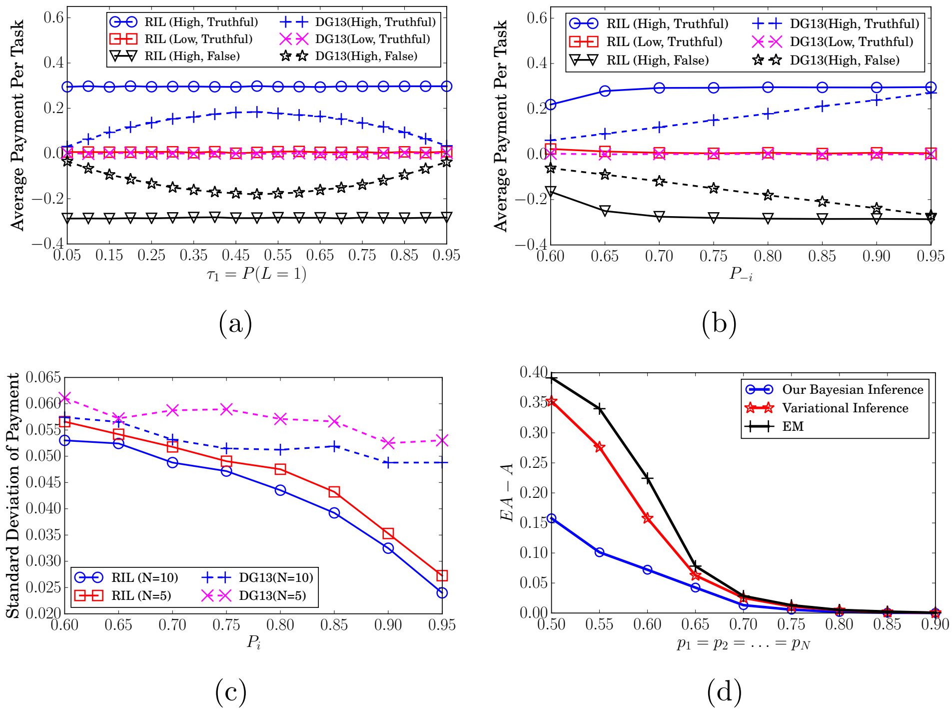


Figure 5.2: Empirical analysis on our Bayesian inference algorithm. (a) Average payment per task given true label's distribution. (b) Average payment per task given PoBCs of workers excluding i. (c) The standard deviation of the payment given worker i's PoBC. (d) The inference bias of the label accuracy.

By contrast, in DG13, the payments severely affected by the distribution of true labels and the strategies of other workers. It is unfair for a worker to get low rewards because of other workers' low efforts or the unbalanced distribution of true labels. In other words, our Bayesian inference is more robust to different environment. Furthermore in Figure 5.2c, we present the standard deviation of the payment to worker i. We let $\tau_1 = 0.5$, $\mathbb{P}_{-i} = \mathbb{P}_H$ and increase $\mathbb{P}_i$ from 0.6 to 0.95. As shown in the figure, our method manages to achieve a noticeably smaller standard deviation compared to DG13. In summary, compared with traditional peer prediction mechanisms that only make use of a small set of workers' labels to decide a payment, the usage of our Bayesian inference algorithm improves its robustness and decreases the variance, because of its ability to fully exploit all the collected labels.

Furthermore, in Figure 5.2d, we compare our Bayesian inference algorithm with two popular inference algorithms in crowdsourcing, that is, the EM estimator [66] and the variational inference estimator [48]. Here, we set workers' PoBC $\mathbb{P}_i$ to be equal and increase the value of $\mathbb{P}_i$ from 0.5 to 0.9. The other settings are the same as Figure 5.2b. From the figure, we can find that, when the quality of labels is very low, the inference bias of the EM and variational inference estimators on the label accuracy can be larger than 0.3 while the range of the label accuracy is only $[0.5, 1.0]$. This observation shows that these two estimators become over-optimistic for low-quality labels, which will be disastrous for our RIL algorithm. Thus, we develop our Bayesian inference algorithm to calibrate the estimates of label accuracy for training our RIL algorithm.

5.4.2 Empirical Analysis on RIL

In this subsection, we focus on investigating whether RIL proposed in Section 5.2.3 consistently manages to learn a good policy to maximize the data requester's cumulative utility $R = \sum_t r_t$. For all the experiments in this subsection, we set $M = 100$, $N = 10$, $\mathbb{P}_H = 0.8$, $b = 0$, $c_H = 0.02$, the available value set of the scaling factor $\mathcal{A} = \{0.1, 1.0, 5.0, 10\}$, the exploration rate $\epsilon = 0.2$ for RIL, $F(A) = A^{10}$, $\eta = 0.1$ for the utility function (Equation 5.4) and the number of time steps for an episode as 28. To reduce the influence of outliers, we report the average over 5 trials. To demonstrate our algorithm's general applicability, we test it under three different worker models, with each capturing a different rationality level. The formal description of the three models is as follows:

- **Rational** workers alway act to maximize their own utilities. Since our incentive mechanism theoretically ensures that exerting high effort is the utility-maximizing strategy for all workers (proved in Section 5.3), it is safe to assume workers always do so as long as the payment is high enough to cover the cost.

- **Quantal Response (QR)** workers [55] exert high efforts with the probability

$$\mathsf{eft}_i^t = \frac{\exp(\lambda \cdot u_{iH}^t)}{\exp(\lambda \cdot u_{iH}^t) + \exp(\lambda \cdot u_{iL}^t)}$$

  where u_{iH}^t and u_{iL}^t denote worker i's expected utility after exerting high or low efforts respectively at time t. λ describe workers' rationality level and we set $\lambda = 3$ in our experiments.

- **Multiplicative Weight Update (MWU)** workers [11] update their probabilities of exerting high efforts at every time step t after receiving the payment as the following equation

$$\mathsf{eft}_i^{t+1} = \frac{\mathsf{eft}_i^t(1+\bar{u}_{\cdot H})}{\mathsf{eft}_i^t(\bar{u}_{\cdot H}-\bar{u}_{\cdot L})+\bar{u}_{\cdot L}+1}$$

where $\bar{u}_{\cdot H}$ and $\bar{u}_{\cdot L}$ denote the average utilities received if exerting high efforts or low efforts at time t respectively. We initialize eft_i^0 as 0.2 in our experiments.

Since we have shown the advantage of using Bayesian inference in Section 6.1, for the sake of fair comparison, we use our payment rule with manually adjusted scaling factor as the benchmark. In this case, how tasks are assigned does not affect the empirical analysis we wish to conduct and thus we let every worker to be assigned the whole task set at each time step t (i.e. $\forall i, m_i^t = M$).

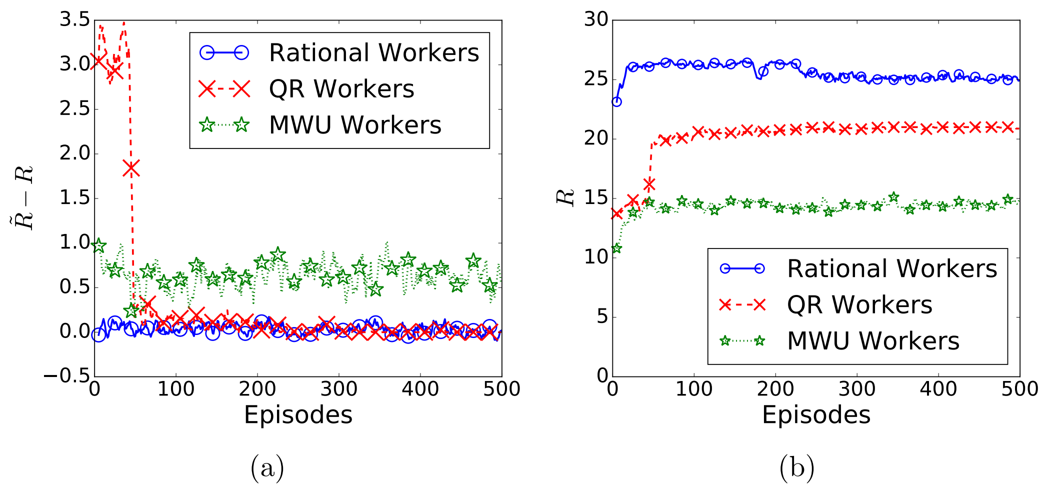


Figure 5.3: Empirical analysis on our RL algorithm. (a) The gap between the estimation of the data requester's cumulative rewards and the real one, smoothed over 5 episodes. (b) The learning curve of our mechanism smoothed over 5 episodes.

Our first set of experiments focus on the estimation bias of the data requester's cumulative utility R. Since the data requester's utility is used as the reward in RIL, it would not be a surprise that the reliability of the estimation of the reward plays a crucial role in determining the well-being of the whole mechanism. As Figure 5.3a shows, the estimated value only deviates from the real one in a very small magnitude after a few episodes of learning, regardless of which worker model the experiments run on. The next set of experiments is about how quickly RIL learns. As Figure 5.3b shows, under all three worker models, RIL manages to pick up and stick to a promising policy in less than 100 episodes. This observation also demonstrates the robustness of RIL under different deploying environment.

Table 5.2: Performance comparison between RIL and traditional mechanisms on three worker models. Requester's cumulative utility normalized over the number of tasks. Standard deviation reported in parenthesis.

Method	Rational	QR	MWU
Fixed Optimal	27.584 (.253)	21.004 (.012)	11.723 (.514)
Heuristic Optimal	27.643 (.174)	21.006 (.001)	12.304 (.515)
Adaptive Optimal	27.618 (.109)	21.017 (.004)	18.475 (.382)
RIL	27.184 (.336)	21.016 (.018)	15.726 (.416)

Lastly, we take the learned policy after 500 episodes with exploration rate turned off (i.e. $\epsilon = 0$) and compares it with two benchmarks constructed by ourself (see Table 5.2). To create the first one, Fixed Optimal, we try all 4 possible fixed value for the scaling factor and report the highest cumulative reward realized by either of them. Note most traditional peer prediction mechanisms assume a fixed scaling factor and thus Fixed Optimal represents the best performance possibly achieved by them. To create the second one, Heuristic Optimal, we divide the value region of $\tilde{A}^t$ into five regions: $[0, 0.6)$, $[0.6, 0.7)$, $[0.7, 0.8)$, $[0.8, 0.9)$ and $[0.9, 1.0]$. For each region, we select a fixed value for the scaling factor a_t. We traverse all $4^5 = 1024$ possible combinations to decide the optimal heuristic strategy. To create the third one, Adaptive Optimal, we change the scaling factor every 4 steps and report the highest cumulative reward via traversing all $4^7 = 16384$ possible configurations. This benchmark is infeasible to be reproduced in real-world practice, once the number of times steps becomes large. Yet it is very close to the global optimal in the sequential setting. As Table 5.2 demonstrates, the two benchmarks plus RIL all achieve a similar performance tested for rational and QR workers. This is because these two kinds of workers have a fixed pattern in response to incentives and thus the optimal policy would be a fixed scaling factor throughout the whole episode. On contrast, MWU workers' learn utility-maximizing strategies gradually, and the learning process is affected by the incentives. Under this model, compared with Fixed Optimal, RIL increases the cumulative reward from 11.7 to 15.6, which is a significant improvement considering the unreachable real optimal is only around 18.5. Up to this point, with three sets of experiments, we demonstrate the competitiveness of RIL and its robustness under different environment, which ensures the ability of our mechanism to adapt to different types of workers in practical usage.

5.5 Summary

In this chapter, we build a novel inference aided reinforcement learning reward mechanism for sequentially acquiring data from crowdsourcing. At each time step, our mechanism uses the Bayesian inference algorithm to learn workers' probability of being correct, and we issue payments to workers that are proportional to the estimated accuracy. When interacting with workers, our mechanism learns the optimal policy to adjust the scaling factor of the payments via our RIL algorithm. We theoretically prove that our mechanism is incentive compatible—i.e. workers can only get the maximal utility by exerting high efforts and reporting truthfully. As a by product, we have also proved the convergence of our Bayesian inference method. We empirically show that our Bayesian inference algorithm can help improve the robustness and lower the variance of payments, which are favorable properties in practice. Meanwhile, our mechanism performs consistently well with different worker models.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This chapter concludes the completed research so far, towards constructing a novel microtask crowdsourcing system, which is more intelligent and more practical in managing the large crowd of workers than existing mechanisms. By considering the unique features of worker recruitment in microtask crowdsourcing, the optimal posted-price mechanism has been proposed in Chapter 3. To reduce to the number of redundant labels needed for high label accuracy, a novel active task assignment mechanism has been proposed in Chapter 4. To fully exploit the information contained in the collected labels, Chapter 5 proposes an inference aided reinforcement learning reward mechanism. The proposed reward mechanism can improve the robustness and, at the same time, lower the variance of rewards. Moreover, in all the above mechanisms, we use machine learning techniques to learn workers' models rather than assuming workers to follow a fixed strategy as the traditional incentive mechanism studies. This design avoids impractical assumption about workers and thus makes our mechanisms more suitable for practical use.

In more detail, in Chapter 3, the optimal posted-price mechanism has been proposed to fully exploit the unique features of microtask crowdsourcing. It uses a multi-armed bandit algorithm to gradually learn the worker model and optimally decide the price. With the same budget, it can recruit more workers than existing mechanisms. At the same time, it requires fewer inputs, which is favorable in real-world applications. To demonstrate the optimality of the proposed mechanism, the classic

Lai-Robbins regret lower bound is derived. This lower bound applies to any possible posted-price mechanism and denotes the best performance that can be achieved. Then, the regret of our mechanism is proven to match this bound. Furthermore, it is proven that, even if the price range is infinite, the regret of the proposed mechanism is not affected. The proposed mechanism can thus be used without imputing a price range as the prior knowledge about workers. In addition to theoretical analysis, the empirical results on various worker models and the real price data collected from MTurk also verify the advantages of the proposed mechanism.

In Chapter 4, an active task assignment mechanism has been proposed to reduce the number of redundant labels needed for achieving high label accuracy. It uses variational inference to learn the worker models in an online fashion and decides the task assignment by using active learning strategies. However, combining these two techniques in microtask crowdsourcing will cause problems in handling the uncertainties of learning and computation efficiency. In addition, traditional variational inference algorithms developed for microtask crowdsourcing cannot incorporate different worker models. To address these problems, the proposed mechanism keeps the prediction optimistic, uses the uncertainty measurement of inference to modulate the scope of task assignment, and predicts the future via an approximation algorithm of variational inference. In addition, a unified formulation of different worker models is developed so that the proposed mechanism can incorporate any of the worker models. Extensive experiments are conducted based on four popular worker models and four MTurk datasets. The empirical results show that the proposed mechanism not only can achieve the highest label accuracy but also has the highest computation efficiency among all existing prediction-based task assignment approaches.

In Chapter 5, a novel inference aided reinforcement learning reward mechanism has been proposed to incentivize workers to generate high-quality labels. It uses the Bayesian inference algorithm to learn workers' probability of being correct, and the rewards are proportional to these estimated probabilities. When interacting with workers, it learns the optimal policy to adjust the scaling factor of the rewards via the reinforcement incentive learning algorithm. The proposed reward mechanism is proven that, exerting high-efforts and truthfully reporting is the utility-maximizing strategy for all workers both at each step and in the long-term interaction. This property ensures rational workers are correctly incentivized. Then, to demonstrate the advantages of the proposed reward mechanism, extensive experiments have been conducted. The empirical results show that employing the Bayesian inference algorithm can help improve the robustness and lower the variance of rewards, which are favorable properties in practice. Meanwhile, the reinforcement incentive learning algorithm can adapt the mechanism to different types of workers, which ensures the proposed mechanism to perform consistently well even with bounded rational and self-learning workers.

In all, the goal of our research in Chapters 3-5 is two-fold. First, we improve the existing learning-based mechanisms by analyzing the unique features and problems of microtask crowdsourcing. For example, our active task assignment mechanism boosts the label accuracy by analyzing the inference uncertainties of microtask crowdsourcing. Second, since no learning-based reward mechanisms exist now, we design a new reward mechanism by developing Bayesian inference and reinforcement learning algorithm specifically for the mechanism design of microtask crowdsourcing. Our mechanism not only achieves the same theoretical objective as traditional reward mechanisms, but also empirically outperforms them. Furthermore, when the assumption of these traditional reward mechanism is violated, our mechanism can still perform consistently well. Putting the proposed mechanisms together, we wish to build a novel microtask crowdsourcing system which is more intelligent and more practical in managing the large crowd of workers.

6.2 Future Work

Based on the current work, the future research is planned in four aspects.

6.2.1 Extension to Our Pricing Mechanism

In the optimal posted-price mechanism proposed in Chapter 3, we still need to try all the possible prices one-by-one. This simple way of searching is quite inefficient. Thus, we expect to adaptively change the searching step to enhance efficiency in the future. Besides, our mechanism can decide the price for the next worker only after the last worker has fed back his/her decision about the offered price. In other words, our current mechanism cannot handle the delay of workers' feedback. According to Chapelle and Li's empirical studies [10], Thompson sampling, as an alternative method to the bandit algorithm that is studied in Chapter 3, can alleviate the influence of delayed feedback by randomizing over actions. Thus, in the future, we also expect to redesign our posted-price mechanism based on Thompson sampling. However, the theoretical analysis on Thompson sampling is very challenging and requires significant further study. Furthermore, our posted-price mechanism assumes workers' decisions on the offered prices follow a fixed distribution. Although this assumption can incorporate the case where workers are bounded rational, it makes our mechanism unable to correctly adjust the price when workers are time-varying. According to Combes and Proutiere's studies [13], we can overcome this non-stationary problem by adding a sliding window to the bandit algorithm. However, how this sliding window will affect the performance of our mechanism requires further study.

6.2.2 Extension to Our Task Assignment Mechanism

In the active task assignment mechanism proposed in Chapter 4, we learn workers' models based on variational inference. Our learning algorithm only uses the collected label as inputs. However, in practice, we usually have contextual information about the targeted tasks [47, 40, 19]. For example, if our task is to label an image dataset (e.g. ImageNet), we can use convolutional neural networks to handle these images, which can help us to evaluate workers' contributions. Thus, one of our future works is to study how to merge the contextual information of tasks and the collected labels from workers in task assignment. In addition, for a small part of tasks, we may be able to obtain the real true labels by, for example, employing a small number of experts [29, 62, 84]. In this case, the obtained real true labels can be used to calibrate the learn models about workers. However, how to incorporate the calibration into our online learning process still requires further study. Moreover, our mechanism now can only handle the push market where there is a fixed worker pool and we can call any worker to label any task at any time. The most common market in practice is the pull market where workers can come and leave at any time. It is thus of great importance to extend our mechanism to the pull market in the future. A possible approach is to use the output of our mechanism to adjust the waiting time of workers. If the worker is of low quality, we can wait for a quite long time to assign him/her the next task.

6.2.3 Extension to Our Reward Mechanism

In the reward mechanism proposed in Chapter 5, we develop a Bayesian inference algorithm to learn workers' models. One advantage of our inference algorithm is that we can theoretically prove the convergence to the true worker models, which is the basis of proving the incentive compatibility. However, similar to the variational inference algorithm used in our task assignment mechanism, our inference algorithm cannot incorporate the contextual information of tasks and the real true labels of a small part of tasks. It will be very challenging to develop an algorithm that can maintain the theoretical guarantee for the convergence to the real worker models and meanwhile incorporate these additional information. This is because machine learning studies often focus on improving the empirical performance while mechanism design studies need to at least ensure the rational workers are correctly incentivized. Besides, in practice, the states of workers can be far more complex than exerting high or low efforts and truthfully or falsely reporting the labels. For example, some workers are active on getting tasks from the requester while the others may be not so active. In the future, it will be an important direction for us to explore the question of how to improve our mechanism with more complex state representations.

6.2.4 Integration of Different Mechanisms

The three mechanisms proposed in Chapters 3-5, each covers one step of microtask crowdsourcing. In the future, we wish to integrate them to further enhance the effects of incentives. For example, in the reinforcement incentive learning algorithm of our reward mechanism, we will handle 100 tasks in each step. How to assign the 100 tasks to the participating workers can be handled by our task assignment mechanism. By doing so, we can let those workers who provide low-quality labels not only get less rewards on each task but also be assigned less tasks. Another example is to integrate the posted-price mechanism and the reward mechanism. The posted-price mechanism only decides the base payment in our final rewards (b in Equation 5.5), and our reward mechanism only adjusts the scaling factor of the rewards (a in Equation 5.5). In the future, we can develop a reinforcement learning algorithm to adjust the base payment and the scaling factor in the meanwhile. By doing so, the price considers workers' response to rewards rather than assuming a fixed distribution for workers' decisions. Also, adjusting the base payment can help us to further enlarge the incentive differences between high- and low-quality workers.

Appendix A

Authors Publications

- **Zehong Hu**, Meng Sha, Moath Jarrah, Jie Zhang and Hui Xi. "Efficient Computation of Emergent Equilibrium in Agent-Based Simulation." In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI)*, 2016.
- **Zehong Hu**. "A Novel Approach to Evaluate Robustness of Incentive Mechanism Against Bounded Rationality", In *Proceedings of International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2016. (Doctor Consortium)
- Donghun Kang, Zhenchao Bing, Wen Song, **Zehong Hu**, Shuo Chen, Jie Zhang and Hui Xi. "Automatic Construction of Agent-based Simulation Using Business Process Diagrams and Ontology-based Models", In *Proceedings of International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2017. (Demo Paper)
- **Zehong Hu** and Jie Zhang. "Optimal Posted-Price Mechanism in Microtask Crowdsourcing", In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, 2017.
- **Zehong Hu** and Jie Zhang. "A Novel Strategy for Active Task Assignment in Crowd Labeling", In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, 2018.
- **Zehong Hu**, Yang Liu, Yitao Liang and Jie Zhang. "A Reinforcement Learning Framework for Eliciting High Quality Information", In *Neural Information*

Processing Systems (NIPS) workshop on Machine Learning in the Presence of Strategic Behavior, 2017.

- **Zehong Hu** and Jie Zhang. "Towards General Robustness Evaluation of Incentive Mechanism Against Bounded Rationality", *IEEE Transactions on Computational Social Systems*, Vol.5, No.3, pp. 698-712, 2018.

- **Zehong Hu**, Yitao Liang, Jie Zhang, Zhao Li and Yang Liu. "Inference Aided Reinforcement Learning for Incentive Mechanism Design in Crowdsourcing", In *Proceedings of Neural Information Processing Systems (NIPS)*, 2018.

- **Zehong Hu**, Jie Zhang and Zhao Li. "General Robustness Evaluation of Incentive Mechanism Against Bounded Rationality Using Continuum-Armed Bandits", In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

Appendix B

B.1 Basic Lemmas

We firstly present some lemmas for our proof later.

Lemma B.1 *If* $x \sim \text{Bin}(n, p)$, $\mathbb{E}t^x = (1 - p + tp)^n$ *holds for any* $t > 0$, *where* $\text{Bin}(\cdot)$ *is the binomial distribution.*

Proof:

$$t^x = e^{x \log t} = m_x(\log t) = \left(1 - p + pe^{\log t}\right)^n \tag{7.1}$$

where $m_x(\cdot)$ *denotes the moment generating function.*

Lemma B.2 *For given* $n, m \geq 0$, *if* $0 \leq p \leq 1$, *we can have*

$$\sum_{x=0}^{n} \sum_{w=0}^{m} C_n^x C_m^w p^{x+w} (1-p)^{y+z} \times B(x+z+1+t, y+w+1) = \int_0^1 [(2p-1)x + 1 - p]^n [(1-2p)x + p]^m x^t \mathrm{d}x$$

Proof: *By the definition of the beta function [60],*

$$B(x, y) = \int_0^{+\infty} u^{x-1} (1+u)^{-(x+y)} \mathrm{d}u \tag{7.2}$$

we can have

$$\begin{aligned} &\sum_{x,w} C_n^x C_m^w p^{x+w} (1-p)^{y+z} B(x+z+1+t, y+w+1) \\ &= \int_0^{+\infty} \mathbb{E}u^x \cdot \mathbb{E}u^z \cdot u^t \cdot (1+u)^{-(n+m+2+t)} \mathrm{d}u \end{aligned} \tag{7.3}$$

where we regard $x \sim \text{Bin}(n, p)$ *and* $z \sim \text{Bin}(m, 1-p)$. *Thus, according to Lemma B.1, we can obtain*

$$\begin{aligned} &\int_0^{+\infty} \mathbb{E}u^x \cdot \mathbb{E}u^z \cdot u^t \cdot (1+u)^{-(n+m+3)} \mathrm{d}u \\ &= \int_0^{+\infty} \frac{[1-p+up]^n \cdot [p+(1-p)u]^m \cdot u^t}{(1+u)^{n+m+2+t}} \mathrm{d}u. \end{aligned} \tag{7.4}$$

For the integral operation, substituting u *with* $v-1$ *at first and then* v *with* $(1-x)^{-1}$*, we can conclude Lemma B.2.*

Lemma B.3 $\sum_{n=0}^{N} C_N^n \cdot x^n = (1+x)^N$.

Lemma B.4 $\sum_{n=0}^{N} C_N^n \cdot n \cdot x^n = N \cdot x \cdot (1+x)^{N-1}$.

Lemma B.5 $\sum_{n=0}^{N} C_N^n \cdot n \cdot x^{N-n} = N \cdot (1+x)^{N-1}$.

Lemma B.6 $\sum_{n=0}^{N} C_N^n \cdot n^2 \cdot x^n = Nx(1+Nx)(1+x)^{N-2}$.

Lemma B.7 $\sum_{n=0}^{N} C_N^n \cdot n^2 \cdot x^{N-n} = N(x+N)(1+x)^{N-2}$.

Lemma B.8 *If* $0 < x < 1$*, we can have*

$$\sum_{n=0}^{\lfloor N/2 \rfloor} C_N^n \cdot x^n \geq \left(1 - e^{-cN}\right) \cdot (1+x)^N$$
$$\sum_{n=\lfloor N/2 \rfloor + 1}^{N} C_N^n \cdot x^{N-n} \geq \left(1 - e^{-cN}\right) \cdot (1+x)^N.$$

where $c = 0.5(1-x)^2(1+x)^{-2}$.

Proof: *To prove the lemmas above, we firstly define*

$$F_t(x) = \sum_{n=0}^{N} C_N^n n^t x^n \tag{7.5}$$

Then, Lemma B.3 can be obtained by expanding $(1+x)^N$*. Lemma B.4 can be proved as follows*

$$\begin{aligned} F_1(x) &= \sum_{n=0}^{N} C_N^n (n+1) x^n - (1+x)^N \\ \sum_{n=0}^{N} C_N^n (n+1) x^n &= \frac{\mathrm{d}}{\mathrm{d}x} \left[x F_0(x) \right] \\ &= Nx(1+x)^{N-1} + (1+x)^N. \end{aligned} \tag{7.6}$$

Lemma B.5 can be obtained as follows

$$\begin{aligned} \sum_{n=0}^{N} C_N^n n x^{N-n} &= x^N \sum_{n=0}^{N} C_N^n n \left(\frac{1}{x}\right)^n \\ &= x^N \cdot N \cdot \frac{1}{x} \cdot \left(1 + \frac{1}{x}\right)^{N-1}. \end{aligned} \tag{7.7}$$

For Lemma B.6, we can have

$$\begin{aligned} F_2(x) &= \sum_{n=0}^{N} C_N^n (n+2)(n+1)x^n - 3F_1(x) - 2F_0(x) \\ &= \left[x^2 F_0(x)\right]' - 3F_1(x) - 2F_0(x) \end{aligned} \tag{7.8}$$

Thus, we can have

$$F_2(x) = Nx(1+Nx)(1+x)^{N-2} \tag{7.9}$$

which concludes Lemma B.6. Then, Lemma B.7 can be obtained by considering Equation 7.10*.*

$$\sum_{n=0}^{N} C_N^n n^2 x^{N-n} = x^N \sum_{n=0}^{N} C_N^n n^2 \left(\frac{1}{x}\right)^n. \tag{7.10}$$

For Lemma B.8, we can have

$$\sum_{n=0}^{\lfloor N/2 \rfloor} C_N^n x^n = (1+x)^N \sum_{n=0}^{\lfloor N/2 \rfloor} C_N^n p^n (1-p)^{N-n} \tag{7.11}$$

where $p = x(1+x)^{-1}$*. Let* $X \sim \mathrm{Bin}(N, p)$*, we can have*

$$\sum_{n=0}^{\lfloor N/2 \rfloor} C_N^n p^n (1-p)^{N-n} \geq 1 - P\left(X \geq N/2\right). \tag{7.12}$$

Since $x < 1$*,* $p < 0.5$ *and* $Np < N/2$*. Considering Hoeffding's inequality, we can get*

$$P\left(X \geq N/2\right) \leq \exp\left[-\frac{N(1-x)^2}{2(1+x)^2}\right] \tag{7.13}$$

which concludes the first inequality in Lemma B.8. Similarly, for the second inequality, we can have

$$\sum_{n=K}^{N} C_N^n x^{N-n} = (1+x)^N \sum_{n=K}^{N} C_N^n (1-p)^n p^{N-n} \tag{7.14}$$

where $K = \lfloor N/2 \rfloor + 1$*. Suppose* $Y \sim \mathrm{Bin}(N, 1-p)$*, we can have*

$$\sum_{n=K}^{N} C_N^n (1-p)^n p^{N-n} \geq 1 - P\left(Y \leq N/2\right). \tag{7.15}$$

Considering Hoeffding's inequality, we can also get

$$P\left(Y \leq N/2\right) \leq \exp\left[-\frac{N(1-x)^2}{2(1+x)^2}\right] \tag{7.16}$$

which concludes the second inequality in Lemma B.8.

Lemma B.9 *For any $x, y \geq 0$, we can have*

$$(1+x)^y \leq e^{xy}.$$

Proof: *Firstly, we can know $(1+x)^y = e^{y\log(1+x)}$. Let $f(x) = x - \log(x)$. Then, we can have $f(0) = 0$ and $f'(x) \geq 0$. Thus, $x \geq \log(1+x)$ and we can conclude Lemma B.9 by taking this inequality into the equality.*

Lemma B.10

$$g(x) = \frac{e^x}{e^x+1}$$

is a concave function when $x \in [0, +\infty)$.

Proof: *$g'(x) = (2+t(x))^{-1}$, where $t(x) = e^x + e^{-x}$. $t'(x) = e^x - e^{-x} \geq 0$ when $x \in [0, +\infty)$. Thus, $g'(x)$ is monotonically decreasing when $x \in [0, +\infty)$, which concludes Lemma B.10.*

Lemma B.11 *For $x \in (-\infty, +\infty)$,*

$$h(x) = \frac{1}{e^{|x|}+1}$$

satisfies

$$h(x) < e^x \text{ and } h(x) < e^{-x}.$$

Proof: *When $x \geq 0$, we can have*

$$h(x) < \frac{1}{e^x} = e^{-x} \leq e^x. \tag{7.17}$$

When $x \leq 0$, we can have

$$h(x) = \frac{e^x}{e^x+1} < e^x \leq e^{-x}. \tag{7.18}$$

Lemma B.12 *If $\lambda = p/(1-p)$ and $0.5 < p < 1$, then*

$$\sum\nolimits_{n=\lfloor N/2 \rfloor}^{N} C_N^n \lambda^{m-n} p^n (1-p)^m \leq [4p(1-p)]^{N/2}$$

$$\sum\nolimits_{n=0}^{\lfloor N/2 \rfloor} C_N^n \lambda^{n-m} p^n (1-p)^m \leq [4p(1-p)]^{N/2}$$

where $m = N - n$.

Proof: *For the first inequality, we can have*

$$\begin{aligned}
&\sum_{n=\lfloor N/2 \rfloor}^{N} C_N^n \lambda^{m-n} p^n (1-p)^m \\
&= \sum_{n=\lfloor N/2 \rfloor}^{N} C_N^n p^m (1-p)^n \leq \sum_{m=0}^{\lfloor N/2 \rfloor} C_N^m p^m (1-p)^n
\end{aligned} \tag{7.19}$$

According to the inequality in [3], we can have

$$\sum_{m=0}^{\lfloor N/2 \rfloor} C_N^m p^m (1-p)^n \leq \exp(-ND) \tag{7.20}$$

where $D = -0.5\log(2p) - 0.5\log(2(p-1))$, *which concludes the first inequality in Lemma B.12.*

For the second inequality, we can have

$$\begin{aligned}
&\sum_{n=0}^{\lfloor N/2 \rfloor} C_N^n \lambda^{n-m} p^n (1-p)^m \\
&= \frac{1}{[p(1-p)]^N} \sum_{n=0}^{\lfloor N/2 \rfloor} C_N^n [p^3]^n [(1-p)^3]^m \\
&= \frac{[p^3 + (1-p)^3]^N}{[p(1-p)]^N} \sum_{n=0}^{\lfloor N/2 \rfloor} C_N^n x^n (1-x)^m
\end{aligned} \tag{7.21}$$

where $x = p^3/[p^3 + (1-p)^3]$. *By using Equation* 7.20, *we can have*

$$\begin{aligned}
&\sum_{n=0}^{\lfloor N/2 \rfloor} C_N^n \lambda^{n-m} p^n (1-p)^m \\
&\leq \frac{[p^3 + (1-p)^3]^N}{[p(1-p)]^N} [x(1-x)]^{N/2} \\
&= [4p(1-p)]^{N/2}
\end{aligned} \tag{7.22}$$

which concludes the second inequality of Lemma B.12.

B.2 Proof for Lemma 5.2

To prove Lemma 5.2, we need to analyze the posterior distribution of $\mathcal{L}$ which satisfies

$$\mathbb{P}(\mathcal{L}|\boldsymbol{L}) = B(\hat{\boldsymbol{\beta}})\prod\nolimits_{i=1}^{N} B(\hat{\boldsymbol{\alpha}}_i)/[C_p \cdot \mathbb{P}(\boldsymbol{L})] \tag{7.23}$$

where C_p is the nomalization constant. This is because the samples are generated based on this distribution. However, both the numerator and denominator in Equation 7.23 are changing with $\boldsymbol{L}$, making the distribution difficult to analyze. Thus, we derive a proper approximation for the denominator of Equation 7.23 at first. Denote the labels generated by N workers for task j as vector $\boldsymbol{L}(j)$. The distribution of $\boldsymbol{L}(j)$ satisfies

$$\mathbb{P}_{\hat{\boldsymbol{\theta}}}[\boldsymbol{L}(j)] = \sum\nolimits_{k=1}^{2} \tau_k \prod\nolimits_{i=1}^{N} \mathbb{P}_i^{\delta_{ijk}} (1 - \mathbb{P}_i)^{\delta_{ij(3-k)}} \tag{7.24}$$

where $\hat{\boldsymbol{\theta}} = [\tau_1, \mathbb{P}_1, \ldots, \mathbb{P}_N]$ denotes all the parameters and $\delta_{ijk} = \mathbb{1}(L_i(j) = k)$. Then, we can have

Lemma B.13 *When $M \to \infty$,*

$$\mathbb{P}(\boldsymbol{L}) \to C_L(M) \cdot \prod\nolimits_{\boldsymbol{L}(j)} \{\mathbb{P}_{\hat{\boldsymbol{\theta}}}[\boldsymbol{L}(j)]\}^{M \cdot \mathbb{P}_{\hat{\boldsymbol{\theta}}}[\boldsymbol{L}(j)]}$$

where $C_L(M)$ denotes a constant that depends on M.

Proof: *Denote the prior distribution of $\boldsymbol{\theta}$ by π. Then,*

$$P(\mathcal{L}|\boldsymbol{\alpha}, \boldsymbol{\beta}) = \prod\nolimits_{j=1}^{M} P_{\boldsymbol{\theta}}(\boldsymbol{x}_j) \int e^{[-M \cdot d_{KL}]} \mathrm{d}\pi(\hat{\boldsymbol{\theta}}) \tag{7.25}$$

$$d_{KL} = \frac{1}{M} \sum_{j=1}^{M} \log \frac{P_{\boldsymbol{\theta}}(\boldsymbol{x}_j)}{P_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x}_j)} \to \mathrm{KL}[P_{\boldsymbol{\theta}}(\boldsymbol{x}), P_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x})] \tag{7.26}$$

where $\boldsymbol{x}_j$ denotes the labels generated for task j. The KL divergence $\mathrm{KL}[\cdot, \cdot]$, which denotes the expectation of the log-ratio between two probability distributions, is a constant for the given $\boldsymbol{\theta}$ and $\hat{\boldsymbol{\theta}}$. Thus, $\int e^{[-M \cdot d_{KL}]} \mathrm{d}\pi(\hat{\boldsymbol{\theta}}) = C_L(M)$. In addition, when $M \to \infty$, we can also have $\sum 1(\boldsymbol{x}_j = \boldsymbol{x}) \to M \cdot P_{\boldsymbol{\theta}}(\boldsymbol{x})$, which concludes Lemma B.13.

Then, we move our focus back to the samples. To quantify the effects of the collected labels, we introduce a set of variables to describe the real true labels and the collected labels. Among the n tasks of which the posterior true label is correct,

- x_0 and y_0 denote the number of tasks of which the real true label is 1 and 2, respectively.
- x_i and y_i denote the number of tasks of which worker i's label is correct and wrong, respectively.

Also, among the remaining $m = M - n$ tasks,

- w_0 and z_0 denote the number of tasks of which the real true label is 1 and 2, respectively.
- w_i and z_i denote the number of tasks of which worker i's label is correct and wrong, respectively.

Thus, we can have $x_i + y_i = n$ and $w_i + z_i = m$. Besides, we use ξ_i to denote the combination (x_i, y_i, w_i, z_i).

To compute the expectation of m/M, we need to analyze the probability distribution of m. According to Equation 5.7 in our thesis, we can know that $\mathbb{P}(m)$ satisfies

$$\mathbb{P}(m) \approx \frac{C_M^m}{Z} \sum_{\xi_0,\ldots,\xi_N} \prod_{i=0}^{N} \mathbb{P}(\xi_i|m) B(\hat{\boldsymbol{\beta}}) \prod_{i=1}^{N} B(\hat{\boldsymbol{\alpha}}_i) \tag{7.27}$$

where $Z = C_p C_L \prod_{\boldsymbol{x}} [P_{\boldsymbol{\theta}}(\boldsymbol{x})]^{M \cdot P_{\boldsymbol{\theta}}(\boldsymbol{x})}$ is independent of ξ_i and m. Meanwhile, $\hat{\beta}_1 = x_0 + z_0 + 1$, $\hat{\beta}_2 = y_0 + w_0 + 1$, $\hat{\alpha}_{i1} = x_i + z_i + 2$ and $\hat{\alpha}_{i2} = x_i + z_i + 1$. When the m tasks of which the posterior true label is wrong are given, we can know that $x_i \sim \text{Bin}(n, \mathbb{P}_i)$ and $w_i \sim \text{Bin}(m, \mathbb{P}_i)$, where $\text{Bin}(\cdot)$ denotes the binomial distribution. In addition, x_i and y_i are independent of w_i, z_i and $\xi_{k \neq i}$. Also, w_i and z_i are independent of x_i and y_i and $\xi_{k \neq i}$. Thus, we can further obtain $\mathbb{P}(m) \approx \hat{Z}^{-1} \cdot C_M^m Y(m)$, where

$$\begin{aligned} &Y(m) = e^{\log H(m, \mathbb{P}_0; M, 0) + \sum_{i=1}^{N} \log H(m, \mathbb{P}_i; M, 1)} \\ &H(m, p; M, t) = \sum\nolimits_{x=0}^{n} \sum\nolimits_{w=0}^{m} 2^{M+1} C_n^x C_m^w \times \\ &\quad p^{x+w}(1-p)^{y+z} B(x + z + 1 + t, y + w + 1) \end{aligned} \tag{7.28}$$

and $\hat{Z} = 2^{-(N+1)(M+1)} Z$. Considering $\sum_{m=1}^{M} \mathbb{P}(m) = 1$, we can know that $\hat{Z} \approx \sum_{m=1}^{M} C_M^m Y(m)$. Note that, we use $\mathbb{P}_0$ to denote the probability of true label 1, namely τ_1.

The biggest challenge of computing $P(m)$ exists in analyzing function $H(m, p; M, t)$ which we put in Section B.3 of this appendix. Here, we directly use the obtained lower and upper bounds depicted in Lemmas B.18 and B.19 and can have

$$\begin{cases} e^{C - K_l m} \lesssim Y(m) \lesssim e^{C - K_u m} & 2m \leq M \\ e^{C + \delta - K_l n} \lesssim Y(m) \lesssim e^{C + \delta - K_u n} & 2m > M \end{cases} \tag{7.29}$$

where $C = H(0, \mathbb{P}_0; M, 0) + \sum_{i=1}^{N} H(0, \mathbb{P}_i; M, 1)$ and

$$K_l = \sum_{i=0}^{N} \log \hat{\lambda}_i \ , \ K_u = 2\sum_{i=0}^{N} \log\left(2\hat{\mathbb{P}}_i\right)$$
$$\delta = \Delta \cdot \log(M) + \sum_{i=1}^{N} (-1)^{1(\mathbb{P}_i > 0.5)} \phi(\hat{\mathbb{P}}_i)$$
$$\hat{\lambda}_i = \max\left\{ \frac{\mathbb{P}_i}{\bar{\mathbb{P}}_i + \frac{1}{M}}, \frac{\bar{\mathbb{P}}_i}{\mathbb{P}_i + \frac{1}{M}} \right\} \ , \ \phi(p) = \log \frac{2\mathbb{P} - 1}{\mathbb{P}}.$$

Besides, we set a convention that $\phi(p) = 0$ when $p = 0.5$. Thereby, the expectations of m and m^2 satisfy

$$\mathbb{E}[m] \lesssim \frac{\sum_{m=0}^{M} m e^{-K_u m} + \sum_{m=0}^{M} m e^{\delta - K_u n}}{\sum_{m=0}^{k} e^{-K_l m} + \sum_{m=k+1}^{M} e^{\delta - K_l n}} \tag{7.30}$$
$$\mathbb{E}[m^2] \lesssim \frac{\sum_{m=0}^{M} m^2 e^{-K_u m} + \sum_{m=0}^{M} m^2 e^{\delta - K_u n}}{\sum_{m=0}^{k} e^{-K_l m} + \sum_{m=k+1}^{M} e^{\delta - K_l n}} \tag{7.31}$$

where $k = \lfloor M/2 \rfloor$. By using Lemmas B.4, B.5, B.6 and B.7, we can know the upper bounds of the numerator in Equations 7.30 and 7.31 are $M(\varepsilon + e^{\delta})(1+\varepsilon)^{M-1}$ and $[M^2\varepsilon^2 + M\varepsilon + e^{\delta}(M^2 + M\varepsilon)](1+\varepsilon)^{M-2}$, respectively, where $\varepsilon = e^{-K_u}$. On the other hand, by using Lemma B.8, we can obtain the lower bound of the denominator as $(1 + e^{\delta})[1 - e^{-c(\omega)M}](1+\omega)^M$, where $\omega = e^{-K_l}$ and $c(\omega) = 0.5(1-\omega)^2(1+\omega)^{-2}$. Considering $M \gg 1$, we can make the approximation that $e^{-c(\omega)M} \approx 0$ and $(1 + e^{\delta})\varepsilon/M \approx 0$. Besides, $(1+\omega)^M \geq 1$ holds because $\omega \geq 0$. In this case, Lemma 5.2 can be concluded by combining the upper bound of the numerator and the lower bound of the denominator.

B.3 H function analysis

Here, we present our analysis on the H function defined in the proof of Lemma 5.2. Firstly, we can have:

Lemma B.14 $H(m, 0.5; M, t) = 2(t+1)^{-1}$.

Lemma B.15 $H(m, p; M, t) = H(n, \bar{p}; M, t)$.

Lemma B.16 *As a function of m, $H(m, p; M, t)$ is logarithmically convex.*

Proof: *Lemma B.14 can be proved by integrating $2x^t$ on $[0, 1]$. Lemma B.15 can be* *proved by showing that* $H(n, \bar{p}; M, t)$ *has the same expression as* $H(m, p; M, t)$*. Thus, in the following proof, we focus on Lemma B.16. Fixing* p*,* M *and* t*, we denote* $\log(H)$ *by* $f(m)$*. Then, we compute the first-order derivative as*

$$H(m)f'(m) = 2^{M+1} \int_0^1 \lambda u^n (1-u)^m x^t \mathrm{d}x \tag{7.32}$$

where $u = (2p-1)x + 1 - p$ *and* $\lambda = \log(1-u) - \log(u)$*. Furthermore, we can solve the second-order derivative as*

$$\begin{aligned} &2^{-2(M+1)} H^2(m) f''(m) = \\ &\int_0^1 g^2(x)\mathrm{d}x \int_0^1 h^2(x)\mathrm{d}x - \left(\int_0^1 g(x)h(x)\mathrm{d}x\right)^2 \end{aligned} \tag{7.33}$$

where the functions $g, h : (0,1) \to \mathbb{R}$ *are defined by*

$$g = \lambda\sqrt{u^n(1-u)^m}\ ,\ h = \sqrt{u^n(1-u)^m}. \tag{7.34}$$

By the Cauchy-Schwarz inequality,

$$\int_0^1 g^2(x)\mathrm{d}x \int_0^1 h^2(x)\mathrm{d}x \geq \left(\int_0^1 g(x)h(x)\mathrm{d}x\right)^2 \tag{7.35}$$

we can know that $f''(m) \geq 0$ *always holds, which concludes that* f *is convex and* H *is logarithmically convex.*

Then, for the case that $t = 1$ and $M \gg 1$, we can further derive the following three lemmas for $H(m, p; M, 1)$:

Lemma B.17 *The ratio between two ends satisfies*

$$\log \frac{H(0, p; M, 1)}{H(M, p; M, 1)} \approx \begin{cases} \log(M) + \epsilon(p) & p > 0.5 \\ 0 & p = 0.5 \\ -\log(M) - \epsilon(\bar{p}) & p < 0.5 \end{cases}$$

where $\epsilon(p) = \log(2p-1) - \log(p)$ and $\epsilon(p) = 0$ if $p = 0.5$.

Lemma B.18 *The lower bound can be calculated as*

$$\log H(m, p) \gtrsim \begin{cases} H(0, p) - k_l \cdot m & 2m \leq M \\ H(M, p) - k_l \cdot n & 2m > M \end{cases}$$

where $k_l = \log\left(\max\left\{p/(\bar{p} + M^{-1}), \bar{p}/(p + M^{-1})\right\}\right)$*.*

Lemma B.19 *The upper bound can be calculated as*

$$\log H(m,p) \lesssim \begin{cases} H(0,p) - k_u \cdot m & 2m \le M \\ H(M,p) - k_u \cdot n & 2m > M \end{cases}$$

where $n = M - m$ and $k_u = 2\log(2 \cdot \max\{p, \bar{p}\})$.

Proof: *By Lemma B.14, $\log H(m, 0.5; M, 1) \equiv 0$, which proves the above three lemmas for the case that $p = 0.5$. Considering the symmetry ensured by Lemma B.15, we thus focus on the case that $p > 0.5$ in the following proof and transform $H(m,p)$ into the following formulation*

$$H(m,p) = \omega(p) \cdot \int_{\bar{p}}^{p} x^n (1-x)^m (x - 1 + p)\mathrm{d}x \tag{7.36}$$

where $\omega(p) = 2^{M+1}/(2p-1)^2$. Then, we can solve $H(0,p)$ and $H(M,p)$ as

$$\begin{aligned} H(0,p) &= \omega(p) \int_{\bar{p}}^{p} x^M (x - \bar{p})\mathrm{d}x \\ &= \frac{(2p)^{M+1}}{(2p-1)(M+1)} - O\left(\frac{(2p)^{M+1}}{M^2}\right) \end{aligned} \tag{7.37}$$

$$\begin{aligned} H(M,p) &= \omega(p) \int_{\bar{p}}^{p} (1-x)^M (x - \bar{p})\mathrm{d}x \\ &= \frac{p(2p)^{M+1}}{(2p-1)^2(M+1)(M+2)} - O\left(\frac{(2\bar{p})^{M+1}}{M+2}\right). \end{aligned} \tag{7.38}$$

Using the Taylor expansion of function $\log(x)$, we can calculate the ratio in Lemma B.17 as

$$\log \frac{H(0,p)}{H(M,p)} = \log(M) + \log \frac{2p-1}{p} + O\left(\frac{1}{M}\right) \tag{7.39}$$

which concludes Lemma B.17 when $M \gg 1$.

Furthermore, we can solve $H(1,p)$ as

$$\begin{aligned} H(1,p) &= \omega(p) \int_{\bar{p}}^{p} x^{M-1} (x - \bar{p})\mathrm{d}x - H(0,p) \\ &= \frac{(2\bar{p} + M^{-1})(2p)^M}{(2p-1)(M+1)} - O\left(\frac{(2p)^{M+1}}{M^2}\right) \end{aligned} \tag{7.40}$$

The value ratio between $m = 0$ and $m = 1$ then satisfies

$$\log \frac{H(1,p)}{H(0,p)} = \log \frac{p}{\bar{p} + M^{-1}} + O\left(\frac{1}{M}\right). \tag{7.41}$$

By Rolle's theorem, there exists a $c \in [m, m+1]$ satisfying

$$\log H(1,p) - \log H(0,p) = f'(c) \tag{7.42}$$

where $f(m) = \log H(m,p)$. *Meanwhile, Lemma B.16 ensures that* $f''(m) \geq 0$ *always holds. Thus, we can have*

$$\log H(m+1,p) - \log H(m,p) \geq \log \frac{H(1,0)}{H(0,p)} \tag{7.43}$$

which concludes the first case of Lemma B.18. Similarly, we compute the ratio between $m = M-1$ *and* M *as*

$$\log \frac{H(M,p)}{H(M-1,p)} = \log \frac{p}{\bar{p} + M^{-1}} + O\left(\frac{1}{M}\right). \tag{7.44}$$

Meanwhile, Rolle's theorem and Lemma B.16 ensure that

$$\log H(m,p) - \log H(m-1,p) \leq \log \frac{H(M,0)}{H(M-1,p)} \tag{7.45}$$

which concludes the second case of Lemma B.18.

Lastly, we focus on the upper bound described by Lemma B.19. According to the inequality of arithmetic and geometric means, $x(1-x) \leq 2^{-2}$ *holds for any* $x \in [0,1]$. *Thus, when* $2m \leq M$ *(i.e.* $n \geq m$*), we can have*

$$H(m,p) \leq 2^{-2m}\omega(p) \cdot \int_{\bar{p}}^{p} x^{n-m}(x-1+p)\mathrm{d}x \tag{7.46}$$

where the equality only holds when $m = 0$.

$$\int_{\bar{p}}^{p} x^{n-m}(x-1+p)\mathrm{d}x = \frac{(2p-1)p^{\delta}}{\delta} + \frac{\Delta}{\delta(\delta+1)} \tag{7.47}$$

where $\delta = n - m + 1$ *and* $\Delta = \bar{p}^{\delta+1} - p^{\delta+1} < 0$. *Hence,*

$$\log \frac{H(m,p)}{H(0,p)} \leq -2m[\log(2p) - \varepsilon(m)] + O\left(\frac{1}{M}\right) \tag{7.48}$$

where $\varepsilon(m) = -(2m)^{-1}[\log(n-m+1) - \log(M+1)]$. *Since* $\log(x)$ *is a concave function, we can know that*

$$\varepsilon(m) \leq (M)^{-1}\log(M+1) = O\left(M^{-1}\right) \tag{7.49}$$

which concludes the first case in Lemma B.19. Similarly, for $2m > M$ *(i.e.* $n < m$*), we can have*

$$\log \frac{H(m,p)}{H(M,p)} \leq -2n[\log(2p) - \hat{\varepsilon}(n)] + O\left(\frac{1}{M}\right) \tag{7.50}$$

where $\hat{\varepsilon}(n) \leq O(M^{-1})$. *Thereby, we can conclude the second case of Lemma B.19. Note that the case where* $p < 0.5$ *can be derived by using Lemma B.15.*

For the case that $t = 0$ and $M \gg 1$, using the same method as the above proof, we can derive the same lower and upper bounds as Lemmas B.19 and B.18. On the other hand, for $t = 0$, Lemma B.17 does not hold and we can have

Lemma B.20 $H(m, p; M, 0) = H(n, p; M, 0)$.

Proof: *When $t = 0$,*

$$H(m, p) = 2^{M+1}(2p-1)^{-1}\int_{\bar{p}}^{p} x^n(1-x)^m \mathrm{d}x. \tag{7.51}$$

Then, substituting x as $1 - v$ concludes Lemma B.20.

B.4 Proof for Lemma 5.3

In our Bayesian inference algorithm, when $M \gg 1$, the estimated accuracy $\tilde{A}$ satisfies

$$\tilde{A} \approx 1 - \mathbb{E}g(\tilde{\sigma}_j) \ \ , \ \ g(\tilde{\sigma}_j) = 1/(1 + e^{|\tilde{\sigma}_j|}). \tag{7.52}$$

From the proof of Theorem 5.1, we can know that $\tilde{\mathbb{P}}_i^t \approx \mathbb{P}_i^t$. In this case, according to Equation 5.11, we can have

$$\tilde{\sigma}_j(\mathbb{P}_i) \approx \log\left(\frac{\tau_1}{\tau_2}\lambda_i^{\delta_{ij1}-\delta_{ij2}}\prod\nolimits_{k\neq i}\lambda_H^{\delta_{kj1}-\delta_{kj2}}\right). \tag{7.53}$$

where $\lambda_i = \mathbb{P}_i/(1-\mathbb{P}_i)$ and $\lambda_H = \mathbb{P}_H/(1-\mathbb{P}_H)$.

We know that $\tilde{A} \leq 1.0$ holds no matter what strategy worker i takes. To prove Lemma 2, we still need to know the lower bound of $\tilde{A}$. Thus, we consider two extreme cases where worker i intentionally provides low-quality labels:

Case 1: If $\mathbb{P}_i = 0.5$, we can eliminate λ_i from Equation 7.53 because $\lambda_i = 1$. Furthermore, according to Lemma B.11, we can know that $g(\tilde{\sigma}_j) < e^{\tilde{\sigma}_j}$ and $g(\tilde{\sigma}_j) < e^{-\tilde{\sigma}_j}$ both hold. Thus, we build a tighter upper bound of $g(\tilde{\sigma}_j)$ by dividing all the combinations of δ_{kj1} and δ_{kj2} in Equation 7.53 into two sets and using the smaller one of $e^{\tilde{\sigma}_j}$ and $e^{-\tilde{\sigma}_j}$ in each set. By using this method, if the true label is 1, we can have $\mathbb{E}_{[L(j)=1]}g(\tilde{\sigma}_j) < q_1 + q_2$, where

$$q_1 = \frac{\tau_2}{\tau_1}\sum\nolimits_{n=K+1}^{N-1} C_{N-1}^{n}(\frac{1}{\lambda_H})^{n-m}\mathbb{P}_H^n(1-\mathbb{P}_H)^m$$
$$q_2 = \frac{\tau_1}{\tau_2}\sum\nolimits_{n=0}^{K} C_{N-1}^{n}{\lambda_H}^{n-m}\mathbb{P}_H^n(1-\mathbb{P}_H)^m$$
$$n = \sum\nolimits_{k\neq i}\delta_{kj1} \ , \ m = \sum\nolimits_{k\neq i}\delta_{kj2}$$

and $K = \lfloor (N-1)/2 \rfloor$. By using Lemma B.12, we can thus get

$$\mathbb{E}_{[L(j)=1]} g(\tilde{\sigma}_j) < c_\tau [4\mathbb{P}_H(1-\mathbb{P}_H)]^{\frac{N-1}{2}}.$$

where $c_\tau = \tau_1\tau_2^{-1} + \tau_1^{-1}\tau_2$. Similarly,

$$\mathbb{E}_{[L(j)=2]} g(\tilde{\sigma}_j) < c_\tau [4\mathbb{P}_H(1-\mathbb{P}_H)]^{\frac{N-1}{2}}.$$

Thereby, $\tilde{A} > 1 - c_\tau[4\mathbb{P}_H(1-\mathbb{P}_H)]^{\frac{N-1}{2}} = 1 - \psi$.

Case 2: If $\mathbb{P}_i = 1 - \mathbb{P}_H$, we can rewrite Equation 7.53 as

$$\tilde{\sigma}_j(1-\mathbb{P}_H) \approx \log\left(\frac{\tau_1}{\tau_2}\lambda_H^{x-y}\prod\nolimits_{k\neq i}\lambda_H^{\delta_{kj1}-\delta_{kj2}}\right)$$

where $x = \delta_{ij2}$ and $y = \delta_{ij1}$. Since $\mathbb{P}_i = 1 - \mathbb{P}_H$, x and y actually has the same distribution as δ_{kj1} and δ_{kj2}. Thus, the distribution of $\tilde{\sigma}_j(1-\mathbb{P}_H)$ is actually the same as $\tilde{\sigma}_j(\mathbb{P}_H)$. In other words, since Theorem 5.1 ensures $\tilde{\mathbb{P}}_i \approx \mathbb{P}_i$, our Bayesian inference algorithm uses the information provided by worker i via flipping the label when $\mathbb{P}_i < 0.5$.

Thus, even if worker i intentionally lowers the label quality, $\tilde{A} \geq 1 - \psi$ still holds. Considering $F(\cdot)$ is a non-decreasing monotonic function, we conclude Lemma 5.3.

Bibliography

[1] Shipra Agrawal and Nikhil R Devanur. Bandits with concave rewards and convex knapsacks. In *Proceedings of the 15th ACM Conference on Economics and Computation*, pages 989–1006, 2014.

[2] Nima Anari, Gagan Goel, and Afshin Nikzad. Mechanism design for crowdsourcing: An optimal 1-1/e competitive budget-feasible mechanism for large markets. In *Proceedings of the 55th Annual Symposium on Foundations of Computer Science*, pages 266–275, 2014.

[3] R. Arratia and L. Gordon. Tutorial on large deviations for the binomial distribution. *Bulletin of Mathematical Biology*, 51(1):125–131, Jan 1989.

[4] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

[5] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Yaron Singer. Learning on a budget: posted price mechanisms for online procurement. In *Proceedings of the 13th ACM Conference on Economics and Computation*, pages 128–145, 2012.

[6] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. Bandits with knapsacks. In *Proceedings of the 54th Annual Symposium on Foundations of Computer Science*, pages 207–216, 2013.

[7] Chris Buckley, Matthew Lease, and Mark D. Smucker. Overview of the trec 2010 relevance feedback track. In *TREC Notebook*, 2010.

[8] R. Burden and J. Faires. *Numerical Analysis*. Cengage Learning, 2004.

[9] Praphul Chandra, Yadati Narahari, Debmalya Mandal, and Prasenjit Dey. Novel mechanisms for online crowdsourcing with unreliable, strategic agents. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 1256–1262, 2015.

[10] Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems*, pages 2249–2257, 2011.

[11] Erick Chastain, Adi Livnat, Christos Papadimitriou, and Umesh Vazirani. Algorithms, games, and evolution. *PNAS*, 111(29):10620–10623, 2014.

[12] Xi Chen, Qihang Lin, and Dengyong Zhou. Statistical decision making for optimal budget allocation in crowd labeling. *Journal of Machine Learning Research*, 16:1–46, 2015.

[13] Richard Combes and Alexandre Proutiere. Unimodal bandits: Regret lower bounds and optimal algorithms. In *Proceedings of the 31st International Conference on Machine Learning*, pages 521–529, 2014.

[14] Florian Daniel, Pavel Kucherbaev, Cinzia Cappiello, Boualem Benatallah, and Mohammad Allahbakhsh. Quality control in crowdsourcing: A survey of quality attributes, assessment techniques, and assurance actions. *ACM Computing Surveys (CSUR)*, 51(1):7, 2018.

[15] Anirban Dasgupta and Arpita Ghosh. Crowdsourced judgement elicitation with endogenous proficiency. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 319–330, 2013.

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

[17] Djellel Eddine Difallah, Michele Catasta, Gianluca Demartini, Panagiotis G Ipeirotis, and Philippe Cudré-Mauroux. The dynamics of micro-task crowdsourcing: The case of amazon mturk. In *Proceedings of the 24th International Conference on World Wide Web*, pages 238–247, 2015.

[18] Yaakov Engel, Shie Mannor, and Ron Meir. Reinforcement learning with gaussian processes. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 201–208, 2005.

[19] Ju Fan, Guoliang Li, Beng Chin Ooi, Kian-lee Tan, and Jianhua Feng. icrowd: An adaptive crowdsourcing framework. In *Proceedings of the International Conference on Management of Data*, pages 1015–1030, 2015.

[20] Michael J Franklin, Donald Kossmann, Tim Kraska, Sukriti Ramesh, and Reynold Xin. Crowddb: answering queries with crowdsourcing. In *Proceedings of the International Conference on Management of Data*, pages 61–72, 2011.

[21] Xi Alice Gao, Andrew Mao, Yiling Chen, and Ryan Prescott Adams. Trick or treat: putting peer prediction to the test. In *Proceedings of the 15th ACM conference on Economics and Computation*, pages 507–524, 2014.

[22] Yang Gao, Yan Chen, and KJ Ray Liu. On cost-effective incentive mechanisms in microtask crowdsourcing. *IEEE Transactions on Computational Intelligence and AI in Games*, 7(1):3–15, 2015.

[23] Yihan Gao and Aditya Parameswaran. Finish them!: Pricing algorithms for human computation. *Proceedings of the VLDB Endowment*, 7(14):1965–1976, 2014.

[24] Aurélien Garivier and Olivier Cappé. The kl-ucb algorithm for bounded stochastic bandits and beyond. In *Proceedings of the 24th Annual Conference On Learning Theory*, pages 359–376, 2011.

[25] Milica Gasic and Steve Young. Gaussian processes for pomdp-based dialogue manager optimization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(1):28–40, 2014.

[26] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

[27] Geoffrey J. Gordon. Reinforcement Learning with Function Approximation Converges to a Region. In *Proceedings of the 14th Annual Conference on Neural Information Processing Systems*, pages 1040–1046, 2000.

[28] Todd L Graves and Tze Leung Lai. Asymptotically efficient adaptive choice of control laws incontrolled markov chains. *SIAM journal on control and optimization*, 35(3):715–743, 1997.

[29] Chien-Ju Ho, Shahin Jabbari, and Jennifer Wortman Vaughan. Adaptive task assignment for crowdsourced classification. In *Proceedings of the 30th International Conference on Machine Learning*, pages 534–542, 2013.

[30] John J Horton and Richard J Zeckhauser. Algorithmic wage negotiations: Applications to paid crowdsourcing. *Proceedings of the 1st International Conference about Crowdsourcing*, pages 2–5, 2010.

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

[32] Zehong Hu, Yitao Liang, Jie Zhang, Zhao Li, and Yang Liu. Inference aided reinforcement learning for incentive mechanism design in crowdsourcing. In *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems*, 2018.

[33] Zehong Hu, Yang Liu, Yitao Liang, and Jie Zhang. A reinforcement learning framework for eliciting high quality information. In *Proceedings of NIPS workshop on Machine Learning in the Presence of Strategic Behavior*, 2017.

[34] Zehong Hu, Meng Sha, Moath Jarrah, Jie Zhang, and Hui Xi. Efficient computation of emergent equilibrium in agent-based simulation. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 2501–2508, 2016.

[35] Zehong Hu and Jie Zhang. Optimal posted-price mechanism in microtask crowdsourcing. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 228–234, 2017.

[36] Zehong Hu and Jie Zhang. A novel strategy for active task assignment in crowd labeling. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 1538–1545, 2018.

[37] Gang Hua, Chengjiang Long, Ming Yang, and Yan Gao. Collaborative active learning of a kernel machine ensemble for recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1209–1216, 2013.

[38] 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, 2010.

[39] Radu Jurca, Boi Faltings, et al. Mechanisms for making crowds truthful. *Journal of Artificial Intelligence Research*, 34(1):209, 2009.

[40] Ece Kamar, Severin Hacker, and Eric Horvitz. Combining human and machine intelligence in large-scale crowdsourcing. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, pages 467–474, 2012.

[41] Yuqing Kong and Grant Schoenebeck. A framework for designing information elicitation mechanisms that reward truth-telling. *arXiv preprint arXiv:1605.01021*, 2016.

[42] Adriana Kovashka, Olga Russakovsky, Li Fei-Fei, Kristen Grauman, et al. Crowdsourcing in computer vision. *Foundations and Trends® in Computer Graphics and Vision*, 10(3):177–243, 2016.

[43] Cecilia Maria; Fayomi Toks; Imaizumi Saori; Ipeirotis Panos Kuek, Siou Chew; Paradi-Guilford. The global opportunity in online outsourcing. *http://documents.worldbank.org/curated/en/138371468000900555/The-global-opportunity-in-online-outsourcing*. Technical report, 2015.

[44] Tze Leung Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.

[45] David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the 11st International Conference on Machine Learning*, pages 148–156, 1994.

[46] Yitao Liang, Marlos C. Machado, Erik Talvitie, and Michael Bowling. State of the art control of atari games using shallow reinforcement learning. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, pages 485–493, 2016.

[47] Christopher H Lin, M Mausam, and Daniel S Weld. Re-active learning: Active learning with relabeling. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 1845–1852, 2016.

[48] Qiang Liu, Jian Peng, and Alexander T Ihler. Variational inference for crowdsourcing. In *Proceedings of the 26th Annual Conference on Neural Information Processing Systems*, pages 692–700, 2012.

[49] Siyuan Liu, Chunyan Miao, Yuan Liu, Han Yu, Jie Zhang, and Cyril Leung. An incentive mechanism to elicit truthful opinions for crowdsourced multiple choice consensus tasks. In *Proceedings of International Conference on Web Intelligence and Intelligent Agent Technology*, pages 96–103, 2015.

[50] Yang Liu and Yiling Chen. Machine-learning aided peer prediction. In *Proceedings of the 18th ACM Conference on Economics and Computation*, pages 63–80, 2017.

[51] Yang Liu and Yiling Chen. Sequential peer prediction: Learning to elicit effort using posted prices. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 607–613, 2017.

[52] Yuan Liu and Jie Zhang. Robustness evaluation of incentive mechanisms. In *Proceedings of the International Conference on Autonomous Agents and Multi-agent Systems*, pages 1293–1294, 2013.

[53] Yuan Liu, Jie Zhang, Bo An, and Sandip Sen. A practical robustness measure of incentive mechanisms. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, pages 1379–1380, 2014.

[54] Yuan Liu, Jie Zhang, Bo An, and Sandip Sen. A simulation framework for measuring robustness of incentive mechanisms and its implementation in reputation systems. *Autonomous Agents and Multi-Agent Systems*, pages 1–20, 2015.

[55] Richard D McKelvey and Thomas R Palfrey. Quantal response equilibria for normal form games. *Games and economic behavior*, 10(1):6–38, 1995.

[56] István Mező. Some infinite sums arising from the weierstrass product theorem. *Applied Mathematics and Computation*, 219(18):9838–9846, 2013.

[57] Nolan Miller, Paul Resnick, and Richard Zeckhauser. Eliciting informative feedback: The peer-prediction method. *Management Science*, 51(9):1359–1373, 2005.

[58] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level Control through Deep Reinforcement Learning. *Nature*, 518(7540):529–533, 02 2015.

[59] Jafar Muhammadi, Hamid R Rabiee, and Abbas Hosseini. A unified statistical framework for crowd labeling. *Knowledge and Information Systems*, 45(2):271–294, 2015.

[60] Frank W. J. Olver. *NIST Handbook of Mathematical Functions.* Cambridge University Press, 2010.

[61] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 115–124, 2005.

[62] Aditya Parameswaran, Stephen Boyd, Hector Garcia-Molina, Ashish Gupta, Neoklis Polyzotis, and Jennifer Widom. Optimal crowd-powered rating and filtering algorithms. *Proceedings of the VLDB Endowment*, 7(9):685–696, 2014.

[63] Dražen Prelec. A bayesian truth serum for subjective data. *science*, 306(5695):462–466, 2004.

[64] Goran Radanovic and Boi Faltings. A robust bayesian truth serum for non-binary signals. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, pages 833–839, 2013.

[65] A Rajeev, Demosthenis Teneketzis, and Venkatachalam Anantharam. Asymptotically efficient adaptive allocation schemes for controlled iid processes: Finite parameter space. *IEEE Transactions on Automatic Control*, 34(3), 1989.

[66] 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(Apr):1297–1322, 2010.

[67] Javier Rivas. Mechanism design and bounded rationality: The case of type misreporting. *Mathematical Social Sciences*, 78:6 – 13, 2015.

[68] R Tyrrell Rockafellar and Stanislav Uryasev. Conditional value-at-risk for general loss distributions. *Journal of banking and finance*, 26(7):1443–1471, 2002.

[69] Paul Rodway, Astrid Schepman, and Jordana Lambert. Preferring the one in the middle: Further evidence for the centre-stage effect. *Applied Cognitive Psychology*, 26(2):215–222, 2012.

[70] Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

[71] Victor S Sheng, Foster Provost, and Panagiotis G Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th International Conference on Knowledge Discovery and Data Mining*, pages 614–622, 2008.

[72] Aashish Sheshadri and Matthew Lease. Square: A benchmark for research on computing crowd consensus. In *Proceedings of the 1st AAAI Conference on Human Computation and Crowdsourcing*, pages 156–164, 2013.

[73] Victor Shnayder, Arpit Agarwal, Rafael Frongillo, and David C Parkes. Informed truthfulness in multi-task peer prediction. In *Proceedings of the 17th ACM Conference on Economics and Computation*, pages 319–330, 2016.

[74] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–359, 10 2017.

[75] Herbert Alexander Simon. *Models of bounded rationality: Empirically grounded economic reason*, volume 3. MIT press, 1982.

[76] Edwin Simpson and Stephen Roberts. Bayesian methods for intelligent task assignment in crowdsourcing systems. In *Decision Making: Uncertainty, Imperfection, Deliberation and Scalability*, pages 1–32. Springer, 2015.

[77] Edwin D Simpson, Matteo Venanzi, Steven Reece, Pushmeet Kohli, John Guiver, Stephen J Roberts, and Nicholas R Jennings. Language understanding in the wild: Combining crowdsourcing and machine learning. In *Proceedings of the 24th International Conference on World Wide Web*, pages 992–1002, 2015.

[78] Yaron Singer and Manas Mittal. Pricing mechanisms for crowdsourcing markets. In *Proceedings of the 22th International Conference on World Wide Web*, pages 1157–1166, 2013.

[79] Adish Singla and Andreas Krause. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 1167–1178, 2013.

[80] Aleksandrs Slivkins and Jennifer Wortman Vaughan. Online decision making in crowdsourcing markets: Theoretical challenges. *ACM SIGecom Exchanges*, 12(2):4–23, 2014.

[81] 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. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 254–263, 2008.

[82] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[83] Csaba Szepesvári. *Algorithms for Reinforcement Learning*. Synthesis lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool, 2010.

[84] Alberto Tarable, Alessandro Nordio, Emilio Leonardi, and Marco Ajmone Marsan. The importance of worker reputation information in microtask-based crowd work systems. *IEEE Transactions on Parallel and Distributed Systems*, 28(2):558–571, 2017.

[85] Long Tran-Thanh, Sebastian Stein, Alex Rogers, and Nicholas R. Jennings. Efficient crowdsourcing of unknown experts using bounded multi-armed bandits. *Artificial Intelligence*, 214:89 – 111, 2014.

[86] Dimitris G Tzikas, Aristidis C Likas, and Nikolaos P Galatsanos. The variational approximation for bayesian inference. *IEEE Signal Processing Magazine*, 25(6):131–146, 2008.

[87] Luis Von Ahn and Laura Dabbish. Designing games with a purpose. *Communications of the ACM*, 51(8):58–67, 2008.

[88] Luis Von Ahn, Ruoran Liu, and Manuel Blum. Peekaboom: a game for locating objects in images. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 55–64, 2006.

[89] Maja Vukovic. Crowdsourcing for enterprises. In *Proceedings of World Conference on Service*, pages 686–692. IEEE, 2009.

[90] Bo Waggoner and Yiling Chen. Output agreement mechanisms and common knowledge. In *Proceedings of the 2nd AAAI Conference on Human Computation and Crowdsourcing*, pages 220–226, 2014.

[91] Lu Wang and Zhi-Hua Zhou. Cost-saving effect of crowdsourcing learning. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, pages 2111–2117, 2016.

[92] Peter Welinder and Pietro Perona. Online crowdsourcing: rating annotators and obtaining cost-effective labels. In *Proceedings of the Computer Vision and Pattern Recognition Workshop*, pages 25–32, 2010.

[93] Jens Witkowski, Yoram Bachrach, Peter Key, and David Christopher Parkes. Dwelling on the negative: Incentivizing effort in peer prediction. In *Proceedings of the 1st AAAI Conference on Human Computation and Crowdsourcing*, pages 190–197, 2013.

[94] Jens Witkowski and David C Parkes. Peer prediction without a common prior. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, pages 964–981, 2012.

[95] Jens Witkowski and David C Parkes. A robust bayesian truth serum for small populations. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pages 1492–1498, 2012.

[96] Ming Yin and Yiling Chen. Bonus or not? learn to reward in crowdsourcing. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 201–207, 2015.

[97] Jie Zhang and Robin Cohen. Design of a mechanism for promoting honesty in e-marketplaces. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence*, pages 1495–1500, 2007.

[98] Xinglin Zhang, Zheng Yang, Zimu Zhou, Haibin Cai, Lei Chen, and Xiangyang Li. Free market of crowdsourcing: Incentive mechanism design for mobile sensing. *IEEE transactions on parallel and distributed systems*, 25(12):3190–3200, 2014.

[99] Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. Truth inference in crowdsourcing: is the problem solved? *Proceedings of the VLDB Endowment*, 10(5):541–552, 2017.

[100] Yudian Zheng, Jiannan Wang, Guoliang Li, Reynold Cheng, and Jianhua Feng. Qasca: A quality-aware task assignment system for crowdsourcing applications. In *Proceedings of the International Conference on Management of Data*, pages 1031–1046, 2015.

[101] Dengyong Zhou, Qiang Liu, John Platt, and Christopher Meek. Aggregating ordinal labels from crowds by minimax conditional entropy. In *Proceedings of the 31st International Conference on Machine Learning*, pages 262–270, 2014.

[102] Changbo Zhu, Huan Xu, and Shuicheng Yan. Online crowdsourcing. *arXiv preprint arXiv:1512.02393*, 2015.