Quality-Aware Data Crowdsourcing and Federated Learning in Wireless Networks

by

Yuxi Zhao

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
December 10, 2022

Keywords: Quality-aware data crowdsourcing, distributed computation, wireless federated learning, channel-aware adaptive optimization

Approved by

Xiaowen Gong, Chair, Assistant Professor of Electrical and Computer Engineering
Shiwen Mao, Professor and Earle C. Williams Eminent Scholar Chair of Electrical and Computer Engineering
SueAnne Nichole Griffith, Assistant Professor of Electrical and Computer Engineering
Yang Zhou, Assistant Professor of Computer Science and Software Engineering
Tao Shu, Associate Professor of Computer Science and Software Engineering

Abstract

Data crowdsourcing (referred to as "crowdsourcing" for brevity) has found a wide range of applications. In principle, crowdsourcing leverages the "wisdom" of a potentially large crowd of workers (e.g., mobile users) for tasks. One main advantage of crowdsourcing lies in that it can exploit the diversity of inherently inaccurate data from many workers by aggregating the data obtained by the crowd, such that the data accuracy (referred to as "data quality") after aggregation can substantially improve. Quality-aware crowdsourcing is beneficial as it makes use of workers' data quality to perform task allocation and data aggregation. However, a worker's quality and data can be her private information that she may have incentive to misreport to the crowdsourcing requester. Moreover, a worker's quality and data can depend on her sensitive information (e.g., location), which can be inferred from the outcomes of task allocation and data aggregation by an adversary. In addition, crowdsourcing is vulnerable to data poisoning attacks, where the attacker reports malicious data to reduce aggregated data accuracy. We study privacy-preserving crowdsourcing mechanisms for truthful data quality elicitation, and malicious data attacks on dynamic crowdsourcing.

In federated learning (FL), machine learning (ML) models are trained distributively on edge devices without transmitting data samples from a large number of devices. In such a setting, the quality of a local model update is intimately related to the variance of the local stochastic gradient, which depends on the mini-batch data size used to compute the update. Wireless federated learning (WFL) can achieve collaborative intelligence in wireless edge networks. A general consensus is that WFL can support intelligent control and management of wireless communications and networks, and can enable many AI applications based on wireless networked systems.

In distributed stochastic gradient descent which is a typical method of FL, the convergence rate of the trained machine learning model in FL depends heavily on which users participate in the learning process, given the heterogeneous quality of their local model updates and the

unique features of wireless edge networks. The quality of a local parameter update is measured by the variance of the update, determined by the data sampling size (a.k.a. mini-batch size) used to compute the update. It is important to observe that the quality of local updates can be treated as a design parameter and used as a *control "knob"* (via the mini-batch size) to be adapted across users and over time. Such quality-aware distributed computation can substantially improve the learning accuracy of FL. To achieve a desired tradeoff between learning accuracy and communication and computation costs, participating devices of FL in each round and their local updates' quality should be determined based on their impacts on the eventual training loss, as well as devices' channel conditions and computation costs. We characterize performance bounds on the training loss as a function of local updates' quality over the training process, for IID and non-IID data with convex setting, non-convex setting, and asynchronous setting. Based on the insights revealed by the performance bounds, we develop cost-effective dynamic distributed learning algorithms that adaptively select participating users and their mini-batch sizes, based on users' communication and computation costs.

In many applications of ML (e.g., image classification), the labels of training data need to be generated manually by human agents (e.g., recognizing and annotating objects in an image), which are usually costly and error-prone. The labeling of training data can be seen as data crowdsourcing. Given the strategic behavior of clients who may not make desired effort in their local data labeling and local model computation (quantified by the mini-batch size used in the stochastic gradient computation), and may misreport their local models to the FL server, we study characterizing the performance bounds on the training loss and devise labeling and computation effort and local model elicitation mechanisms which incentivize strategic clients to make truthful efforts as desired by the server in local data labeling and local model computation, and also report true local models to the server.

Acknowledgments

My sincere appreciation goes to my major advisor Prof. Xiaowen Gong for his great support and persevering guidance during my Ph.D. study. Based on the depth and width of his extensive knowledge base, he inspires me on choosing research topics, formulating important problems, and develop effective solutions. He is always passionate about research and gives me full support when I needed help.

In addition, I would like to thank my committee members, Prof. Shiwen Mao, Prof. SueAnne Griffith, Prof. Yang Zhou, and the university reader Prof. Tao Shu. Your advice and support are the torch lighting up my dissertation road, and your discussions and suggestions enrich my knowledge base and deepen my understanding of the research areas.

I would like to thank my parents Jun Zhao and Shuhua Gao, and all my family members. They have given me endless and unconditional love. They always believe in me and are my strongest backing. I also want to express my thankfulness to my friends: Ting Yu, Chuchu Chen, Cuiling Wang, Junyao Xing, Yifan Ye, who support and encourage me like my sisters, and all my friends in Auburn, especially, Nianchu Hou, the person who understands me the most in the world. Last but not least, I thank Dr. Ruolin Zhou for motivating and encouraging me to pursue my Ph.D..

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Quality-Aware Data Crowdsourcing

Data crowdsourcing is a promising paradigm that leverages the "wisdom" of a potentially large crowd of "workers" in many application domains. The applications of crowdsourcing can be generally categorized as physical sensing (also known as "crowdsensing") such as spectrum sensing [1], traffic monitoring [2], environmental monitoring [3], and human intelligence such as image labeling and speech transcribing [4, 5]. One main advantage of crowdsourcing lies in that it can exploit the diversity of inherently inaccurate data from many workers by aggregating the data obtained by the crowd, such that the data accuracy (referred to as "data quality") after aggregation can substantially improve. With enormous opportunities and growing popularities of data-driven technologies, crowdsourcing is a promising paradigm to harness the power of big data via machine learning, and enable artificial intelligence in various application domains, such as image classification [4] and indoor localization [6].

To exploit the potential of crowdsourcing, it is beneficial to allocate tasks to workers based on their *quality*. A worker's quality[1] can capture the *intrinsic* accuracy of the worker's data relative to the ground truth of the interested variable, and it generally varies for different workers depending on a worker's characteristics (e.g., location, sensors' capabilities). For example, if the task is to detect whether a licensed wireless device is transmitting or not (for opportunistic spectrum access by unlicensed users), then the quality of a worker's data is the probability

[1] We use "worker quality" and "quality" exchangeably in this paper. "Worker quality" is distinguished from "data quality".

of correct detection, which depends on the worker's location relative to the licensed device. Workers generally have *diverse* quality.

A worker's quality is often unknown to the worker and also the requester (e.g., in spectrum sensing, the location of the wireless device to be observed is unknown). In such situation, there are two methods to know the workers' quality. First, a worker can learn her quality based on the knowledge of her characteristics, such as her location[2]. Second, the requester can learn workers' quality based on the data collected from them. In a dynamic setting of crowdsourcing where tasks are assigned to and performed by workers sequentially (e.g., in spectrum crowdsensing, tasks can arrive over time, where each task is to measure signals in a particular time slot), the requester can carry out quality learning on the fly, while making use of the learned quality information to perform task assignment and data aggregation. Such online quality learning can improve data accuracy and cost-effectiveness of crowdsourcing and is essential for the practical deployment of crowdsourcing services. However, both methods can expose some issues.

When reporting the quality to the requester, a strategic worker may have incentive to misreport her quality to the requester, in order to benefit. While workers' quality is useful information for fully reaping the benefits of crowdsourcing, it may contain *sensitive* information about the individual workers, which needs to be protected. Moreover, crowdsourcing is vulnerable to *data poisoning attacks*, where an attacker controls malicious workers to report manipulated data to the requester, typically with the goal of reducing the requester's aggregated data accuracy. Due to the random nature of workers' data and unknown ground truths of tasks, it is difficult for the requester to distinguish a malicious worker from a normal worker according to their data.

[2]Alternatively, a worker can report her characteristics (e.g., location) that determines her quality to the requester, so that the requester can learn the worker's quality. In this case, reporting the worker's quality is equivalent to reporting her characteristics.

1.2 Federated Learning in Wireless Network

Federated learning (FL) is an emerging and promising ML framework, which performs training of ML models in a distributed manner. Instead of collecting data from a potentially large number of users to a central server in the cloud for training, FL allows the data to remain at users' end devices (such as smartphones), and trains a global ML model on the server by collecting and aggregating model updates locally computed on each user's device based on her local data. One significant advantage of using FL is to preserve the privacy of individual devices' data. Moreover, since only local ML model updates instead of local data are sent to the server, the communication costs can be greatly reduced. Furthermore, FL can exploit substantial computation capabilities of ubiquitous smart devices, which are often under-utilized. In particular, when FL is used in a wireless edge network, data samples generated at individual wireless devices can be exploited via local computation and global aggregation based on distributed ML. As a result, *wireless federated learning* (WFL) can achieve collaborative intelligence in wireless edge networks. A general consensus is that WFL can support intelligent control and management of wireless communications and networks (such as in [7, 8, 9, 10, 11]), and can enable many AI applications based on wireless networked systems, including connected and autonomous vehicles, collaborative robots, multi-user virtual/mixed reality.

As is standard, learning accuracy is a key performance metric for FL. The accuracy of the trained ML model in FL depends heavily on which devices participate in the training process and the quality of their local model updates. Specifically, stochastic gradient descent (SGD) is a popular method for ML training that is widely studied in the literature (e.g., in [12, 13, 14, 15]). When SGD is used for FL, the quality of a local model update in each iteration can be measured by the variance of the gradient, which depends on the *mini-batch size* used to compute the gradient. A key observation is that the quality of local updates (determined by the mini-batch size) can be treated as a *design parameter* and used as a control knob to be adapted across devices and over time. Such quality-aware computation can substantially improve the learning accuracy of WFL.

1.3 Overview of the Dissertation

In this dissertation, we aim to investigate quality-aware data crowdsourcing and federated learning in wireless networks. For quality-aware data crowdsourcing, we study truthful incentive mechanisms that elicit workers' private information with privacy-preserving property, and data poisoning attacks on dynamic crowdsourcing. For WFL, we study the performance bounds on the training loss as a function of local updates' quality over the training process for different FL setting, and develop cost-effective dynamic distributed learning algorithms that adaptively select participating users and their mini-batch sizes, based on users' costs in wireless networks. Given the commonality of data crowdsourcing and labeling quality of the training data of FL, we study the impact of the strategic behavior of FL clients and develop incentive mechanisms that incentivize strategic clients to make truthful efforts as desired by the server in local data labeling and local model computation, and also report true local models to the server.

In the first part, based on a general linear regression model of machine learning, we devise truthful quality-aware crowdsensing mechanisms for quality and effort elicitation, which incentivize workers to truthfully report their private worker quality to the requester, and make effort as desired by the requester. The truthful design of the mechanisms overcomes the differences of ground truths of workers' tasks, and the coupling in the joint elicitation of workers' quality, effort, and data. Under the mechanisms, we investigated the socially optimal and the requester's optimal effort assignments, and analyze their performance. We show that the requester's optimal assignment is determined by the "virtual quality" rather than the highest quality among workers, which depends on the worker's quality and the quality's distribution.

In the second part, we study Privacy-preserving crowdsourcing mechanisms for truthful Data Quality Elicitation (PDQE). In these mechanisms, we design differentially private task allocation and data aggregation algorithms to prevent the inference of a worker's quality and data from the outcomes of these algorithms. In the meantime, the mechanisms also incentivize workers to truthfully report their quality and data and make desired efforts. We first focus on the mechanisms for a single task (S-PDQE) and then extend it to the case of multiple tasks

(M-PDQE). We further show that both the mechanisms achieve a bounded performance gap compared to the optimal strategy.

In the third part, we study malicious data attacks on dynamic crowdsourcing where tasks are assigned and performed sequentially, and we explore online quality learning as a defense mechanism against the attack by finding malicious workers with low quality. We first focus on the asymptotic setting where workers' quality is accurately learned by the requester, based on which we then turn to the general non-asymptotic setting where the quality is estimated online with errors. For each setting, we first characterize the conditions under which the attack strategy can effectively reduce the aggregated data accuracy. Our results show that the malicious noise variance needs to be within a certain range for the attack to be effective. Then we analyze the harm of effective attack strategies. It reveals that the regret of the online quality learning algorithm can be substantially increased from $\mathcal{O}(\log^2 T)$ (upper bound) to $\Omega(T)$ (lower bound) due to effective attacks. To further mitigate the attack, we also study median and maximum influence of estimation based data aggregation as defense mechanisms. Our results provide useful insights on the impacts of data poisoning attacks when online quality learning is used to defend against the attack.

In the fourth part, we study quality-aware distributed computation for FL, which controls the quality of users' local updates via the sampling sizes. We first characterize the dependency of learning accuracy bounds on the quality of users' local updates over the learning process. It reveals that the impacts of local updates' quality on learning accuracy increase with the number of rounds in the learning process. Based on these insights, we develop cost-effective dynamic distributed learning algorithms that adaptively select participating users and their sampling sizes, based on users' communication and computation costs.

In the fifth part, we study quality-aware distributed computation for FL with non-convex problems and asynchronous algorithms. We first characterize performance bounds on the training loss as a function of local updates' quality over the training process, for both non-convex and asynchronous settings. Our findings reveal that the impact of a local update's quality on the training loss 1) increases with the stepsize used for that local update for non-convex learning, and 2) increases when there are more other users' local updates which are coupled with

that local update (depending on the update delays) for asynchronous learning. Based on these useful insights, we design channel-aware adaptive algorithms that determine users' mini-batch sizes over the training process, based on the impacts of local updates' quality on the training loss as well as users' wireless channel conditions (which determine the update delays) and computation costs.

In the sixth part, we study FL with crowdsourced data labeling where the local data of each participating client of FL are labeled manually by the client. We consider the strategic behavior of clients who may not make desired effort in their local data labeling and local model computation (quantified by the mini-batch size used in the stochastic gradient computation), and may misreport their local models to the FL server. We first characterize the performance bounds on the training loss as a function of clients' data labeling effort, local computation effort, and reported local models, which reveal the impacts of these factors on the training loss. With these insights, we devise Labeling and Computation Effort and local Model Elicitation (LCEME) mechanisms which incentivize strategic clients to make truthful efforts as desired by the server in local data labeling and local model computation, and also report true local models to the server. The truthful design of the LCEME mechanism exploits the non-trivial dependence of the training loss on clients' hidden efforts and private local models, and overcomes the intricate coupling in the joint elicitation of clients' efforts and local models. Under the LCEME mechanism, we characterize the server's optimal local computation effort assignments and analyze their performance.

In the seventh part, we discuss some possible future works. One direction is to consider non-convex and asynchronous FL where users have non-IID local data. In this case, the training loss is heavily affected by the data heterogeneity, and the optimal mini-batch size and user selection design can be very different from in the IID data set. Another direction is to study the case when users perform heterogeneous numbers of local iterations. In this scenario, the training loss is affected by the additional randomness of multiple local updates. Furthermore, when users have non-IID local data, the learned global model may not be consistent with the objective of FL due to heterogeneous numbers of local updates of users.

Chapter 2

Truthful Quality-Aware Data Crowdsensing for Machine Learning

2.1 Introduction

Data crowdsensing (referred to as "crowdsensing" for brevity) leverages the "wisdom" of a potentially large crowd of workers who provide data in tasks that specified by the requester. The applications are enabled by smart devices equipped with powerful sensing, networking, and computing capabilities. The scope of these applications is expected to expand rapidly with the emerging Internet of Things (IoT). A key advantage of crowdsensing lies in that it can exploit the diversity of inherently inaccurate data from many workers by aggregating the data obtained by the crowd, such that the data accuracy (also referred to as "data quality") after aggregation can be substantially enhanced. It has found a wide range of applications such as spectrum sensing [16, 1], traffic monitoring [2, 17], and environmental monitoring [18, 19, 3, 20].

Another major driving force for the popularity of crowdsensing is the recent success of machine learning in various domains of data analytics. The massive amount of data in the era of big data can be exploited by machine learning, which can serve as the foundation to build artificial intelligence (AI). The power of machine learning typically relies on the availability of training data, which can be collected from crowdsensing. One important application of crowdsensing for machine learning is wireless indoor localization [21], which uses wireless signals measured by mobile users as the training data.

The value of data collected in crowdsourcing heavily depends on the *quality* of data provided by the participating workers. A common metric of data quality is how accurate the data

is compared to the ground truth we aim to estimate (e.g., the difference between the data and ground truth). The data provided by workers are usually inaccurate due to various factors (e.g., noise, interference, error). In general, the quality of data *varies* for different workers, and depends on a specific worker's characteristics or context (e.g., sensor's capability, or location). For example, if the task is to measure the transmit signal from a wireless device, then the signal to noise ratio (SNR) received by a worker from that device determines the worker's data quality, and workers generally have distinct SNRs depending on their locations.

To exploit the potential of crowdsourcing, it is imperative for the crowdsourcing requester to *know* workers' data quality. First of all, the quality of data is important *context* information to determine *how to use the data*. For example, when data provided by different workers for the same task are aggregated into one piece of information, it is beneficial to give larger weights to data of higher accuracy in the data aggregation. For another instance, the data can be used as training data for machine learning if the data accuracy exceeds some threshold, or otherwise it will not be used. Moreover, the information of data quality can be exploited to determine *which worker(s) to use*. Intuitively, it is beneficial to allocate a task to worker(s) with higher rather than lower data quality. Therefore, *quality-aware crowdsourcing* can substantially improve the *value* and *usefulness* of data in crowdsourcing.

A worker can learn its quality based on the knowledge of its characteristics, such as its location[1]. However, the quality of a worker's can be its private information, which is unknown to and cannot be verified by the crowdsourcing requester. For example, a worker's location is often its private information that is unknown to the requester. As a result, a strategic worker may have incentive to manipulate its quality revealed to the requester so as to gain an advantage.

In addition to the worker quality, the data quality of a worker is also affected by its effort exerted in a crowdsourcing task. The data quality of a worker when it makes effort in the task is higher than when it makes no effort. Due to the inaccurate nature of the data, a strategic worker may report some arbitrary data to the requester without making effort in the task, while the requester is not able to verify whether effort was actually made.

[1]Alternatively, a worker can report its characteristics (e.g., location) that determines its quality to the requester, so that the requester can learn the worker's quality. In this case, reporting the worker's quality is equivalent to reporting its characteristics.

When strategic workers with private worker quality and hidden effort, our goal is to incentivize workers to truthfully reveal their worker quality, and make actual effort as desired by the crowdsourcing requester. Such a truthful mechanism is desirable as it eliminates the possibility of manipulation, which would encourage workers to participate in crowdsourcing. More importantly, the joint truthful elicitation of quality and effort ensures that the requester can *correctly know the data accuracy of the collected data*, which is a key metric of crowdsourcing.

The joint elicitation of quality and effort calls for new truthful design that is different from existing mechanisms. First, a worker's payoff as a function of its quality and effort has a different structure from that of its private participating cost. As a result, existing designs for cost elicitation cannot work for the problem here. Second, as workers perform tasks with different and unknown ground truths, it is highly non-trivial to design a worker's reward function to achieve truthful elicitation. Third, the joint elicitation of quality and effort needs to overcome the intricate coupling therein.

The main contribution of this chapter can be summarized as follows.

- Under a quality-aware crowdsourcing framework for a general linear regression model, we devise truthful crowdsourcing mechanisms for quality and effort elicitation. The truthful mechanisms incentivize workers to truthfully report their private quality and make effort as desired by the crowdsourcing requester. The truthful design of the mechanisms is achieved by exploiting the intricate correlations between different ground truths of workers' tasks, and overcoming the non-trivial coupling in the joint elicitation of quality and effort.

- Under the truthful mechanisms, we characterize the socially optimal (SO) and the requester's optimal (RO) effort assignments, and analyze their performance. We show that the requester's optimal assignment is determined by the "virtual quality" rather than the highest quality among workers, which depends on the worker's quality and the quality's distribution.

- We provide simulation results which demonstrate the truthfulness of the mechanisms and the performance of the RO and SO effort assignments.

The rest of this chapter is organized as follows. Section 2.2 discusses related work. In section 2.3, we describe the machine learning linear regression model with private data quality, which is the system model for crowdsensing, and formulate the problems of truthful mechanism design. In Section 2.4, we proposed the mechanisms to elicit truthful quality and effort from workers working on different tasks. In Section 2.5, we give the optimal effort assignment under the mechanisms for single and multiple worker situations. Simulation results are presented in Section 2.6. Section 2.7 concludes this chapter and discusses future work.

2.2 Related Works

Truthful mechanisms for data crowdsoucing. In crowdsensing, a worker's cost on its task can be a private information that the worker may not want to report it truthfully. There are many mechanisms that aim to incentivize workers to reveal their costs in crowdsensing [22, 23]. However, the quality of a worker can also be a private information that needs to be elicited. As a function of the worker's quality, a worker's payoff is structurally different from that of its private cost. In addition, strategic workers can take hidden actions that are not desired by a requester. Some recent studies have investigated this problem in the context of crowdsourcing [24, 25, 26, 27, 28]. These works proposed mechanisms that can incentivize workers to truthfully make effort, but none of them considered workers' data quality elicitation.[28] used peer prediction based on machine learning to incentivize workers to report their observed data without knowing the distribution of worker's data quality. [24] aims to jointly elicit workers' desired efforts and true quality. Due to the intricate coupling between the elicitation of quality and effort, it proposed mechanisms to incentivize workers to truthfully report their quality and make efforts as desired by the requester. However, [24] assumes that workers work on the same task, which often does not hold in the context of machine learning as usually different tasks are assigned to workers. In this chapter, we assign different tasks to workers, and design truthful mechanisms to elicit private quality and hidden efforts from strategic workers.

Quality-aware data crowdsoucing. The requester relies on workers' quality to assign tasks to the workers. However, a worker's quality can be its private information that the worker does not want to truthfully report. There are few works that study the quality-aware

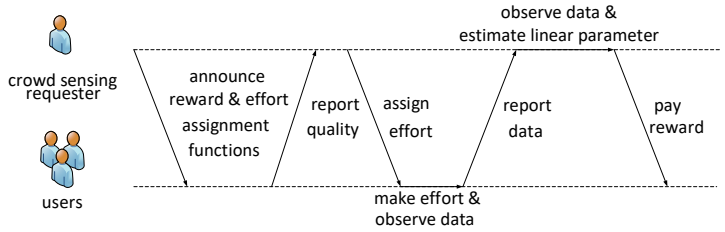

Figure 2.1: Structure and procedure of the crowdsensing system.

crowdsensing. Some works have been done to learn the quality information of workers' data [24, 29, 30, 31, 32]. [29, 30] used the correlation of data for the same task to study the quality information of workers. [31, 32] used online approach to learn workers' abilities and assign tasks. So that the requester can find the correct solution to a set of multiple labeling tasks. [24] focused on designing truthful mechanisms to elicit workers' private quality, where workers perform on the same task. The mechanisms proposed in this chapter consider workers that are assigned different tasks in the context of machine learning and elicit the private quality from them.

2.3 System Model and Problem Formulation

A crowdsensing requester recruits a set of workers $\mathcal{N} \triangleq 1, ..., N$ to work on a set of different tasks. For convenience, let $\mathcal{N}^+ \triangleq \mathcal{N} \cup 0$, where worker 0 represents the requester. Fig. 2.1 illustrates the structure and procedure of the crowdsensing system.

2.3.1 Crowdsensing with private worker quality

Label observation, worker quality, and work effort

The workers work on the tasks that were assigned by the requester. We consider continuous-valued feature $X = \{x_i | x_i \in \mathbb{R}^m, i \in \mathcal{N}^+\}$, which is known to the requester, where x_i is a $m \times 1$ column vector, which means that each worker i works on m tasks. Worker i receives the feature x_i. Every feature x_i follows an arbitrary prior distribution. Each worker observes label $Y = \{y_i | y_i \in \mathbb{R}, i \in \mathcal{N}^+\}$ corresponding to the feature X. Here we use linear regression model as our data model, we assume the relationship of Y and X is expressed as a linear function.

The reason why we use linear model is that linear regression model of machine learning is one of the fundamental supervised machine learning algorithms due to its relative simplicity and well-known properties[33]. In this chapter we focus on the linear regression model, and we will study other models of machine learning in our future work. The objective of machine learning is to infer and estimate the linear relationship between Y and X, which can be expressed with the linear parameter $a \in \mathbb{R}^m$. Each task has a corresponding linear parameter. Which means a is a $1 \times m$ row vector. Furthermore, the observed label Y also contains independent additive noises, which consists of a system noise and an individual worker's observation noise [33]. Observed label Y can be expressed as

$$y_i \triangleq a x_i + \epsilon + n_i, \tag{2.1}$$

where for each task

$$\epsilon \sim \mathcal{N}(0, \sigma), \tag{2.2}$$

and

$$n_i \sim \mathcal{N}(0, \frac{q_i}{e_i}). \tag{2.3}$$

We can see the value of Y is affected not only by the linear parameter a, but also the variance of additive noises. We assume the means of the two noises are 0 without loss of generality (WLOG). The system noise ϵ is the intrinsic disturbance of the system (e.g., due to fading of wireless channels). We assume the variance of each worker's system noise is the same. Besides the system noise, the variance of each individual worker's noise is equal to the ratio of the worker's quality q_i and effort e_i, which is different for different workers. The variance of the additive noise is an unknown information for the requester. However, it can be calculated with information reported by workers, we will discuss this later.

Given worker i's effort e_i, the quality $q_i > 0$ is a parameter that quantifies the accuracy of label y_i. The quality q_i is an intrinsic coefficient that captures worker i's capability for the task. A smaller q_i means higher quality. The value of q_i varies for different workers. We assume that every worker $i \in \mathcal{N}^+$ in the system knows its own quality, and the requester does not know it.

For ease of exposition, we assume that each worker's quality is within the range of $[\underline{q}, \bar{q}]$, which is known to the requester. Given worker i's quality q_i, the effort $e_i > 0$ is the parameter that quantifies the amount of work that worker i devoted to the task, a higher e_i implies the worker is making more effort for the task. e_i is inversely proportional to the noise, which means that when the worker devotes more to the task, the accuracy of y_i is higher. We assume that every worker can fully control the amount of effort they make, and the requester can not know it.

Quality reporting and effort assignment.

We assume the crowdsensing tasks are assigned arbitrarily to the workers. The requester assigns an effort e_i' that it desires worker i to exert in the task, based on the quality of all the workers. To this end, each worker reports its own quality q_i' to the requester, which may not be equal to its actual quality q_i, because the worker can manipulate it to its own advantage.[2] After the requester knows all workers' reported quality, based on a specific effort assignment function, the requester will assign effort to the workers. The effort that assigned to worker i can be expressed as

$$e_i'(q'), \tag{2.4}$$

Then the requester notifies the value of assigned effort e_i' to the worker i. The requester pre-defines and announces the effort assignment function to all the workers before workers report their qualities. The effort assigned to worker i is determined not only by its own reported quality q_i', but also by all the other workers' quality. Although each worker's assigned effort is related to all workers' reported quality, the assigned effort e_i' varies for different workers. After the requester has assigned the effort, the workers work on the tasks. However, workers can decide how much effort to make, which can be different with the amount of assigned effort. Then each worker i will collect label y_i, and report it to the requester truthfully.

Data estimation for linear regression and reward payment.

The objective of linear regression is to estimate the linear parameter a, which quantifies the linear relationship between Y and X. For the convenience of notation, in this section, we

[2]Alternatively, each worker can report its relevant private information (e.g., device model, location) that determines its worker quality to the requester, based on which the requester can learn the worker's quality (e.g., using history data of the device model, or a channel model based on the device's location).

assume that $m = 1$, i.e., every worker works on one individual task. After workers report their information, the requester obtains an estimation value $\hat{a}$ of the interested parameter a. WLOG, we assume $a \sim \mathcal{N}(0, 1)$ [34] (Theorem 10.3). We express the parameters in vectors as

$$Y = [y_1, y_2, ..., y_N]^{\mathbf{T}}, \tag{2.5}$$

$$X = [x_1, x_2, ..., x_N]^{\mathbf{T}}. \tag{2.6}$$

The additive noise (the sum of system noise and individual worker's noise) and its variance are

$$W = [\epsilon + n_1, \epsilon + n_2, ..., \epsilon + n_N]^{\mathbf{T}}, \tag{2.7}$$

$$C = [\sigma + \frac{q_1}{e_1}, \sigma + \frac{q_2}{e_2}, ..., \sigma + \frac{q_N}{e_N}]^{\mathbf{T}}. \tag{2.8}$$

Then the label can be expressed as

$$Y = aX + W. \tag{2.9}$$

The estimation of the linear parameter a can be expressed as [34]

$$\begin{aligned} \hat{a} &= \mu_a + \sigma_a X^{\mathbf{T}}(X\sigma_a X^{\mathbf{T}} + C)^{-1}(Y - X\mu_a) \\ &= X^{\mathbf{T}}(XX^{\mathbf{T}} + C)^{-1}Y. \end{aligned} \tag{2.10}$$

The estimation loss is quantified by the minimum mean square error (MMSE) of the linear parameter a, which quantifies the utility of crowdsensing:

$$\begin{aligned} l(X, q, e, \sigma) &= E[(\hat{a} - a)^2] \\ &= \sigma_a - \sigma_a X^{\mathbf{T}}(X\sigma_a X^{\mathbf{T}} + C)^{-1} X\sigma_a \\ &= \frac{1}{1 + \sum_{i \in \mathcal{N}} \frac{x_i^2}{\sigma + \frac{q_i}{e_i}}}. \end{aligned} \tag{2.11}$$

As we can see in (2.11), the estimation loss relies not only on all workers' quality and effort, but also on the value of the tasks.

We assume any worker j ($j \neq i$) always truthfully reports quality and makes effort as the requester desired. For convenience, here we use worker 1 and worker 2's information. To incentivize workers to truthfully behave, the requester rewards workers based on a function of the reported quality and assigned effort, i.e.,

$$r_i(x_i, y_i, x_1, y_1, x_2, y_2, \frac{q_1}{e_1}, \frac{q_2}{e_2}, {e_i}', q').$$

The reward function is also pre-defined by the requester and announced to all the workers before they report their qualities. The reward function only depends on the information that is known by the requester, i.e., X, Y, e_i', q', q_1/e_1, q_2/e_2. For convenience, we omit worker 1 and worker 2's information in the expression of the reward function, then the reward function is expressed as

$$r_i(x_i, y_i, e_i', q'). \tag{2.12}$$

2.3.2 Mechanism design objective

After the workers have gone through the process we discussed above, they will be paid by the requester. worker i's payoff u_i can be expressed as

$$u_i(x_i, y_i, e_i', q') = r_i(x_i, y_i, e_i', q') - c_i e_i, \tag{2.13}$$

which is the difference of the reward paid by the requester and the cost of working on the task.

Here c_i is the resource (e.g., sensing time, energy) that is used by worker i for each unit of effort they devoted to the task. The cost $c_i e_i$ is a linear function of the effort e_i, which represents the total resource that has been devoted to the crowdsensing task by worker i. In this chapter, we assume that workers' devices have similar capability to work on the tasks, and then the cost for a unit of effort can be considered the same for all workers, i.e., $c = c_i$ for $i \in \mathcal{N}^+$. The requester knows the cost coefficient c. This is a reasonable assumption, when workers' smart

devices are working under uniform (or similar) conditions, workers are likely have the same capability working on the tasks, so that the cost for a unit of effort is common knowledge to the requester and workers.

After the workers work on the tasks, the requester obtain it's payoff. The requester's payoff u_0 can be expressed as

$$u_0(X, Y, e', q', e, q, \sigma) = -l(X, q, e, \sigma) - \sum_{i \in \mathcal{N}} r_i(x_i, y_i, {e_i}', q'). \tag{2.14}$$

which is the difference of the crowdsensing's utility and the total reward paid to the workers.

Workers may behave untruthfully for their own advantage, e.g., workers may report untrue quality and/or making less effort to get more reward from the requester. This act will not only reduce the requester's payoff, but also affect the utility of crowdsensing, as it can lead to inaccurate estimation of the linear parameter a, which is a critical performance metric that needs to be ensured in machine learning. Also, workers' manipulation would discourage other workers to participate in crowdsensing. For the reasons we discussed above, here we aim to design a mechanism that can incentivize workers to report true quality and make effort as the requester demanded. This can be achieved by defining the effort assignment function $e_i'(q')$ and reward function $r_i(x_i, y_i, e_i', q')$. The mechanism should have the following two features:

Definition 2.1 A mechanism achieves truthful strategies of all workers as a *Nash equilibrium* (NE) if, given other workers truthfully report their quality and make effort as the requester assigned, the best strategy for worker i to maximize its payoff is to truthfully report its quality and make effort as assigned by the requester, i.e.,

$$E_{X,Y}[u_i(x_i, y_i, e_i', q_i, q_{-i})] \geq E_{X,Y}[u_i(x_i, y_i, e_i, q_i', q_{-i})], \forall (q_i', e_i), \forall q_{-i}. \tag{2.15}$$

Another aspect we should notice is that the payoff of every worker u_i should be non-negative, so that the worker will have the incentive to participate in more crowdsensing tasks. This property is formally known as individual rationality as stated below.

Definition 2.2 A mechanism is *individually rational* (IR) if for each worker i, its expected payoff is non-negative if it truthfully report its quality and make effort as the requester assigned, i.e.,

$$E_{X,Y}[u_i(x_i, y_i, e'_i, q_i, q'_{-i})] \geq 0, \forall q'_{-i}. \tag{2.16}$$

2.4 Truthful Quality and Effort Elicitation for Crowdsensing

In this section, we aim to design mechanisms that satisfy the truthful and IR properties to incentivize workers to report true quality and make effort as the requester desired.

Here we present the mechanisms as follows.

Definition 2.3 The mechanisms are defined by any effort assignment function $e'_i(q')$ that satisfies (2.17) and a reward function $r_i(x_i, y_i, e'_i, q')$ given by (2.18) based on that $e'_i(q')$:

$$e'_i(q'_i, q'_{-i}) \geq e'_i(q''_i, q'_{-i}), q'_i \leq q''_i, \tag{2.17}$$

$$\begin{aligned} r_i(x_i, y_i, e'_i, q') = c\int_{q'_i}^{\bar{q}} \frac{e'_i(q, q'_i)}{q} dq + 2ce'_i(q') - \frac{ce'^2_i(q')}{q'_i}\{x_i^2[E(\frac{y_2}{x_2} - \frac{y_i}{x_i})^2 - \frac{\bar{q_2}}{x_2^2}] \\ - (\frac{x_i^2}{x_2^2} + 1)\frac{E(\frac{y_1}{x_1} - \frac{y_2}{x_2})^2 - \frac{\bar{q_1}}{x_1^2} - \frac{\bar{q_2}}{x_2^2}}{\frac{1}{x_1^2} + \frac{1}{x_2^2}}\}. \end{aligned} \tag{2.18}$$

It is defined in (2.17) that if the quality of worker i improves, the worker will be assigned more effort, given any quality of other workers. Next we will show how the mechanisms achieve the truthful property. First, we show that the expected payoff of each worker depends on its true quality and effort (Lemma 2.1). Second, we show that if the elicitation of true quality is achieved, the elicitation of effort is also achieved (Lemma 2.2). Then we show that when the worker makes optimal effort, the elicitation of its quality is also achieved (Lemma 2.3).

First we show that the expected payoff of worker i is a function of its true quality and actual effort (2.18).

Lemma 2.1 Given that worker i works on its task and reports its feature x_i and label y_i, and that it reports its quality q_i' and makes effort e_i, we can express its expected payoff as

$$E_{X,Y}[u_i(x_i, y_i, e_i, q')] = c\int_{q_i'}^{\bar{q}} \frac{e_i'(q, q_i')}{q} dq + 2ce_i'(q') - \frac{ce_i'^2(q')}{q_i'}\frac{q_i}{e_i} - ce_i. \tag{2.19}$$

The variances of the additive noises (σ and q_i/e_i) are unknown to the requester. However, they can be calculated by the information that is known by the requester.

When $m = 1$,

$$\frac{y_1}{x_1} - \frac{y_2}{x_2} = (\frac{\epsilon}{x_1} + \frac{n_1}{x_1}) - (\frac{\epsilon}{x_2} + \frac{n_2}{x_2}). \tag{2.20}$$

Let$\bar{q_i} = \frac{q_i}{e_i}$, then we have

$$E(\frac{y_1}{x_1} - \frac{y_2}{x_2})^2 = \frac{\sigma}{{x_1}^2} + \frac{\bar{q_1}}{{x_1}^2} + \frac{\sigma}{{x_2}^2} + \frac{\bar{q_2}}{{x_2}^2},$$
$$E(\frac{y_1}{x_1} - \frac{y_i}{x_i})^2 = \frac{\sigma}{{x_1}^2} + \frac{\bar{q_1}}{{x_1}^2} + \frac{\sigma}{{x_i}^2} + \frac{\bar{q_i}}{{x_i}^2}.$$

Then we can express σ and $\bar{q_i}$ as

$$\sigma = \frac{E(\frac{y_1}{x_1} - \frac{y_2}{x_2})^2 - \frac{\bar{q_1}}{{x_1}^2} - \frac{\bar{q_2}}{{x_2}^2}}{\frac{1}{{x_1}^2} + \frac{1}{{x_2}^2}}, \tag{2.21}$$

$$\begin{aligned}\bar{q_i} &= \frac{q_i}{e_i} \\ &= {x_i}^2[E(\frac{y_1}{x_1} - \frac{y_i}{x_i})^2 - \frac{\sigma}{{x_1}^2} - \frac{\bar{q_1}}{{x_1}^2} - \frac{\sigma}{{x_i}^2}] \\ &= {x_i}^2[E(\frac{y_2}{x_2} - \frac{y_i}{x_i})^2 - \frac{\bar{q_2}}{{x_2}^2}] - (\frac{{x_i}^2}{{x_2}^2} + 1)\frac{E(\frac{y_1}{x_1} - \frac{y_2}{x_2})^2 - \frac{\bar{q_1}}{{x_1}^2} - \frac{\bar{q_2}}{{x_2}^2}}{\frac{1}{{x_1}^2} + \frac{1}{{x_2}^2}}.\end{aligned} \tag{2.22}$$

When $m > 1$, let

$$X_k = [x_{(k-1)m+1}, x_{(k-1)m+2}, ..., x_{km}],$$

$$Y_k = [y_{(k-1)m+1}, y_{(k-1)m+2}, ..., y_{km}],$$

$$N_k = [n_{(k-1)m+1}, n_{(k-1)m+2}, ..., n_{km}],$$

where X_k is a $m \times m$ dimension matrix, $k = 1, 2, \ldots$. Y_k and N_k are $1 \times m$ row vectors. Then we have

$$\begin{aligned} Y_k X_k^{-1} &= a X_k X_k^{-1} + N_k X_k^{-1} + [\epsilon, \epsilon, ..., \epsilon] X_k^{-1} \\ &= a + N_k X_k^{-1} + [\epsilon, \epsilon, ..., \epsilon] X_k^{-1}. \end{aligned}$$

Using worker 1 and worker 2's information we have

$$Y_1 X_1^{-1} - Y_2 X_2^{-1} = N_1 X_1^{-1} - N_2 X_2^{-1} + [\epsilon, \epsilon, ..., \epsilon](X_1^{-1} - X_2^{-1}). \tag{2.23}$$

Then we can calculate the variances of workers' noises, σ and q_i/e_i. Based on the information that is known by the requester, the expected payoff of worker i is a function of its true quality and actual effort, and the worker can only affect it by its reported quality and actual effort.

Lemma 2.2 The elicitation of true quality leads to the elicitation of effort. Given that worker i reports quality q_i', its optimal effort can be expressed as

$$e_i = \sqrt{\frac{q_i}{q_i'}} e_i'(q'). \tag{2.24}$$

Note that when worker i reports its true quality, its optimal effort is equal to the effort that assigned by the requester. This shows that if worker i wants to maximize its payoff when it reports its true quality, it needs to make effort as the requester assigned. Then the expected payoff of worker i is

$$c \int_{q_i'}^{\bar{q}} \frac{e_i'(q, q_i')}{q} dq + 2c e_i'(q') - 2c \sqrt{\frac{q_i}{q_i'}} e_i'(q'). \tag{2.25}$$

In (2.25) we can see that worker i can only affect its payoff by its reported quality q_i'. In the next lemma, we will show that worker i's optimal reported quality q_i' is its true quality, under the condition (2.17) on the effort assignment function [24].

Lemma 2.3 Given that worker i reports its feature x_i, label y_i, and makes its optimal effort e_i as in (2.24), its optimal reported quality q_i' is its true quality q_i.

As we discussed in Lemmas 1, 2, and 3, when worker i reports its true quality $q_i' = q_i$, and makes the optimal effort $e_i = e_i'$, its expected payoff can be expressed as

$$c \int_{q_i}^{\bar{q}} \frac{e_i'(q, q'_{-i})}{q} dq. \tag{2.26}$$

The equation satisfies the IR property, since (2.26) is nonnegative due to the fact that $e_i'(q') \geq 0, \forall q'$ [24].

Theorem 2.1 The mechanisms satisfy truthful and IR properties.

Property 2.1 Here we discuss the rationale of the mechanisms. The requester's aim is to incentivize workers to report true quality q and make actual effort e as the requester assigned. Thus, worker i's reward function r_i should be a function of its true quality q_i and actual effort e_i. Otherwise, workers will deceive the requester to gain more reward, and the efficiency of the crowdsensing system will be affected. However, the requester can only use the information which known by itself (i.e., X, Y, q', e_i') to define the reward function. Using (2.22), we can design the reward function as a function of the true quality q_i and actual effort e_i (as in Lemma 1 and (2.19)). In refined function, worker i's optimal effort equals to its assigned effort e_i' when worker i reports its true quality (as in Lemma 2 and (2.24)). Given that the actual effort is optimized, worker i's payoff only depends on q_i', e_i', and q_i (as in (2.25)). Next we further design the function such that, to make the reward function to be maximized, worker i's reported quality q_i' should equal to its true quality q_i when other parameters are fixed (as in Lemma 3).

Property 2.2 From (2.26) we can see, given the assigned effort $e_i'(q')$ and workers' qualities q, as the upper bound of quality $\bar{q}$ increases, the payoff of worker i increases, and the payoff of the requester decreases. When the lower bound is fixed, a lower upper bound means the range of qualities is smaller. This is a benefit for the requester because the level of uncertainty of workers' quality is lower. Assume that workers have the same quality, i.e., $q_i = \bar{q}, \forall i$, then worker i's payoff would be 0, which means that the mechanism is fully "efficient" for the requester's interest.

2.5 Optimal Effort Assignment for Truthful Crowdsensing

In the above sections, we have shown that the mechanisms can achieve truthful and IR properties. In this section, we will show how the requester assigns efforts to maximize its payoff and system efficiency.

We assume that workers report true qualities $q' = q$ and make efforts as the requester assigned $e = e'$. Thus, for convenience, we use q and e instead of q' and e', respectively. We further assume that worker i's task x_i follows a normal prior distribution $\mathcal{N}(0, 1)$, workers' qualities follow independent and identical uniform distributions over the interval $[\underline{q}, \bar{q}]$.

Definition 2.4 The effort assignment function $e(q)$ that can maximize the requester's expected payoff is the *crowdsensing requester's optimal (CO) effort assignment* $e^{co}(q)$, i.e.,

$$\{e^{co}(q), \forall q\} \triangleq \max_{e(q)} E_{X,Y}[u_0(X, Y, e, q, \sigma)]. \tag{2.27}$$

We first consider the case where at most one worker is assigned effort. The advantage of this single-worker assignment is that it simplifies the implementation of crowdsensing: the requester needs to collect data from only one worker rather than potentially many workers. Also, the result of this case provides useful insights for the general case.

Theorem 2.2 For the case of single-worker assignment, the requester's optimal effort assignment is given by

$$e_i^{co} = \max\{\frac{q_i}{\sigma + x_i^2}(x_i\sqrt{\frac{1}{\alpha(q_i)}} - 1), 0\}, \tag{2.28}$$

where

$$\alpha(q_i) = c(\frac{F(q)}{f(q)} + q_i), \forall i \tag{2.29}$$

which is the virtual quality of a worker.

We can see that the virtual quality depends on worker i's quality and the distribution of quality $F(q)$ and $f(q)$. $f(q)$ and $F(q)$ are the probability density function (PDF) and cumulative density function (CDF) of each worker's quality, respectively.

Next we consider the general case where multiple workers can be assigned effort. We can show the following useful fact.

Lemma 2.4 The requester's expected payoff, i.e., $E_{X,Y}[u_0(X, Y, e, q, \sigma)]$, is a convex function of effort e.

Based on Lemma 2.4, for the case of multi-worker assignment, to find the optimal effort assignment that maximizes the requester's payoff, we can use the barrier method to solve the following convex optimization problem:

$$\begin{aligned} &\min \ -E_Q[\bar{u_0}(e(Q))] \\ &\text{subject to } e_i \geq 0. \end{aligned} \tag{2.30}$$

Next we study the social welfare v of the system, which is the difference between the crowdsensing's utility and the total cost for all workers, to quantify the system efficiency. The social welfare v can be expressed as

$$v(e(q)) \triangleq -E[l(X, q, e, \sigma)] - \sum_{i \in \mathcal{N}} c_i e_i. \tag{2.31}$$

Definition 2.5 The effort function $e(q)$ that maximizes the social welfare is the *socially optimal (SO) effort assignment* $e^{so}(q)$, i.e.,

$$\{e^{so}(q), \forall q\} \triangleq \max_{e(q)} v(e(q)). \tag{2.32}$$

We first have the following result for the case of single-worker assignment.

Theorem 2.3 For the case of single-worker assignment, the socially optimal effort assignment is given by

$$e_i^{so} = \max\{\frac{q_i}{\sigma + x_i^2}(\frac{x_i}{\sqrt{cq_i}} - 1), 0\}. \tag{2.33}$$

As we can see in (2.28) and (2.33), given the workers' quality, the variance of system noise, and the probability distribution of quality, the optimal effort assignment is determined

by the feature x_i of the task, which is determined before the requester assigns the effort. This is because the requester's expected payoff and social welfare both depend on the estimation loss l, which is a function of the task's feature.

We can show the following fact for the general case of multi-worker assignment.

Lemma 2.5 The social welfare v, i.e., $v(e(q))$, is a convex function of e and q.

Similar to the requester's optimal effort assignment, to obtain the optimal effort assignment that maximizes social welfare, it is equivalent to solve the following convex optimization problem:

$$\begin{aligned} &\min \ -E[v(e(q))] \\ &\text{subject to } e_i \geq 0. \end{aligned} \tag{2.34}$$

2.6 Simulation Results

In this section, we prove the truthful and IR properties have been achieved, and discuss the performance of the optimal effort assignments under different conditions.

2.6.1 Worker's payoff

To demonstrate the truthful and IR properties, we compare a worker's expected payoff when it reports its true quality and makes actual effort with that when it reports quality untruthfully and/or does not make effort as the requester assigned. We use the CO effort assignment $e_i^*(q)$ for the mechanisms and set the system parameters as follows: $N = 1$, $m = 1$, $c = 0.3$, $q_i \in [0.5, 3.5]$, $q_1 = 1.2$.

Fig.2.2 illustrates the worker's expected payoff when the worker makes assigned effort $e_1^*(q_1')$, or optimal effort $\sqrt{\frac{q_1}{q_1'}}e_1^*(q_1')$, as its reported quality q' varies. By making either assigned effort or optimal effort, the expected payoff of the worker can reach the maximum value, which is the payoff when the worker reports true quality and makes effort as the requester assigned simultaneously. When the worker untruthfully reports its quality, the payoff of the worker is always no greater than that when it reports its true quality. When it reports more untruthfully,

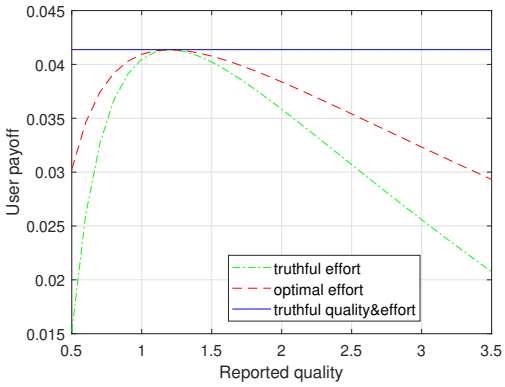

Figure 2.2: Impact of reported quality q_1'

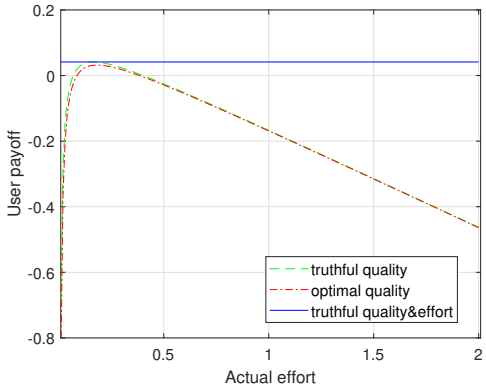

Figure 2.3: Impact of actual effort e_1'

the worker's expected payoff keeps decreasing. We can see that the truthful property has been achieved by the mechanisms so that workers have the incentive to behave truthfully.

Fig.2.3 shows that when the worker reports its true quality or the optimal quality, the worker's expected payoff varies with the actual effort the worker made. We can see when the worker makes effort as the requester assigned, its expected payoff is maximized. Besides, when the worker truthfully reports its quality, its expected payoff is always higher than that when it reports the highest quality. Also, from fig.2.2, and fig.2.3 we can see that the payoff is non-negative when the worker truthfully reports its quality and makes the effort desired by the requester, which confirms that the IR property is achieved by the mechanisms.

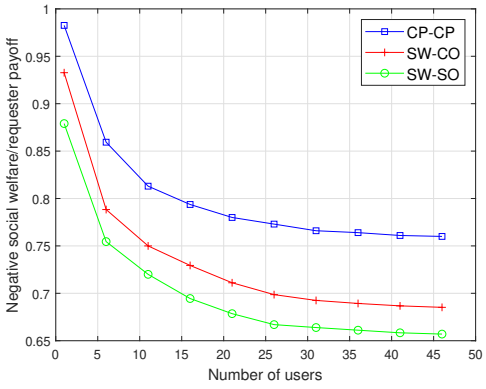

Figure 2.4: Impact of the number of workers N

2.6.2 Requester's payoff

To prove the system efficiency of the CO effort assignment, we compare the expected requester's payoff (CP) attained by the CO effort assignment (CP-CO) with the expected social welfare (SW) attained by the SO effort assignment (SW-SO), and the expected SW attained by the CO effort assignment (SW-CO). We set the system parameters as follows: $N \in [1, 50]$, $m = 1$, $c = 0.5$, $q_i \in [0.5, 3.5]$. For convenience, we illustrate the negative of social welfare or the requester's payoff in all figures.

Fig.2.4 gives the tendency of CP-CO, SW-CO, and SW-SO when the number of workers varies. We can see that with the increase of the number of workers, all three curves are decreasing. It is because that when there are more workers working on the tasks, the estimation loss will decrease. After the number of workers reaches a certain number, the trend of the curves becomes gentle, which means that it may not be the more workers the better, since more workers means more occupation of social resource.

Fig.2.5 illustrates the impact of the cost on each unit of effort c on the performance of CP-CO, SW-CO, and SW-SO. As shown in the figure, all three values increase as the cost c increases. When the cost c is small, the gap between social welfares is small, and it increases with c grows. This is because when the cost is large, tasks are assigned to the workers whom have better qualities.

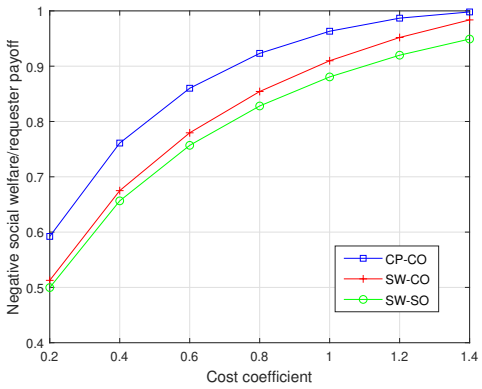

Figure 2.5: Impact of cost c

2.7 Conclusion

In this chapter, to provide high quality data for machine learning, we devised the truthful mechanisms to elicit private quality from strategic workers that work on different tasks and to incentivize workers to make efforts as the requester desires. Under the mechanisms, we investigated the optimal effort assignments that maximize the requester's payoff and social welfare, which are the functions of the linear parameter's estimation error.

2.8 Appendix

2.8.1 Proof of Lemma 2.1

We have shown in (2.22) and (2.23) that the unknown ratio of quality q_i and effort e_i can be expressed by the information known by the requester (feature X, label Y, q_1/e_1, q_2/e_2) no matter how many tasks worker i works on.

2.8.2 Proof of Theorem 2.1

The IR property has been proved by (2.26), from Lemma 2 and Lemma 3 we can know that given worker j ($j \neq i$) reports its true quality and make effort as the requester assigned, when the worker truthfully reports its quality and makes optimal effort, it can maximize its payoff, then the truthful property has been proved.

2.8.3 Proof of Lemma 2.4

From (2.10) and (2.11) we have the estimation of the linear parameter a and the expression of the estimation loss $l(X, q, e, \sigma)$. When worker i reports its true quality and makes actual effort, its reward can be expressed by (2.26), then we have the expected requester's payoff, which is given by

$$\begin{aligned} & E[u_0(X, y_i, e', q', e, q, \sigma)] \\ & = -l(X, q, e, \sigma) - \sum_{i \in \mathcal{N}} r_i(x_i, y_i, {e_i}', q') \\ & = -\frac{1}{1 + \sum \frac{x_i^2}{\sigma + \frac{q_i}{e_i}}} - \sum_{i \in \mathcal{N}} \alpha(q_i) \frac{e_i}{q_i}. \end{aligned} \tag{2.35}$$

Where

$$\alpha(q_i) = c(\frac{F(q_i)}{f(q_i)} + q_i), \tag{2.36}$$

it is given by [24]. For convenience, we express $E[u_0(X, y_i, e', q', e, q, \sigma)]$ as $E[u_0(e(q))]$.

Given a function $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$, if all the second partial derivatives of f exist and continuous over the domain of f, the Hessian matrix of f can be expressed as

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix} \tag{2.37}$$

The element on jth row and kth column of $E[u_0(e)]$'s hessian matrix $\mathbf{H}$ is given by

$$\begin{aligned} H_{j,k} &= \frac{\partial^2 E[u_0(e(q))]}{\partial e_j \partial e_k} \\ &= \frac{q_j q_k e_j e_k}{(1 + \sum_{i \in \mathcal{N}} \frac{x_i^2}{\sigma + \frac{q_i}{e_i}})^4 (\sigma e_j + q_j)^2 (\sigma e_k + q_k)}, \\ & \qquad\qquad \forall j, k \in \mathcal{N}. \end{aligned} \tag{2.38}$$

By calculation we have $|\mathbf{H}| \geq 0$, the expected requester's payoff function is convex.

2.8.4 Proof of Theorem 2.2

From Lemma 4 we have the expected requester's payoff function is convex. From (2.35) we have the expected requester's payoff can be expressed as

$$E_{X,Q}[\bar{u_0}(e(Q))] = E_{X,Q}[-\frac{1}{1+\sum \frac{x_i^2}{\sigma+\frac{q_i}{e_i}}} - \sum_{i\in\mathcal{N}} \alpha(q_i)\frac{e_i}{q_i}]. \tag{2.39}$$

For single-worker assignment, we find the optimal solution by solving

$$\frac{\partial[E_{X,Q}[\bar{u_0}(e(Q))]]}{\partial e_i} = \frac{\partial[-\frac{1}{1+\frac{x_i^2}{\sigma+\frac{q_i}{e_i}}} - \alpha(q_i)\frac{e_i}{q_i}]}{\partial e_i} = 0, \tag{2.40}$$

which yields

$$e_i^{co} = \max\{\frac{q_i}{\sigma+x_i^2}(x_i\sqrt{\frac{1}{\alpha(q_i)}} - 1), 0\}. \tag{2.41}$$

2.8.5 Proof of Lemma 2.5

Same as Lemma 4, the expected social welfare $E[v(e(q))]$ is given by

$$\begin{aligned} E[v(e(q))] &= -E[l(X,q,e,\sigma)] - \sum_{i\in\mathcal{N}} c_i e_i \\ &= -\frac{1}{1+\sum \frac{x_i^2}{\sigma+\frac{q_i}{e_i}}} - \sum_{i\in\mathcal{N}} c_i e_i. \end{aligned} \tag{2.42}$$

The element on jth row and kth column of $E[v(e(q))]$'s Hessian matrix $\mathbf{H}$ is given by

$$\begin{aligned} H_{j,k} &= \frac{\partial^2 E[v(e(q))]}{\partial e_j \partial e_k} \\ &= \frac{q_j q_k e_j e_k}{(1+\sum_{i\in\mathcal{N}} \frac{x_i^2}{\sigma+\frac{q_i}{e_i}})^4(\sigma e_j + q_j)^2(\sigma e_k + q_k)}, \\ &\qquad\qquad \forall j,k \in \mathcal{N}. \end{aligned} \tag{2.43}$$

By calculation we have $|\mathbf{H}| \geq 0$, the expected social welfare function is convex.

2.8.6 Proof of Theorem 2.3

From Lemma 5 we have the expected social welfare function is convex. From (2.42) we have the expression of the expected social welfare.

For single-worker assignment, we find the optimal solution by solving

$$\frac{\partial[E_{X,Q}[v(e(q))]]}{\partial e_i} = \frac{\partial[-\frac{1}{1+\frac{x_i^2}{\sigma+\frac{q_i}{e_i}}} - ce_i]}{\partial e_i} = 0, \tag{2.44}$$

which yields

$$e_i^{so} = \max\{\frac{q_i}{\sigma + x_i^2}(\frac{x_i}{\sqrt{cq_i}} - 1), 0\}. \tag{2.45}$$

Chapter 3

Privacy-Preserving Incentive Mechanisms for Truthful Data Quality in Data Crowdsourcing.

3.1 Introduction

Data crowdsourcing (referred to as "crowdsourcing" for brevity) has found a wide range of applications. The applications of crowdsourcing can be generally categorized as physical sensing (also known as "crowdsensing") such as spectrum sensing [1], traffic monitoring [2], environmental monitoring [3], and human intelligence such as image labeling and speech transcribing [4, 5]. In principle, crowdsourcing leverages the "wisdom" of a potentially large crowd of workers (e.g., mobile users) for tasks. One main advantage of crowdsourcing lies in that it can exploit the diversity of inherently inaccurate data from many workers by aggregating the data obtained by the crowd, such that the data accuracy (referred to as "data quality") after aggregation can substantially improve. With enormous opportunities and growing popularities of data-driven technologies, crowdsourcing is a promising paradigm to harness the power of big data via machine learning, and enable artificial intelligence in various application domains, such as image classification [4] and indoor localization [6].

To exploit the potential of crowdsourcing, it is beneficial to allocate tasks to workers based on their *quality*. A worker's quality[1] can capture the *intrinsic* accuracy of the worker's data relative to the ground truth of the interested variable, and it generally varies for different workers depending on a worker's characteristics (e.g., location, sensors' capabilities). For example, if the task is to detect whether a licensed wireless device is transmitting or not (for opportunistic spectrum access by unlicensed users), then the quality of a worker's data is the probability of

[1] We use "worker quality" and "quality" exchangeably in this chapter. "Worker quality" is distinguished from "data quality".

correct detection, which depends on the worker's location relative to the licensed device. Workers generally have *diverse* quality. A worker can learn her quality based on the knowledge of her characteristics, such as her location[2] (as in Fig. 3.1).

Although quality-aware crowdsourcing is promising, the quality of a worker can be her *private* information, which is unknown to and cannot be verified by the crowdsourcing requester. For example, a worker's location is often her private information that is unknown to the requester. As a result, a strategic worker may have incentive to misreport her quality to the requester, in order to benefit. For example, a worker of low quality may pretend to have high quality in the hope of receiving a high reward for providing high quality data. In addition to the worker quality, the effort exerted by a worker in the task can also be her hidden action that cannot be observed by the requester. Therefore, a strategic worker may make no effort in the task while the requester is not able to verify whether an effort is made. Furthermore, the data obtained by a worker from performing the task could also be her private information that she can misreport in favor of herself.

On the other hand, while workers' quality is useful information for fully reaping the benefits of crowdsourcing, it may contain *sensitive* information about the individual workers, which needs to be protected. For example, as a worker's location can determine her quality, the location can be inferred from the quality, which can be further used to infer the worker's identity. Although the quality of a worker would not be directly revealed to other workers, it may be inferred from the outcome of the task allocation process, which depends on workers' quality. Moreover, the data reported by a worker can also be her sensitive information that should not be disclosed. As an example, the data obtained by a worker can also depend on her location, which can be inferred from the data. However, even a worker only reports her data to the requester, it may be inferred from the result aggregated from all workers' reported data.

In the presence of strategic workers with private quality and data, and hidden effort, we aim to incentivize workers to truthfully reveal their *quality* and *data*, and make *efforts* as desired by the requester. The joint truthful elicitation of quality, effort, and data ensures that the requester

[2]Alternatively, a worker can report her characteristics (e.g., location) that determines her quality to the requester, so that the requester can learn the worker's quality. In this case, reporting the worker's quality is equivalent to reporting her characteristics.

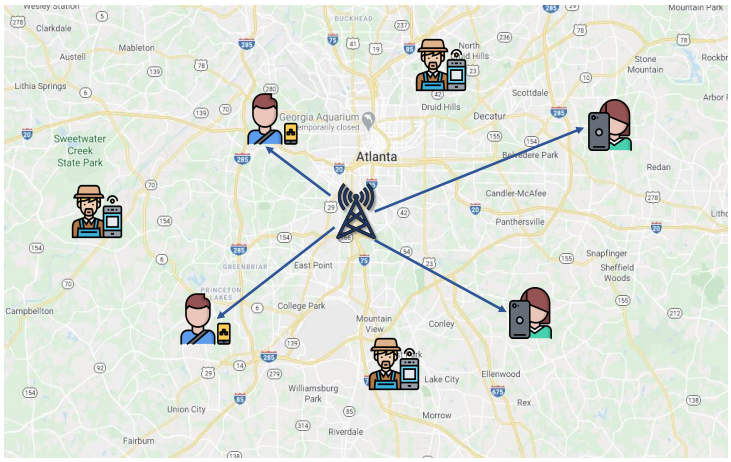

Figure 3.1: In spectrum crowdsensing, a worker's quality for a task depends on her location with respect to the wireless device to be observed, which can be her private information that needs to be protected.

can *know the true accuracy* of the aggregated data, which is an important performance metric of crowdsourcing. More importantly, for privacy-aware workers with sensitive quality and data information, the incentive mechanism should protect a worker's *quality and data* from being inferred by other workers based on the outcomes of the mechanism. This privacy-preserving mechanism resolves workers' privacy concerns and thus encourage their participation.

The design and analysis of the privacy-preserving mechanisms for truthful data quality elicitation need to address challenges that are quite different from existing works. First, to protect the privacy of workers' quality and data, the mechanism needs to randomize the processes of both task allocation and data aggregation. This complication affects the truthful design of the mechanism, and also the efficiency of the mechanism (in terms of the accuracy of aggregated data). Second, due to the stochastic dependency of a worker's private data on her private quality and hidden effort, the joint elicitation of quality, effort, and data needs to overcome the coupling therein.

The main contributions of this chapter can be summarized as follows.

- Under a quality-aware crowdsourcing framework, we devise Single-task and Multi-task Privacy-preserving crowdsourcing mechanisms for truthful Data Quality Elicitation (S-PDQE and M-PDQE). S-PDQE incentivizes strategic workers to truthfully reveal their private quality and data, and make hidden efforts as desired by the crowdsourcing requester when there is a single task to be assigned to workers. M-PDQE achieves the

truthful properties when there are multiple tasks to be assigned to different workers. The truthful design of the mechanisms overcomes the lack of ground truth and the coupling in the joint elicitation of a worker's private quality and data and hidden effort.

- In both mechanisms, we design differentially private task allocation and data aggregation algorithms that prevent the inference of a worker's quality and data from the outcomes of these algorithms. The design of the task allocation algorithm addresses the coupling between the privacy-preserving design and the truthful design. We also provide a bound on the performance gap of the mechanisms in terms of the data accuracy compared to the optimal strategies. To the best of our knowledge, this is the first paper that studies crowdsourcing incentive mechanisms for truthful data quality while protecting workers' quality and data privacy.

- We use simulations based on real-world data to evaluate the performance of M-PDQE and S-PDQE. The results show that both mechanisms achieve the desired truthfulness, individual rationality, and differentially private properties, alongside near-optimal data accuracy.

The remainder of this chapter is organized as follows. Section 3.2 reviews related work. In Section 3.3 and Section 3.4, we describe the system model of quality-aware crowdsourcing and formulate the problem of designing truthful and privacy-preserving mechanisms. In Section 3.5 and 3.6, we introduce the design of S-PDQE and M-PDQE and analyze the properties of the mechanisms in terms of truthfulness, differential privacy, and performance gap with respect to the optimal strategies, respectively. Simulation results are presented in Section 3.7. Section 3.8 concludes this chapter and discusses future work.

3.2 Related Work

3.2.1 Privacy-preserving mechanisms for crowdsourcing

Privacy-preserving mechanisms for crowdsourcing have been studied in many works. While a few works have used cryptographic techniques to protect workers' privacy [35, 36, 37], most

studies have used the concept of differential privacy [38]. Many of these studies have focused on preventing a worker's bid (typically its participating cost) from being inferred by other workers [39, 40, 41, 42]. Some studies have considered protecting workers' data privacy from each other [43, 44]. On the other hand, some works have focused on protecting workers' cost or data privacy from the crowdsourcing requester [45, 46, 42]. Location privacy has also been studied in a few works [47, 48]. Different from these prior works, this paper aims to protect the privacy of workers' quality and data, while achieving truthful data quality elicitation.

3.2.2 Quality-aware crowdsourcing

Data quality in crowdsourcing has been studied in a few works [29, 49, 50, 43, 51]. One interesting line of works [50, 43, 51] in this direction have studied truthful mechanisms for quality-aware crowdsourcing where workers have private participating costs. Some other works have focused on learning data quality of workers, e.g., by exploiting the correlation of their data [29, 30], or allocating tasks on the fly [49, 52]. Different from these works, this paper focuses on the situation where quality is a worker's private information that is unknown to the requester. A few recent works [24, 53] have designed truthful mechanisms for quality elicitation in quality-aware crowdsourcing. Compared to these works, this paper studies privacy-preserving and truthful mechanisms that not only achieves data quality elicitation, but also protect the privacy of workers' quality and data.

3.2.3 Truthful mechanisms for crowdsourcing

There have been a lot of research on truthful mechanisms for crowdsourcing [22, 25, 45, 54, 55]. Most of these mechanisms have focused on incentivizing workers to truthfully reveal their private participating costs. Different from these works, this paper studies the situation where workers have private data quality that they can misreport. As a result, existing mechanisms for cost elicitation (such as the classical VCG auction and the characterization of truthful mechanisms [56, Theorem 9.36]) cannot work for quality elicitation. Furthermore, this paper aims at joint elicitation of quality, effort, and data, which is more challenging than elicitation of cost or quality only.

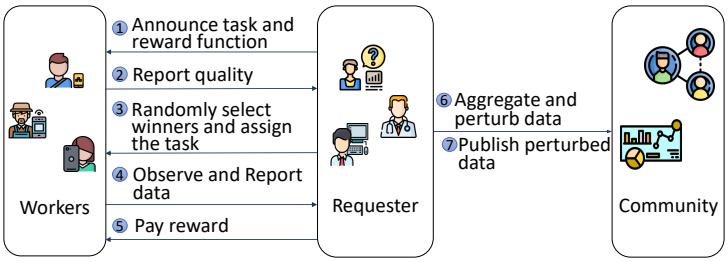

Figure 3.2: Structure and procedure of the quality and privacy aware data crowdsourcing system.

Table 3.1: Main Notation

Symbol	Description
q_i	True quality of worker i
d_i	True data of worker i
e_i	Actual effort of worker i
q'_i	Reported quality of worker i
d'_i	Reported data of worker i
e'_i	Desired effort of worker i
$\boldsymbol{q}$	Workers' true quality
$\boldsymbol{d}$	Workers' true data
$\boldsymbol{e}$	Workers' actual effort
$\boldsymbol{q}'$	Workers' reported quality
$\boldsymbol{d}'$	Workers' reported data
$\boldsymbol{e}'$	Workers' desired effort
$\boldsymbol{q}_{-i}$	Workers' true quality other than worker i's
$\boldsymbol{d}_{-i}$	Workers' true data other than worker i's
$\boldsymbol{e}_{-i}$	Workers' actual effort other than worker i's
$\boldsymbol{q}'_{-i}$	Workers' reported quality other than worker i's
$\boldsymbol{d}'_{-i}$	Workers' reported data other than worker i's
$\boldsymbol{e}'_{-i}$	Workers' desired effort other than worker i's

There have also been some studies on truthful mechanisms for crowdsourcing where workers have hidden efforts [26, 25, 57, 58] and private data [26, 25, 57]. This paper is different from these works as our goal is not only to jointly elicit workers' private data, quality, and hidden effort, but also to protect the privacy of workers' quality and data.

3.3 System Model

In this section, we describe the quality-aware crowdsourcing system studied in this paper. We consider a crowdsourcing system that consists of a requester (also referred to as worker 0) and a set of available workers $\mathcal{U} = \{1, 2, \cdots, n\}$. Fig.3.2 illustrates the structure and procedure of the crowdsourcing system with following key elements.

Data observation. The requester aims to estimate an unknown variable X of a crowdsourcing task by recruiting the workers to observe X. For ease of exposition, we assume that

the variable X takes a binary value from $\{-1, +1\}$. After performing the task, worker i obtains data D_i and reports it to the requester. The accuracy of data D_i is measured by the *correct probability* p_i, which is the probability that D_i is equal to the ground truth of X, given by

$$p_i = \Pr_{X|D_i(q_i,e_i)}(d_i) = q_i e_i + \hat{q}(1 - e_i). \tag{3.1}$$

Here p_i depends on the *quality* q_i of worker i and the *effort* e_i exerted by worker i in the task, and $\hat{q}$ denotes the correct probability when the worker makes no effort (e.g., using the prior distribution of X).

Worker quality. Given that worker i makes an effort in the task, the quality $q_i \in [1/2, 1]$ of worker i is her correct probability p_i. The quality of a worker is an intrinsic coefficient that captures the worker's capability for the task. A larger q_i means higher quality. We assume that each worker knows her quality, which is unknown to the requester. For ease of exposition, we assume that each worker's quality is within the range of $[\underline{q}, \overline{q}]$, which is known to the requester.

Worker effort. The effort e_i represents whether worker i makes an effort in the task, where $e_i = 0$ and $e_i = 1$ indicate making and not making an effort, respectively. If worker i makes an effort, then p_i is equal to worker i's quality q_i; if not, p_i is equal to $\hat{q}$. To ensure that making effort is meaningful, we assume that $q_i > \hat{q}$.

Task allocation. The requester allocates the task to workers by assigning effort e'_i to each worker i based on their quality. If worker i is selected as a winner, then her assigned effort is $e'_i = 1$, otherwise it's $e'_i = 0$. To this end, each worker reports her quality q'_i to the requester. Since the true quality q_i is worker i's private information, she may misreport her quality such that $q'_i \neq q_i$. Based on the workers' reported quality profile $\boldsymbol{q}' = \{q'_i : i \in \mathcal{U}\}$, the requester allocates the task to a subset $\mathcal{S} \subset \mathcal{N}$ of workers where $2 \leq |\mathcal{S}| \leq n$, and determines each worker's reward r_i according to a reward function. The reward function is predefined by the requester and announced to all workers before they report their quality. Note that the reward function can only depend on the information that is known to the requester, i.e., $\boldsymbol{q}'$, $\boldsymbol{e}'$, and $\boldsymbol{d}'$. In the design of the reward paid to worker i, we will also use another worker j's ($j \neq i$) information.

Data aggregation. After the requester collects the data from workers, it uses a weighted data aggregation method to calculate an estimate x_0 of the unknown variable X based on the collected data, given by

$$x_0\left(\boldsymbol{q}', \boldsymbol{e}', \boldsymbol{d}'\right) = \operatorname{sign}\left(\sum_{i\in\mathcal{S}} \alpha_i d_i'\right),$$

where d_i is a sample realization of the random data D_i, d_i' is worker i's reported data, and $\alpha_i = \left(2q_i' - 1\right)^2$ is worker i's data weight [59], which depends on worker i's reported quality. Based on the aggregation method, the utility of crowdsourcing is measured by the accuracy p_c of the aggregated data x_0, i.e.,

$$p_c\left(\boldsymbol{q}', \boldsymbol{e}', \boldsymbol{d}'\right) = E_{X|\boldsymbol{d}(\boldsymbol{q},\boldsymbol{e})}\left[\mathbf{1}_{X=x_0(\boldsymbol{q}',\boldsymbol{e}',\boldsymbol{d}')}\right]. \tag{3.2}$$

Note that the expectation is taken over the posterior distribution $X|\boldsymbol{d}$ conditioned on workers' true data, and $\boldsymbol{d}\left(\boldsymbol{q}, \boldsymbol{e}\right)$ implies that workers' data is determined by workers' quality and effort. If the task is not assigned to any worker (i.e., $\mathcal{S} = \emptyset$), the accuracy p_c is defined to be 0.

Worker payoff. Each worker i's payoff is the difference between the reward paid by the requester and her cost in the task, given by

$$u_i\left(\boldsymbol{q}', e_i', e_i, d_i', d_j\right) = r_i\left(\boldsymbol{q}', e_i', d_i', d_j\right) - c_i e_i, \tag{3.3}$$

where the cost coefficient c_i captures the resources consumed by worker i if she makes an effort $e_i = 1$ in the task. If worker i make no effort $e_i = 0$, it incurs no cost. Here we assume that workers have the same cost coefficient (i.e., $c = c_i, \forall i \in \mathcal{U}$) which is known to the requester. This assumption is reasonable when the cost of performing a task is determined by a uniform market price (e.g., \$0.5 for each task). We can also relax the restrict of the uniform cost coefficient, though the analysis becomes more complicated. The detailed reward function will be given in section 3.5.

3.4 Problem Formulation

In this section, based on the quality-aware crowdsourcing framework described in Section 3.3, we formulate the problem of mechanism design with three objectives as follows.

3.4.1 Truthful Elicitation of Data Quality

Since the workers have private quality and data and make hidden effort, if any worker manipulates her reported quality, reported data, or exerted effort, then the estimate of the ground truth x_0 can be incorrect, i.e.,

$$x_0\left(\boldsymbol{q}', \boldsymbol{e}', \boldsymbol{d}'\right) \neq x_0(\boldsymbol{q}, \boldsymbol{e}, \boldsymbol{d}).$$

More importantly, an incorrect estimate will affect the accuracy of aggregated data, i.e.,

$$p_c\left(\boldsymbol{q}', \boldsymbol{e}', \boldsymbol{d}'\right) \neq p_c(\boldsymbol{q}, \boldsymbol{e}, \boldsymbol{d}).$$

This means that untruthful workers would result in the requester's incorrect knowledge of the data accuracy. This is undesirable since the data accuracy is a key performance metric for crowdsourcing, and the requester may need the correct data accuracy to make decisions. Thus motivated, our mechanism should incentivize workers to behave truthfully. Therefore, a primary mechanism design objective is to achieve the truthfulness property defined below.

Definition 3.1 A mechanism achieves truthful strategies of all workers as a *Nash equilibrium (NE)* in expectation if, given other workers truthfully report their quality and data and make desired effort, the optimal strategy for each worker i to maximize her expected payoff is to truthfully report her quality and data and make desired effort. i.e.,

$$\begin{aligned} &E_{D_j|d_i(q_i,e_i)}\left[u_i\left(q_i, \boldsymbol{q}_{-i}, e_i', e_i', d_i, d_j\right)\right] \geq \\ &E_{D_j|d_i(q_i,e_i)}\left[u_i\left(q_i', \boldsymbol{q}_{-i}, e_i', e_i, d_i', d_j\right)\right], \forall (q_i', e_i, d_i'), \end{aligned}$$

where d_j $(j \neq i)$ is worker j's data.

Note that in the above inequality, corresponding to the right side, the two e_i's on the left side have meanings as following: the first e_i' denotes worker i's assigned effort, and the second e_i' denotes the actual effort made by worker i which equals to the requester's desired effort $e_i = e_i'$.

To incentivize workers to participate in crowdsourcing, the payoff of each worker i should be non-negative. This property is formally known as individual rationality as stated below.

Definition 3.2 A mechanism is *individually rational (IR)* if for each worker i, given that she truthfully reports her quality and data and makes the desired effort, her expected payoff is non-negative, i.e.,

$$E_{D_j|d_i(q_i)}\left[u_i\left(q_i, \boldsymbol{q}'_{-i}, e_i', e_i', d_i, d_j\right)\right] \geq 0, \forall \boldsymbol{q}'_{-i}.$$

3.4.2 Protecting Privacy of Workers' Quality and Data

As our mechanism incentivizes a worker to truthfully report her private quality and data, these may be the worker's sensitive information that needs to be protected. For example, a worker's identity can be inferred from her location, which can be further inferred from her quality or data reported to the requester. On the other hand, it is possible for a curious worker (or an adversary) to infer another worker's private quality and/or data from the outcomes of the mechanism. This leakage of privacy would discourage workers from participating in crowdsourcing.

We use the following example to illustrate how quality information leakage can happen. Consider a crowdsourcing system where the requester selects 3 winners from 5 workers to perform the task. Assume that the workers' quality profile is $\boldsymbol{q} = \{0.65, 0.7, 0.8, 0.75, 0.85\}$. Thus the optimal winner set is $\{5, 3, 4\}$. Suppose worker 2 intends to infer other workers' quality, and her quality is changed from 0.7 to 0.8 while the other workers' quality remains the same. Then the new winner set is $\{5, 2, 3\}$. As a result, worker 2 can conclude that worker 4's quality is between 0.7 and 0.8. After several rounds, worker 2 can narrow down worker 4's quality range, or even infer the exact value.

Motivated by the discussions above, another important goal of this paper is to design a differentially private mechanism that can protect workers' quality privacy and data privacy. The formal definition is as below:

Definition 3.3 A mechanism $\mathcal{M}$ is ϵ-*differential privacy* if for any two input sets $\mathcal{A}$ and $\mathcal{B}$ with a single input difference, and for any set of outcomes $\mathcal{O} \subseteq Range(\mathcal{M})$:

$$\Pr[\mathcal{M}(\mathcal{A}) \in \mathcal{O}] \leq \exp(\epsilon) \times \Pr[\mathcal{M}(\mathcal{B}) \in \mathcal{O}],$$

where ϵ is a small positive constant and $Range(\mathcal{M})$ is the outcome space of mechanism $\mathcal{M}$.

Another relevant concept is approximate differential privacy, which is a relaxation of differential privacy, defined as follows.

Definition 3.4 A mechanism $\mathcal{M}$ is (ϵ, δ)-*differential privacy* if for any two input sets $\mathcal{A}$ and $\mathcal{B}$ with a single input difference, and for any set of outcomes $\mathcal{O} \subseteq Range(\mathcal{M})$:

$$\Pr[\mathcal{M}(\mathcal{A}) \in \mathcal{O}] \leq \exp(\epsilon) \times \Pr[\mathcal{M}(\mathcal{B}) \in \mathcal{O}] + \delta.$$

3.4.3 Accuracy of Aggregated Data

Besides the properties of truthfulness and differential privacy, another important objective of our mechanism is to improve the accuracy p_c of the aggregated data. Recall that we use a weighted aggregation method to calculate the estimate x_0 of the target variable X. According to [59], the optimal weight that maximizes the expected accuracy is given by $\alpha_i = (2q_i' - 1)^2$, which leads to a lower bound of the expected accuracy:

$$E[p_c] \geq 1 - \exp\left(-\frac{1}{2}\sum_{i \in \mathcal{S}} \alpha_i\right). \tag{3.4}$$

In the rest of this paper, we use $\sum_{i \in \mathcal{S}} \alpha_i$ instead of p_c as an approximate metric of the accuracy. This is reasonable since when the lower bound of $E[p_c]$ meets some threshold requirement, the accuracy $E[p_c]$ also meets that requirement.

3.5 Privacy-Preserving Mechanism for Truthful Data Quality Elicitation: Single-Task Case

In this section, to achieve the objectives formulated in Section 3.4, we present the design of a Single-Task Privacy-preserving mechanism for truthful Data Quality Elicitation (S-PDQE).

S-PDQE consists of a differentially private task allocation algorithm and a differentially private data aggregation algorithm.

3.5.1 Preliminaries of Differential Privacy

We first introduce some basics of differential privacy which will be used in the mechanism design later.

We use the exponential mechanism [60] to achieve differential privacy for workers' quality. It is a general mechanism to design privacy-preserving mechanisms, which uses a score function $f(\mathcal{A}, o)$ to map the input set $\mathcal{A}$ and an outcome $o \in \mathcal{O}$ to a real-valued score. The better the outcome o is for the input set $\mathcal{A}$, the higher the score associated with the outcome o. The probability of outcome o increases exponentially with its score getting higher. The concept of the exponential mechanism is defined as follows.

Definition 3.5 Given an outcome space $\mathcal{O}$, an input set $\mathcal{A}$, a score function $f(\mathcal{A}, o)$ and a constant ϵ_q, the exponential mechanism $\epsilon_f^{\epsilon_q}(\mathcal{A})$ chooses an outcome $o \in \mathcal{O}$ with probability

$$\Pr\left[\epsilon_f^{\epsilon_q}(\mathcal{A}) = o\right] \propto \exp(\epsilon_q f(\mathcal{A}, o)).$$

We define Λ to be the largest possible difference in the score function when applied to two inputs that differ in one value. We will use two properties of the exponential mechanism as follows.

Proposition 3.1 [61] The exponential mechanism achieves $(2\epsilon_q\Lambda)$-*differential privacy*.

Proposition 3.2 [60] For any $\alpha \geq 0$, the exponential mechanism which achieves $(2\epsilon_q\Lambda)$-*differential privacy* ensures that

$$\begin{array}{l} \Pr[f\left(\mathcal{A}, \epsilon_f^{\epsilon_q}(\mathcal{A})\right) < \max_o f(\mathcal{A}, o) - \ln\left(|\mathcal{O}| / |\mathcal{O}^*|\right)/\epsilon_q \\ -\alpha/\epsilon_q] \quad \leq \exp(-\alpha) \end{array},$$

where $\mathcal{O}^*$ is the subset of $\mathcal{O}$ that achieves $f(\mathcal{A}, o) = \max_o f(\mathcal{A}, o)$.

Proposition 3.3 [61] The sequential application of exponential mechanism $\mathcal{M}_i$, each giving ϵ_i-differential privacy, yields $\sum_i \epsilon_i$-differential privacy.

To provide differential privacy for workers' data, we use the randomized response mechanism [62]. This mechanism randomizes each bit independently by flipping it with a certain probability, with the definition given below.

Definition 3.6 Given a set $\mathcal{B} = (b_1, b_2, \cdots)$, in which $b_i \in \{-1, 1\}$, $\forall i$, and a constant ϵ_d, let $R(b)$ denote a Bernoulli random variable with $\Pr(R(b_i) = b_i) = \frac{e^{\epsilon_d}}{e^{\epsilon_d}+1}$ and $\Pr(R(b_i) = -b_i) = \frac{1}{e^{\epsilon_d}+1}$. The randomized response mechanism outputs $(R(b_1), R(b_2), \cdots)$.

Proposition 3.4 The randomized response mechanism achieves ϵ_d-*differential privacy*.

3.5.2 Truthful and Differentially Private Single-Task Allocation

Next we present the differentially private task allocation algorithm of the S-PDQE mechanism in detail. We will show how this algorithm achieves the desired properties of the mechanism later in this section.

Design Overview. The task allocation algorithm integrates the exponential mechanism with a truthful incentive mechanism to achieve truthfulness and differential privacy. The winners are selected iteratively in this algorithm. In each iteration, each worker is assigned a probability to be selected. The requester selects one winner based on the probability distribution in each iteration. The process repeats until K winners are selected. In the end, the requester pays each winner according to the reward function.

Algorithm Design.

- *Phase I: Winner selection.* The mechanism assigns each worker $i \in \mathcal{R}$ a probability of being selected as follows. The requester first calculates each worker's contribution α_i to data accuracy, which is determined by each worker's reported quality. Each worker's contribution α_i is given by

$$\alpha_i = (2q_i' - 1)^2,$$

Algorithm 1: Task allocation with reward payments

Input: The set of users $\mathcal{U}$; Number of winners K; Quality profile $\boldsymbol{q}$; differential privacy parameter ϵ_q and $\delta \in (0, \frac{1}{2}]$.

Output: A set of winners $\mathcal{S}$ and winners' reward profile $\boldsymbol{r}$.

$\mathcal{S} \leftarrow \emptyset, \mathcal{R} \leftarrow \mathcal{U}$;
// Phase I;
foreach $i \in \mathcal{U}$ **do**
    $r_i \leftarrow 0$, $e_i \leftarrow 0$;
while $|\mathcal{S}| < K$ **do**
    foreach $i \in \mathcal{R}$ **do**
        Calculate the probability $\Pr_i\left(q_i', \boldsymbol{q}_{-i}'\right)$ of each worker being selected as a winner according to the score function;
    Select one worker randomly, denoted by i', according to the computed probability distribution;
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{i'\}, \mathcal{R} \leftarrow \mathcal{R} \setminus \{i'\}, e_i' \leftarrow 1$;
// Phase II;
foreach $i \in \mathcal{S}$ **do**
    calculate r_i according to (3.6);
return $\mathcal{S}$ *and* $\boldsymbol{r}$.

which is also the weight in data aggregation. It is obvious that α_i is an increasing function of worker i's reported quality q_i'. To apply the exponential mechanism, we take $f(q_i) = \alpha_i$ as the score function, for any worker $i \in \mathcal{R}$. The probability $\Pr_i$ of worker i to be selected as a winner in each iteration is

$$\Pr_i\left(q_i', \boldsymbol{q}_{i-1}'\right) = \begin{cases} \frac{\exp\left(\epsilon' \frac{\alpha_i}{\alpha_{max}}\right)}{\sum_{j \in \mathcal{R}} \exp\left(\epsilon' \frac{\alpha_j}{\alpha_{max}}\right)}, & \text{if } i \in \mathcal{R} \\ 0, & \text{otherwise} \end{cases} \tag{3.5}$$

where $\epsilon' = \epsilon_q \alpha_{max}/(4e\Delta \ln(e/\delta))$ and $\Delta = \overline{q} - \underline{q}$. We also normalize α_i as $\frac{\alpha_i}{\alpha_{\max}}$ to ensure that the value of the score function is non-negative. The score function ensures that a worker who makes more contribution to the accuracy will have a higher probability to be selected as a winner.

- *Phase II: Reward determination.* The requester pays each winner i ($i \in \mathcal{S}$) a reward, which can be expressed as

$$r_i\left(q_i', \boldsymbol{q}_{-i}', e_i', d_i', d_j\right) = k\left[\frac{\mathbf{1}_{d_j=d_i} + q_j - 1}{2q_j - 1} - q_i'\right] + \frac{\int_{\underline{q}}^{q_i'} kq\mathrm{Pr}_i\left(q, \boldsymbol{q}_{i-1}'\right) dq + ce_i'}{\mathrm{Pr}_i\left(q_i', \boldsymbol{q}_{i-1}'\right)}, \tag{3.6}$$

where q_j is worker j's quality, k is any constant that satisfies

$$k \geq \frac{c}{(\underline{q} - \hat{q})\mathrm{Pr}(\underline{q})},$$

and $\mathbf{1}_A$ is the indicator function that is equal to 1 if condition A is true and 0 otherwise. A worker receives no reward if she is not selected as a winner.

3.5.3 Differentially Private Data Aggregation

Next we present the differentially private data aggregation algorithm of S-PDQE. We use the randomized response mechanism to protect workers' data privacy. The details are given in Algorithm 2.

Algorithm 2: Data perturbation

Input: Aggregated data x_0; differential privacy parameter ϵ_d.
Output: Perturbed data $\widehat{x}_0$.
1 Generate $R(x_0)$ using the randomized response mechanism in Definition 6;
2 $\widehat{x}_0 \leftarrow R(x_0)$;
3 **return** $\widehat{x}_0$.

Here $R(x_0)$ is defined in Definition 3.6. Algorithm 2 takes the aggregated data and differential privacy parameter ϵ_d as inputs. For each aggregated data, the algorithm performs the randomized response mechanism on it. The tradeoff between privacy and data accuracy is controlled by the differential privacy parameter ϵ_d.

3.5.4 Performance Analysis of S-PDQE Mechanism

In this section, we provide thorough performance analysis of S-PDQE. It shows that the mechanism achieves the properties of computational efficiency, truthfulness, individual rationality, differential privacy, and bounded approximation gap.

First we show that the task allocation algorithm of S-PDQE is computational efficiency.

Theorem 3.1 The complexity of the task allocation algorithm of S-PDQE is $\mathcal{O}(Kn)$, where K is the number of winners, and n is the number of workers.

Proof: We can see that the outer while-loop (Line 6-12) runs K rounds, and the inner for-loop (Line 7-9) runs at most n rounds. Thus, the total computational complexity of the task allocation algorithm is $\mathcal{O}(Kn)$.

Then we prove the truthful and individually rational properties.

Theorem 3.2 The task allocation algorithm of S-PDQE with the reward payments is truthful and individually rational.

Proof: From (3.3) and (3.6), we can express worker i's expected payoff as

$$\begin{aligned}
&E[u_i(\boldsymbol{q}', e_i', e_i, d_i', d_j)] \\
&= \Pr_i\left(q_i', \boldsymbol{q}_{i-1}'\right) \times r_i(\boldsymbol{q}', e_i', d_i', d_j) + \left(1 - \Pr_i\left(q_i', \boldsymbol{q}_{i-1}'\right)\right) \times 0 - ce_i \\
&= k\left[\frac{\mathbf{1}_{d_j=d_i} + q_j - 1}{2q_j - 1} - q_i'\right] \Pr_i\left(q_i', \boldsymbol{q}_{i-1}'\right) + \int_{\underline{q}}^{q_i'} kq \Pr_i\left(q, \boldsymbol{q}_{i-1}'\right) dq + ce_i' - ce_i.
\end{aligned}$$

Next we use three lemmas to prove that, given that each worker aims to maximize her expected payoff, 1) her optimal reported data is her true data $d_i' = d_i$; 2) her optimal effort is the desired effort $e_i = e_i'$; 3) her optimal reported quality is her true quality $q_i' = q_i$.

Lemma 3.1 Given that worker i reports any quality q_i' and makes any effort e_i, her optimal reported data is her true data $d_i' = d_i$.

The proof is similar to that of Lemma 1 in [24], and is thus omitted here. Next we express worker i's expected payoff when she reports her true data $d_i' = d_i$. Since

$$\begin{aligned}
&E_{d_j|d_i(q_i,e_i)}\left[\mathbf{1}_{d_j=d_i}\right] = \Pr_{d_j|d_i(q_i,e_i)}(d_i) \\
&= q_j \Pr_{X|d_i(q_i,e_i)}(d_i) + (1-q_j)\left(1 - \Pr_{X|d_i(q_i,e_i)}(d_i)\right) \\
&= (2q_j - 1)\Pr_{X|d_i(q_i,e_i)}(d_i) + 1 - q_j,
\end{aligned}$$

we have

$$E_{d_j|d_i(q_i,e_i)}\left[\frac{\mathbf{1}_{d_j=d_i} + q_j - 1}{2q_j - 1}\right] = \Pr_{X|d_i(q_i,e_i)}(d_i). \tag{3.7}$$

Thus, from (3.1) and (3.7), worker i's expected payoff is given by

$$\begin{aligned}
&E_{d_j|d_i(q_i,e_i)}[u_i(\boldsymbol{q}', e_i', e_i, d_i, d_j)] \\
&= k[\hat{q} + (q_i - \hat{q})e_i - q_i']\Pr_i\left(q_i', \boldsymbol{q}_{i-1}'\right) + \int_{\underline{q}}^{q_i'} kq\Pr_i\left(q, \boldsymbol{q}_{i-1}'\right)dq + ce_i' - ce_i.
\end{aligned} \tag{3.8}$$

For convenience, we define

$$E[u_i(\boldsymbol{q}', q_i, e_i', e_i)] = E_{d_j|d_i(q_i,e_i)}[u_i(\boldsymbol{q}', e_i', e_i, d_i, d_j)].$$

Lemma 3.2 Given that worker i reports any quality q_i' and truthfully reports her data d_i, her optimal actual effort is the desired effort $e_i = e_i'$.

Proof: Using (3.8), when worker i is selected as a winner, i.e., $e_i' = 1$, we have

$$\begin{aligned}
&E[u_i(\boldsymbol{q}', q_i, 1, 1)] - E[u_i(\boldsymbol{q}', q_i, 1, 0)] \\
&\qquad = k(q_i - \hat{q})\Pr_i\left(q_i', \boldsymbol{q}_{i-1}'\right) - c \geq 0.
\end{aligned}$$

Hence the optimal effort to make for a winner is $e_i = e_i' = 1$. When worker i is not a winner, it can be easily seen that the optimal effort is $e_i = e_i' = 0$. □

Lemma 3.3 Given that worker i reports her true data d_i and makes the desired effort e_i', her optimal reported quality is her true quality $q_i' = q_i$.

Proof: It suffices to show that $E[u_i(q_i, \boldsymbol{q}'_{-i}, e'_i)] \geq E[u_i(q, \boldsymbol{q}'_{-i}, e'_i)], \forall q \neq q_i$. Let $q'_i \geq q_i$. As $\Pr_i\left(q'_i, \boldsymbol{q}'_{i-1}\right)$ is an increasing function of q_i, we have

$$\begin{aligned}
&E[u_i(q_i, \boldsymbol{q}'_{-i})] - E[u_i(q'_i, \boldsymbol{q}'_{-i})] \\
&= k(q_i - q_i)\Pr_i\left(q'_i, \boldsymbol{q}'_{i-1}\right) + \int_{\underline{q}}^{q_i} kq\Pr_i\left(q, \boldsymbol{q}'_{i-1}\right) dq \\
&- (k(q_i - q'_i)\Pr_i\left(q'_i, \boldsymbol{q}'_{i-1}\right) + \int_{\underline{q}}^{q'_i} kq\Pr_i\left(q, \boldsymbol{q}'_{i-1}\right) dq) \\
&= k(q'_i - q_i)\Pr_i\left(q'_i, \boldsymbol{q}'_{i-1}\right) - \int_{q_i}^{q'_i} kq\Pr_i\left(q, \boldsymbol{q}'_{i-1}\right) dq \\
&\geq k(q'_i - q_i)\Pr_i\left(q'_i, \boldsymbol{q}'_{i-1}\right) - k(q'_i - q_i)\Pr_i\left(q'_i, \boldsymbol{q}'_{i-1}\right) = 0.
\end{aligned}$$

Now let $q'_i \leq q_i$. Then we have

$$\begin{aligned}
&E[u_i(q_i, \boldsymbol{q}'_{-i})] - E[u_i(q'_i, \boldsymbol{q}'_{-i})] \\
&= k(q_i - q_i)\Pr_i\left(q'_i, \boldsymbol{q}'_{i-1}\right) + \int_{\underline{q}}^{q_i} kq\Pr_i\left(q, \boldsymbol{q}'_{i-1}\right) dq \\
&- (k(q_i - q'_i)\Pr_i\left(q'_i, \boldsymbol{q}'_{i-1}\right) + \int_{\underline{q}}^{q'_i} kq\Pr_i\left(q, \boldsymbol{q}'_{i-1}\right) dq) \\
&= \int_{q'_i}^{q_i} kq\Pr_i\left(q, \boldsymbol{q}'_{i-1}\right) dq - k(q_i - q'_i)\Pr_i\left(q'_i, \boldsymbol{q}'_{i-1}\right) \\
&\geq k(q_i - q'_i)\Pr_i\left(q'_i, \boldsymbol{q}'_{i-1}\right) - k(q_i - q'_i)\Pr_i\left(q'_i, \boldsymbol{q}'_{i-1}\right) = 0.
\end{aligned}$$

Given that worker i reports her true quality, her expected payoff is given by

$$E[u_i(q_i, \boldsymbol{q}'_{-i})] = \int_{\underline{q}}^{q_i} kq\Pr_i\left(q, \boldsymbol{q}'_{i-1}\right) dq.$$

As $\Pr_i\left(q_i, \boldsymbol{q}'_{i-1}\right) \geq 0$ and it is increasing with q_i, it can be easily seen that individual rationality is achieved. □

Next we show that the mechanism achieves differential privacy for workers' quality.

Theorem 3.3 For any constants $\epsilon_q > 0$ and $\delta \in \left(0, \frac{1}{2}\right]$, the task allocation algorithm of S-PDQE achieves $(\epsilon_q(e-1)/e, \delta)$-differential privacy for workers' quality, where e is the base of the natural logarithm.

Proof: We use $\boldsymbol{q}$ and $\boldsymbol{q}'$ denote two quality profiles that only differ in one worker d's quality. Let $M(\boldsymbol{q})$ and $M(\boldsymbol{q}')$ be the sequences of winners selected by our algorithm with inputs $\boldsymbol{q}$ and $\boldsymbol{q}'$, respectively. In our proof, we show that our score function achieves differential privacy when an arbitrary sequence of workers $W = i_1, i_2, \cdots, i_l$ with any length l is selected. Given quality profiles $\boldsymbol{q}$ and $\boldsymbol{q}'$, from (3.10) we can have the relative probability of our algorithm is

$$\frac{\Pr[M(\boldsymbol{q}) = W]}{\Pr[M(\boldsymbol{q}') = W]} = \prod_{j=1}^{l} \frac{\dfrac{\exp\left(\epsilon' \frac{\left(2q_{i_j}-1\right)^2}{(2q_{\max}-1)^2}\right)}{\sum_{i\in\mathcal{U}_j} \exp\left(\epsilon' \frac{\left(2q'_{i_j}-1\right)^2}{(2q_{\max}-1)^2}\right)}}{\dfrac{\exp\left(\epsilon' \frac{\left(2q'_{i_j}-1\right)^2}{(2q_{\max}-1)^2}\right)}{\sum_{j\in\mathcal{U}_j} \exp\left(\epsilon' \frac{\left(2q'_{i_j}-1\right)^2}{(2q_{\max}-1)^2}\right)}}$$

$$= \prod_{j=1}^{l} \frac{\exp\left(\epsilon' \frac{\left(2q_{i_j}-1\right)^2}{(2q_{\max}-1)^2}\right)}{\exp\left(\epsilon' \frac{\left(2q'_{i_j}-1\right)^2}{(2q_{\max}-1)^2}\right)} \times \prod_{j=1}^{l} \frac{\sum_{j\in\mathcal{U}_j} \exp\left(\epsilon' \frac{\left(2q'_{i_j}-1\right)^2}{(2q_{\max}-1)^2}\right)}{\sum_{i\in\mathcal{U}_j} \exp\left(\epsilon' \frac{\left(2q_{i_j}-1\right)^2}{(2q_{\max}-1)^2}\right)},$$

where $\mathcal{U}_j = \mathcal{U} \setminus \{i_1, i_2, \cdots, i_{j-1}\}$. Then, we discuss two cases of this equation. When $q_d > q'_d$, we have $(2q_d - 1)^2 > (2q'_d - 1)^2$, then

$$\prod_{j=1}^{l} \frac{\sum_{j\in\mathcal{U}_j} \exp\left(\epsilon' \frac{\left(2q'_{i_j}-1\right)^2}{(2q_{\max}-1)^2}\right)}{\sum_{i\in\mathcal{U}_j} \exp\left(\epsilon' \frac{\left(2q_{i_j}-1\right)^2}{(2q_{\max}-1)^2}\right)} \leq 1.$$

Therefore, we have

$$\begin{aligned}
\frac{\Pr[M(\boldsymbol{q})=W]}{\Pr[M(\boldsymbol{q}')=W]} &\leq \frac{\exp\left(\epsilon'\frac{(2q_d-1)^2}{(2q_{\max}-1)^2}\right)}{\exp\left(\epsilon'\frac{\left(2q'_d-1\right)^2}{(2q_{\max}-1)^2}\right)} \\
&= \exp\left(\epsilon'\frac{(2q_d-1)^2-(2q'_d-1)^2}{(2q_{\max}-1)^2}\right) = \exp\left(\epsilon'\frac{4(q_d-q'_d)(q_d+q'_d-1)}{(2q_{\max}-1)^2}\right) \\
&\leq \exp\left(4\epsilon'\frac{q_d-q'_d}{(2q_{\max}-1)^2}\right) \leq \exp\left(\epsilon'\frac{4\Delta}{(2q_{\max}-1)^2}\right).
\end{aligned}$$

Now let $q_d < q'_d$,we have $(2q_d-1)^2 < (2q'_d-1)^2$, then

$$\prod_{j=1}^{l}\frac{\exp\left(\epsilon'\frac{\left(2q_{i_j}-1\right)^2}{(2q_{\max}-1)^2}\right)}{\exp\left(\epsilon'\frac{\left(2q'_{i_j}-1\right)^2}{(2q_{\max}-1)^2}\right)} \leq 1.$$

Therefore, we have

$$\begin{aligned}
\frac{\Pr[M(\boldsymbol{q})=W]}{\Pr[M(\boldsymbol{q}')=W]} &\leq \prod_{j=1}^{l}\frac{\sum_{j\in\mathcal{U}_j}\exp\left(\epsilon'\frac{\left(2q'_d-1\right)^2}{(2q_{\max}-1)^2}\right)}{\sum_{i\in\mathcal{U}_j}\exp\left(\epsilon'\frac{(2q_d-1)^2}{(2q_{\max}-1)^2}\right)} \\
&= \prod_{j=1}^{l}\frac{\sum_{i\in\mathcal{U}_j}\exp\left(\epsilon'\frac{\left(2q'_d-1\right)^2-(2q_d-1)^2}{(2q_{\max}-1)^2}+\epsilon'\frac{(2q_d-1)^2}{(2q_{\max}-1)^2}\right)}{\sum_{i\in\mathcal{U}_j}\exp\left(\epsilon'\frac{(2q_d-1)^2}{(2q_{\max}-1)^2}\right)} \\
&= \prod_{j=1}^{l}E_{i\in\mathcal{U}_j}\left[\exp\left(\epsilon'\frac{(2q'_d-1)^2-(2q_d-1)^2}{(2q_{\max}-1)^2}\right)\right] \\
&\leq \prod_{j=1}^{l}E_{i\in\mathcal{U}_j}\left[\exp\left(4\epsilon'\frac{(q_d-q'_d)}{(2q_{\max}-1)^2}\right)\right].
\end{aligned}$$

For all $\epsilon' \leq (2q_{\max}-1)^2/4\Delta$, we have

$$\begin{aligned}
&\prod_{j=1}^{l}E_{i\in\mathcal{U}_j}\left[\exp\left(4\epsilon'\frac{(q_d-q'_d)}{(2q_{\max}-1)^2}\right)\right] \leq \prod_{j=1}^{l}E_{i\in\mathcal{U}_j}\left[1+\frac{4(e-1)\epsilon'(q_d-q'_d)}{(2q_{\max}-1)^2}\right] \\
&\leq \exp\left(\frac{4(e-1)\epsilon'}{(2q_{\max}-1)^2}\sum_{j=1}^{l}E_{i\in\mathcal{U}_j}\left[q_d-q'_d\right]\right).
\end{aligned}$$

Note that $\Pr\left[\sum_{j=1}^{l} E_{i\in\mathcal{U}_j}\left[q_d - q'_d\right] > \Delta\ln(e/\delta)\right] \leqslant \delta$ [60]. Let $\mathcal{W}$ be the outcome space, where $W \in \mathcal{W}$ is a sequence of winners. Let $\mathcal{W}' = \{W \in \mathcal{W} | \sum_{j=1}^{l} E_{i\in\mathcal{U}_j}\left[q_d - q'_d\right] > \Delta\ln(e/\delta)\}$, and $\mathcal{W}'' = \mathcal{W} \setminus \mathcal{W}'$, i.e., $\mathcal{W}' \cap \mathcal{W}'' = \mathcal{W}$. Then we have

$$
\begin{aligned}
&\Pr[M(\boldsymbol{q}) \in \mathcal{W}] \\
&= \sum_{W\in\mathcal{W}} \Pr[M(\boldsymbol{q}) = W] \\
&= \sum_{W\in\mathcal{W}'} \Pr[M(\boldsymbol{q}) = W] + \sum_{W\in\mathcal{W}''} \Pr[M(\boldsymbol{q}) = W] \\
&\leq \sum_{W\in\mathcal{W}'} \exp\left(\frac{4(e-1)\epsilon'}{(2q_{\max}-1)^2}\Delta\ln(e/\delta)\right) \Pr[M(\boldsymbol{q}')=W] \\
&\quad + \delta \\
&\leq \exp\left(\frac{4(e-1)\epsilon'}{(2q_{\max}-1)^2}\Delta\ln(e/\delta)\right) \Pr[M(\boldsymbol{q}') \in \mathcal{W}] + \delta \\
&= \exp\left(\epsilon_q(e-1)/e\right) \Pr[M(\boldsymbol{q}') \in \mathcal{W}] + \delta
\end{aligned}
$$

□

Next we show that the task allocation algorithm achieves a bounded gap from the optimal task allocation strategy.

Theorem 3.4 With a probability at least $1 - 1/n^{O(1)}$, the task allocation algorithm of S-PDQE ensures that the aggregated data accuracy p_c is at least $1-\exp(-\frac{1}{2}\sum_{i\in\mathcal{S}^*}\alpha_i+\mathcal{O}(\ln(n)))$, where $\mathcal{S}^*$ is the optimal winner set, which is the set of the K workers with the highest quality.

Proof: We note that $\mathcal{S}^*$ is the optimal solution to our problem. Recall that the winner set selected by our mechanism is $\mathcal{S}$. We use a sequence $\mathcal{W}$ of winners to represent the order that the winners are selected, i.e., $\mathcal{W} = w_1, w_2, \cdots, w_K$.

For each optimal winner in $\mathcal{S}^*$ and randomly selected winner in $\mathcal{W}$ that in the same position, according to Proposition 3.2, we have

$$\Pr[\alpha_{w_i} \geq \alpha_i - \mathcal{O}(\ln(n))] \geq 1 - \frac{1}{n^{\mathcal{O}(1)}}.$$

It implies that

$$\sum_{i\in\mathcal{S}^*}\alpha_i \le \sum_{w_i\in\mathcal{W}}\alpha_{w_i} + \mathcal{O}(\ln(n))$$

with a probability at least $1-\frac{1}{n^{\mathcal{O}(1)}}$. Then we have

$$\exp(-\frac{1}{2}\sum_{i\in\mathcal{S}^*}\alpha_i) \ge \exp(-\frac{1}{2}\sum_{w_i\in\mathcal{W}}\alpha_{w_i})\exp(-\mathcal{O}(\ln(n)))$$

with a probability at least $1-\frac{1}{n^{\mathcal{O}(1)}}$. According to (3.2) and (3.4) we can have

$$\begin{aligned} p_c &\ge 1-\exp(-\frac{1}{2}\sum_{w_i\in\mathcal{W}}\alpha_{w_i}) \\ &\ge 1-\exp(-\frac{1}{2}\sum_{i\in\mathcal{S}^*}\alpha_i + \mathcal{O}(\ln(n))) \end{aligned}$$

with a probability at least $1-1/n^{\mathcal{O}(1)}$. □

Next we show that the mechanism achieves differential privacy for the aggregated data.

Theorem 3.5 The data perturbation algorithm of S-PDQE achieves ϵ_d-differential privacy for the aggregated data, where $\epsilon_d > 0$ is a constant.

Theorem 3.5 can be easily proven according to the property of the randomized response mechanism that is given in Proposition 3.4.

Overall, according to the sequential composition of differential privacy, the S-PDQE mechanism achieves $(\epsilon_q(e-1)/e+\epsilon_d, \delta)$-differential privacy for workers' quality and ϵ_d-differential privacy for workers' data.

Next we show that the performance loss due to data perturbation is bounded.

Theorem 3.6 The data perturbation algorithm of S-PDQE ensures that the accuracy of the perturbed data is at least $\frac{1}{e^{\epsilon_d}+1} + \frac{e^{\epsilon_d}-1}{e^{\epsilon_d}+1}p_c$, where p_c is the accuracy of aggregated data before perturbation.

Proof: The accuracy after data perturbation p_c' is

$$\begin{aligned} p_c' &= \Pr(R(x_0)=x_0)p_c + \Pr(R(x_0)=-x_0)(1-p_c) \\ &= p_c\frac{e^{\epsilon_d}}{e^{\epsilon_d}+1} + (1-p_c)\frac{1}{e^{\epsilon_d}+1} \\ &= \frac{1}{e^{\epsilon_d}+1} + \frac{e^{\epsilon_d}-1}{e^{\epsilon_d}+1}p_c. \end{aligned}$$

□

Based on the above result, we can conclude that the data accuracy of the mechanism has a bounded gap from the optimal strategy.

Corollary 3.1 S-PDQE ensures that, with a probability at least $1 - 1/n^{O(1)}$, the overall accuracy is at least $\frac{1}{e^{\epsilon_d}+1} + \frac{e^{\epsilon_d}-1}{e^{\epsilon_d}+1}\left(1 - \exp(-\frac{1}{2}\sum_{i\in\mathcal{S}^*}\alpha_i + \mathcal{O}(\ln(n)))\right)$, where $\mathcal{S}^*$ is the optimal winner set, which is the set of K workers with the highest quality, and n is the number of workers.

3.6 Privacy-Preserving Mechanism for Truthful Data Quality Elicitation: Multi-Task Case

In this section, we consider a multi-task quality-aware crowdsourcing system, where the requester has a set of m tasks, denoted as $\mathcal{T} = \{1, 2, \ldots, m\}$. A worker has diverse quality for different tasks (e.g., because the distances between a worker and the tasks are different). A worker can be assigned to at most one task, but a task can be assigned to multiple workers. The utility of the multi-task crowdsourcing system is measured by the total accuracy of all the tasks, i.e.,

$$\sum_{j\in\mathcal{T}} p_j\left(\boldsymbol{q}', \boldsymbol{e}', \boldsymbol{d}'\right) = \sum_{j\in\mathcal{T}} E_{X_j|\boldsymbol{d}_j(\boldsymbol{q},\boldsymbol{e})}\left[\mathbf{1}_{X_j=x_j(\boldsymbol{q}',\boldsymbol{e}',\boldsymbol{d}')}\right], \tag{3.9}$$

where j denotes task j.

In the following, we present the design and analysis of a Multi-task Privacy-preserving mechanism for truthful Data Quality Elicitation (M-PDQE) that works for the multi-task setting above. M-PDQE has the same design objectives with S-PDQE, and it consists of a differentially private multi-task allocation algorithm and a differentially private data aggregation algorithm.

3.6.1 Truthful and Differentially Private Multi-Task Allocation

We first present the differentially private multi-task allocation algorithm of M-PDQE.

Design Overview. Compared with S-PDQE, since all the tasks share the same pool of workers and at most one task can be allocated to a worker, the task allocation algorithm of M-PDQE needs to be carefully designed so that the utility of crowdsourcing can be optimized.

The task allocation algorithm of M-PDQE integrates the exponential mechanism with a truthful incentive mechanism to achieve truthfulness and differential privacy. The winners are selected iteratively in this algorithm. In each iteration, each reported task-quality pair q'_{ij} ($i \in \mathcal{U}$, $j \in \mathcal{T}'$) is assigned a probability to be selected. The requester selects one winner for a task based on the probability distribution in each iteration until K_j winners are selected for task j, $\forall j \in \mathcal{T}$, where K_j is determined by the requester according to the accuracy requirement for each task j. Since the tasks are allocated independently, it is possible that more than one tasks are assigned to a worker. Thus the conflict elimination process is performed to ensure that at most one task is assigned to each winner. The process repeats until all the tasks are assigned with no conflict. In the end, the requester pays each winner according to the reward function. The detailed task allocation algorithm is presented by Algorithm 3, in which, for ease of expression, we assume $K_j = 1, \forall j \in \mathcal{T}$. We can extend the algorithm to the case of $K_j > 1$ by simply copying task j to K_j identical tasks.

Algorithm Design.

- *Phase I: Winner selection.* The exponential mechanism assigns each task-quality pair q_{ij}, $i \in \mathcal{R}$, $j \in \mathcal{T}'$, a probability of being selected. We take $f(q_{ij}) = q'_{ij}$ as the score function for any worker $i \in \mathcal{R}$ and any task $j \in \mathcal{T}'$. The probability Pr_i^j of selecting worker i as the winner of task j is

$$\Pr_i^j\left(q'_{ij}\right) = \begin{cases} \dfrac{\exp\left(\epsilon' \frac{q'_{ij}}{q^j_{\max}{}'}\right)}{\sum_{a \in \mathcal{U}} \exp\left(\epsilon' \frac{q'_{aj}}{q^j_{\max}{}'}\right)}, & \text{if } i \in \mathcal{U} \\ 0, & \text{otherwise} \end{cases} \tag{3.10}$$

  where $\epsilon' = \frac{1}{2}\epsilon/(e\Delta \ln(e/\delta))$, $\Delta = \overline{q} - \underline{q}$, and $q^j_{\max} = \max\{q_{ij}, \forall i \in \mathcal{U}\}$. We normalize q_{ij} as $\frac{q_{ij}}{q^j_{\max}}$ to ensure that the value of the score function is non-negative. The score function ensures that a task-quality pair that makes more contribution to the total accuracy will have a higher probability to be selected. We call the worker who reported the selected task-quality pair as a winner.

Algorithm 3: Multi-task allocation with reward payments

Input: The set of users $\mathcal{U}$; The set of tasks $\mathcal{T}$; Quality profile $\boldsymbol{q}$; differential privacy parameter ϵ_q and $\delta \in (0, \frac{1}{2}]$.

Output: Winner set $\mathcal{S} = \{s_j, \forall j \in \mathcal{T}\}$ and winners' reward profile $\boldsymbol{r}$.

$\boldsymbol{q}' \leftarrow \boldsymbol{q}, \mathcal{T}' \leftarrow \mathcal{T}$;

// Phase I;

foreach $i \in \mathcal{U}$ **do**

  $r_i \leftarrow 0$, $w_i \leftarrow 0$, $e'_{ij} = 0, \forall j \in \mathcal{T}$;

foreach $j \in \mathcal{T}'$ **do**

  foreach $i \in \mathcal{U}$ **do**

    Calculate the probability $\Pr_i^j \left(q'_{ij}\right)$ of each worker being selected as a winner of each task according to the score function;

  Select one worker randomly, denoted by i', according to the computed probability distribution;

  $w_{i'} \leftarrow j$, $s_j \leftarrow i'$, $e_{i'w_{i'}} \leftarrow 1$;

// Phase II;

while $\exists j \neq j'$ *s.t.* $s_j = s_{j'}$ **do**

  Perform conflict elimination;

// Phase III;

foreach $i \in \mathcal{S}$ **do**

  calculate r_i;

return $\mathcal{S}$ *and* $\boldsymbol{r}$.

- *Phase II: Conflict elimination.* The conflict elimination process ensures that at most one task is assigned to each winner without sacrificing the differential privacy provided by the exponential mechanism. The main idea is, when worker i wins for more than one task, we keep one of the winning tasks for worker i and re-assign the remaining tasks with the objective of maximizing the total accuracy. The conflict elimination process is repeated until exactly one task is assigned to each winner.

  We use an example to explain the process of conflict elimination. Suppose task 1 and task 2 are assigned to worker i in Phase I, i.e., $s_1 = s_2 = i$. According to the probability calculated from the score function, two different winners s_1' and s_2' are selected for task 1 and task 2, respectively. Then there are two possible conflict elimination situations for worker i:

$$v_1 : s_1 = i, s_2 = s_2',$$

$$v_2 : s_1 = s_1', s_2 = i.$$

  We compare the sum of reported quality of the winners in each situation and select the situation that has the highest sum. Note that a conflict elimination process may generate another conflict, and the process will be iteratively performed until there is no conflict.

- *Phase III: Reward determination.* The requester pays each winner i ($i \in \mathcal{S}$) a reward for performing task w_i, which can be expressed as

$$\begin{aligned} & r_i^{w_i}\left(q'_{iw_i}, \boldsymbol{q}'_{-iw_i}, e'_{iw_i}, d'_{iw_i}, d_{0w_i}\right) \\ & \quad = k\left[\frac{1_{d_{0w_i}=d_{iw_i}} + q_{0w_i} - 1}{2q_{0w_i} - 1} - q'_{iw_i}\right] \\ & \quad + \frac{\int_{\underline{q}}^{q'_{iw_i}} lq\Pr_i^{w_i}\left(q, \boldsymbol{q}'_{-iw_i}\right) dq}{\Pr_i^{w_i}\left(q'_{iw_i}, \boldsymbol{q}'_{-iw_i}\right)} + \frac{ce'_{iw_i}}{\Pr_i^{w_i}\left(q'_{iw_i}, \boldsymbol{q}'_{-iw_i}\right)}, \end{aligned} \tag{3.11}$$

  where $w_i \in \mathcal{T}$ is worker i's winning task, q_{0w_i} and d_{0w_i} are the requester's quality and data of task w_i respectively, e'_{iw_i} is the effort that worker i exerted in task w_i, l is a small

positive number, and k is any constant that satisfies

$$k \geq \frac{c}{(\underline{q} - \hat{q})\Pr(\underline{q})}.$$

A worker receives no reward for the tasks that are not assigned to her.

3.6.2 Multi-Task Differentially Private Data Aggregation

Next we present the differentially private data aggregation algorithm of M-PDQE. We use the randomized response mechanism (as in S-PDQE) to protect workers' data privacy. The details are given in Algorithm 4.

Algorithm 4: Multi-task data perturbation

Input: Aggregated data $\{x_1, \ldots, x_m\}$; differential privacy parameter ϵ_d.
Output: Perturbed data $\{\widehat{x}_1, \ldots, \widehat{x}_m\}$.

```
Generate R(x_1), ..., R(x_m) using the randomized response mechanism in Definition 6;
foreach j ∈ T do
    x̂_j ← R(x_j);
return {x̂_1, ..., x̂_m}.
```

Here $R(x_j)$ is defined in Definition 3.6. Algorithm 4 takes the aggregated data of tasks $\{x_1, \ldots, x_m\}$ and differential privacy parameter ϵ_d as inputs. For the aggregated data of each task, the algorithm applies the randomized response mechanism to it. The tradeoff between privacy and data accuracy is controlled by the differential privacy parameter ϵ_d.

3.6.3 Performance Analysis of M-PDQE Mechanism

Next we use thorough performance analysis to show that M-PDQE achieves computational efficiency, truthfulness, individual rationality, differential privacy, and bounded approximation gap.

First we show that the task allocation algorithm of M-PDQE is computational efficiency.

Theorem 3.7 The complexity of the task allocation algorithm of M-PDQE is $\mathcal{O}(K_{\max}mn + m^2n)$, where $K_{\max}$ is the largest number of winners a task can select, and m and n are the number of tasks and workers, respectively.

Proof: First, we can see that the outer for-loop (Line 6-12) runs at most $K_{\max}m$ rounds, and the inner for-loop (Line 7-9) runs at most n rounds. Thus, the computational complexity of obtaining the winner set without the conflict elimination is $\mathcal{O}(K_{\max}mn)$. Next, for each winner with conflicts exist, the conflict elimination performs at most $\mathcal{O}(K_{\max}mn)$ (all tasks are re-assigned). Since there are at most $K_{\max}m$ winners, the computational complexity of conflict elimination is $\mathcal{O}(K_{\max}^2m^2n)$. Therefore, the total complexity of the task allocation algorithm of M-PDQE is $\mathcal{O}(K_{\max}mn + m^2n)$.

Then we prove the truthful and individual rational properties. In this section, we assume that each user is interested in manipulating her reported quality and data and actual effort for her *winning task* to improve her expected payoff for that task, rather than *jointly* manipulating her actions for all tasks to improve her total expected payoff of *all tasks*. This is because the latter case requires the worker to know other workers' quality, which is difficult to achieve for the worker. The next result shows that our mechanism is truthful for the former case.

Theorem 3.8 The multi-task allocation algorithm with the reward payment is truthful for each worker's quality, effort, and data for her winning task, and is individually rational.

Proof: The detailed proof of truthfulness is omitted here since it's similar with that of Theorem 3.2. Worker i's expected payoff for her winning task w_i can be expressed as

$$\begin{aligned}
&E[u_i^{w_i}(\boldsymbol{q}'_{w_i}, e'_{iw_i}, e_{iw_i}, d'_{iw_i}, d_{0w_i})] \\
&= k\left[\frac{1_{d_{0w_i}=d_{iw_i}} + q_{0w_i} - 1}{2q_{0w_i} - 1} - q'_{iw_i}\right] \mathrm{Pr}_{iw_i}\left(q'_{iw_i}\right) + \\
&\quad \int_{\underline{q}}^{q'_{iw_i}} lq\mathrm{Pr}_{iw_i}\left(q, \boldsymbol{q}'_{-iw_i}\right) dq + ce'_{iw_i} - ce_{iw_i}.
\end{aligned}$$

From Lemma 3.1, 3.2, and 3.3, we can get worker i's expected payoff for task w_i, given that worker i reports her true data, quality, and makes the desired effort, can be expressed as

$$E[u_i^{w_i}(q_{iw_i}, \boldsymbol{q}'_{-iw_i})] = \int_{\underline{q}}^{q_{iw_i}} kq\mathrm{Pr}_i^{w_i}\left(q, \boldsymbol{q}'_{-iw_i}\right) dq.$$

As $\mathrm{Pr}_i^{w_i}(q_{iw_i}, \boldsymbol{q}'_{-iw_i}) \geq 0$ and it is increasing with q_{iw_i}, it can be easily seen that individual rationality is achieved. □

Then we show that the mechanism achieves differential privacy of workers' quality.

Theorem 3.9 For any constants $\epsilon_q > 0$ and $\delta \in \left(0, \frac{1}{2}\right]$, the multi-task allocation algorithm achieves $2m\epsilon_q$-differential privacy for workers' quality, where e is the base of the natural logarithm.

Proof: According to Theorem 3.3, for each task the multi-task allocation algorithm achieves $2\epsilon_q$-differential privacy. Since the tasks are assigned iteratively, the multi-task allocation algorithm is a sequential combination of the single-task allocation algorithms. Thus according to Proposition 3.3, the multi-task allocation algorithm achieves $2m\epsilon_q$-differential privacy for workers' quality, where m is the number of tasks. □

Next we show that the task allocation algorithm achieves a bounded gap from the optimal task allocation strategy when $K_j = 1, \forall j \in \mathcal{T}$.

Theorem 3.10 With a probability at least $1 - 1/n^{O(1)}$, the multi-task allocation algorithm ensures that the total data accuracy of all tasks $\sum_{j\in\mathcal{T}} p_j$ is at least $\sum_{i\in\mathcal{S}^*} q_{iw_i} - \mathcal{O}(\ln(n))$, where $\mathcal{S}^*$ is the optimal winner set.

Proof: For each winner in the optimal solution $\mathcal{S}^*$ and randomly selected winner in $\mathcal{S}$ that in the same position, according to Proposition 3.2, we have

$$\sum_{i\in\mathcal{S}^*} q_{iw_i} \leq \sum_{i\in\mathcal{S}} q_{iw_i} + \mathcal{O}(\ln(n))$$

with a probability at least $1 - \frac{1}{n^{\mathcal{O}(1)}}$, which lead to the conclusion that

$$\sum_{j\in\mathcal{T}} p_j \geq \sum_{i\in\mathcal{S}^*} q_{iw_i} - \mathcal{O}(\ln(n))$$

with a probability at least $1 - \frac{1}{n^{\mathcal{O}(1)}}$, where p_j is task j's accuracy, i.e., $\sum_{j\in\mathcal{T}} p_j = \sum_{i\in\mathcal{S}} q_{iw_i}$. □

Next we show that the mechanism achieves differential privacy for data aggregation.

Theorem 3.11 The multi-task data perturbation algorithm achieves ϵ_d-differential privacy for aggregated data, where $\epsilon_d > 0$ is a constant.

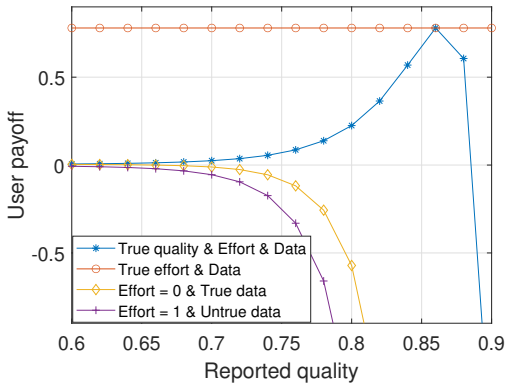

Figure 3.3: Impact of reported quality q_i'.

Theorem 3.11 can be easily proven according to the property of the randomized response mechanism that is given in Proposition 3.4. Next we show that performance loss due to the data perturbation is bounded.

Theorem 3.12 The multi-task data perturbation algorithm ensures that the accuracy of perturbed data is at least $\frac{m}{e^{\epsilon_d}+1} + \frac{e^{\epsilon_d}-1}{e^{\epsilon_d}+1}\sum_{j\in\mathcal{T}} p_j$, where m is the number of tasks.

Proof: From Theorem 3.6, the expected accuracy of task j after data perturbation p_j' is

$$p_j' = \frac{1}{e^{\epsilon_d}+1} + \frac{e^{\epsilon_d}-1}{e^{\epsilon_d}+1} p_j.$$

The summation of the expected task's accuracy can be expressed as

$$\begin{aligned}\sum_{j\in\mathcal{T}} p_j' &= \sum_{j\in\mathcal{T}} \left(\frac{1}{e^{\epsilon_d}+1} + \frac{e^{\epsilon_d}-1}{e^{\epsilon_d}+1} p_j \right) \\ &= \frac{m}{e^{\epsilon_d}+1} + \frac{e^{\epsilon_d}-1}{e^{\epsilon_d}+1} \sum_{j\in\mathcal{T}} p_j.\end{aligned}$$

□

Based on the above result, we can conclude that the data accuracy has a bounded gap from the optimal strategy.

Corollary 3.2 M-PDQE ensures that, with a probability at least $1 - 1/n^{O(1)}$, the total accuracy of all the tasks is at least $\frac{m}{e^{\epsilon_d}+1} + \frac{e^{\epsilon_d}-1}{e^{\epsilon_d}+1}\left(\sum_{i\in\mathcal{S}^*} q_{iw_i} - \mathcal{O}(\ln(n))\right)$, where $\mathcal{S}^*$ is the optimal winner set, n is the number of workers, and m is the number of tasks.

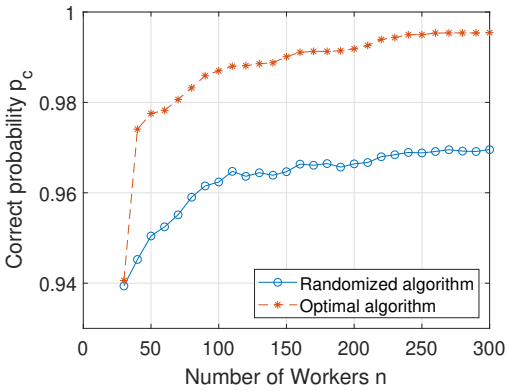

Figure 3.4: Impact of the number of workers n on S-PDQE.

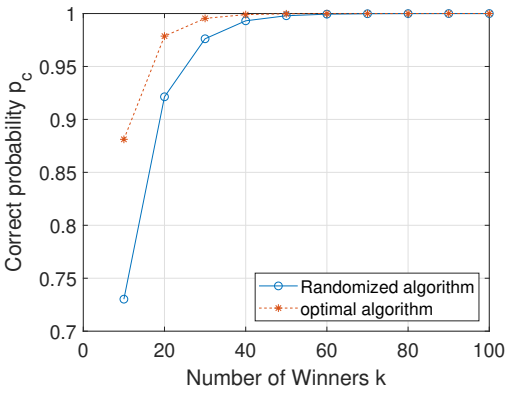

Figure 3.5: Impact of the number of winners K on S-PDQE.

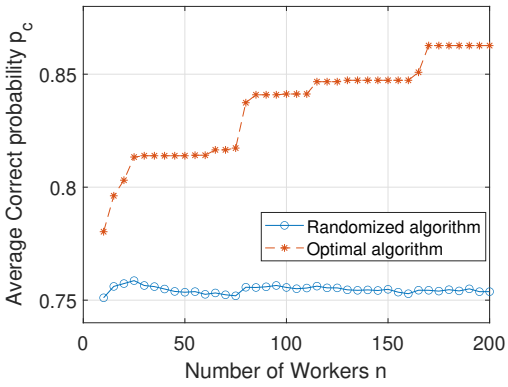

Figure 3.6: Impact of the number of workers n on M-PDQE. ($\epsilon_q = 10$)

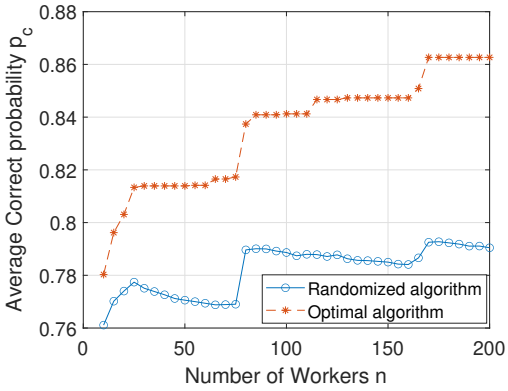

Figure 3.7: Impact of the number of workers n on M-PDQE. ($\epsilon_q = 25$)

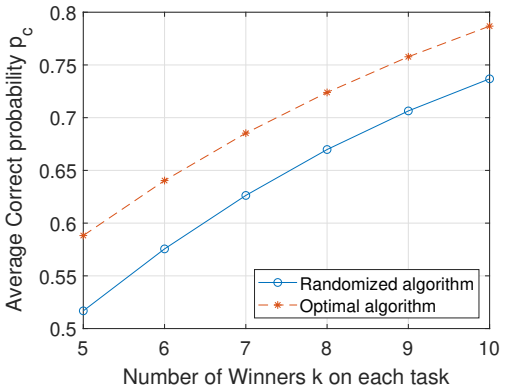

Figure 3.8: Impact of the number of winners K on M-PDQE.

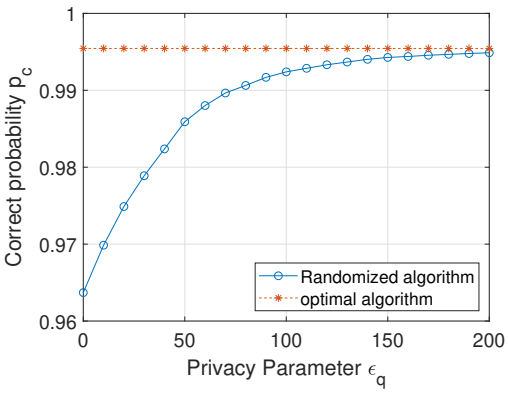

Figure 3.9: Impact of the differential privacy parameter ϵ_q on S-PDQE.

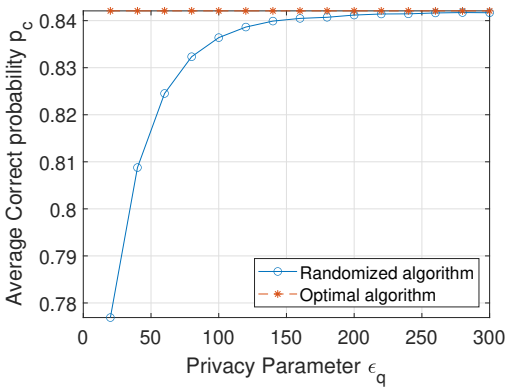

Figure 3.10: Impact of the differential privacy parameter ϵ_q on M-PDQE.

3.7 Simulation Results

In this section, we evaluate the performance of S-PDQE and M-PDQE mechanisms in terms of various metrics using simulation results. For S-PDQE, we compare it with the optimal strategy which selects the K workers with the highest quality to perform the task, without any privacy-preserving mechanism. For M-PDQE, we compare it with a greedy task allocation algorithm without a privacy-preserving mechanism. The simulation results are based on a real dataset [63] which consists of the locations of taxis scattered in the center of Rome. The crowdsourcing task is to detect whether a licensed wireless device is transmitting or not. The tasks are distributed over the area. We consider the taxis as workers. The quality of a worker for a task is calculated as a function of the worker's distance to the wireless device, with a smaller distance indicating higher quality.

3.7.1 Truthful Quality Elicitation

The system parameters are set as follows: $n = 20$, $K = 100$, $c = 0.4$, $q_i \in (1/2, 1)$, $\epsilon_q = 10$, $\delta = 0.5$, the worker's true quality $q_i = 0.86$. As we can see in Fig. 3.3, a worker can maximize her payoff when she truthfully reports her quality and data and makes her effort. When the worker reports untruthfully, her payoff is always no greater than that when she reports truthfully. This shows that the worker's optimal strategy is to behave truthfully. Also, we can see that the payoff is non-negative when the worker truthfully reports her quality, which confirms that the individual rationale property is achieved by the mechanism.

3.7.2 Data Accuracy

S-PDQE. We compare the accuracy p_c of the randomized task allocation algorithm with that of the optimal algorithm when the number of workers changes. We set the system parameters as follows: $n \in [30, 300]$, $K = 30$, $\epsilon_q = 10$, $\delta = 0.5$. The impact of number of workers on the accuracy is shown in Fig. 3.4. We can see that the accuracy of the randomized algorithm is lower than that of the optimal algorithm. This is because as the S-PDQE mechanism selects winners randomly, it cannot always select workers that have the most contributions to the accuracy. However, as the number of workers increases, the accuracy of each algorithm increases. This is because with more workers, the algorithm can find more workers with higher quality to perform the task.

We also compare the accuracy of these two algorithms when the number of winners changes. We set the system parameters as follows: $n = 300$, $K \in [10, 100]$, $\epsilon_q = 10$, $\delta = 0.5$. From Fig. 3.5, we can see that the accuracy of the randomized algorithm is lower than that of the optimal algorithm. The reason is the same as discussed earlier for Fig. 3.4. We also observe that as the number of winners increases, the accuracy gap between the two algorithms becomes smaller. The reason is that when more winners are selected, the difference between the winner sets of the randomized algorithm and the optimal algorithm becomes smaller.

M-PDQE. For M-PDQE, we evaluate the average correct probability on each task. We first compare the accuracy of the greedy algorithm and the randomized algorithm when the number of workers changes. We set the system parameters as follows: $n \in [30, 200]$, $m = 10$, $\delta = 0.5$. Fig. 3.6 and Fig. 3.7 give the results when the differential privacy parameter are $\epsilon_q = 10$ and $\epsilon_q = 25$. We see that the accuracy of the randomized is lower than that of the optimal algorithm. The reason is the same as discussed earlier for Fig. 3.4. We can also see that the accuracy of the randomized algorithm in Fig. 3.7 is growing faster than that in Fig. 3.6. This is because the higher the differential privacy parameter is, the less privacy is provided by the mechanism, and the higher probability that a worker with better quality is chosen. We also compare the accuracy of the two algorithms when the number of winners on each task changes. We set the system parameters as follows: $n = 300$, $K \in [45, 70]$, $\epsilon_q = 0.4$, $\delta = 0.5$. We observe that when there are more workers working on the same task, the average accuracy increases.

The reason is obvious since we use the weighted average rule to aggregate the data, and from Fig. 3.8 we have the expected accuracy increases when the number of winner increases.

3.7.3 Differential Privacy

We evaluate the impact of the differential privacy level on the data accuracy. We set the system parameters as follows:

	n	m	k	ϵ_q	δ
S-PDQE	300	1	30	[0,200]	0.5
M-PDQE	100	10	1	[20,300]	0.5

From Fig. 3.9 and Fig. 3.10, we observe that as the privacy parameter ϵ_q increases, the performance of the randomized algorithm improves, and the gap between the two algorithms becomes smaller. This is because as ϵ_q increases, the winner set found by the randomized algorithm becomes closer to the optimal winner set.

3.8 Conclusion

In this chapter, we devise two privacy-preserving mechanisms for truthful data quality elicitation for quality-aware crowdsourcing. For both S-PDQE mechanism and M-PDQE mechanism, we develop a task allocation algorithm which achieves truthful elicitation of workers' quality, effort, and data, and differential privacy of workers' quality. We also develop the data aggregation algorithm which achieves differentially privacy of workers' data. Besides truthfulness and differential privacy, the mechanisms also achieve approximate data accuracy maximization. We also use real data based simulations to demonstrate the desired properties of the mechanisms.

Chapter 4

Data Poisoning Attacks and Defenses in Dynamic Crowdsourcing with Online Data Quality Learning.

4.1 Introduction

Data crowdsourcing (referred to as "crowdsourcing" for brevity) leverages the "wisdom" of a potentially large crowd of workers who provide data in tasks that specified by the requester. It has found a wide range of applications including mobile sensing (such as spectrum sensing [16, 1], traffic monitoring [2, 17], environmental monitoring [18, 19, 3, 20]), and human annotation for machine learning [21] based data analytics (such as image annotation, named entity recognition). Many of these applications are enabled by smart devices with powerful sensing, networking, and computing capabilities, and the scope of these applications is expected to expand rapidly with the emerging Internet of Things (IoT). A key advantage of crowdsourcing lies in that it can exploit the diversity of inherently inaccurate data from many workers by aggregating the data obtained by the crowd, such that the data accuracy (also referred to as "data quality") after the aggregation can be substantially enhanced.

The value of crowdsourced data is determined by the quality of data collected from participating workers. A worker's data quality (referred to as "quality" for brevity) captures the accuracy of the worker's data relative to the ground truth to be estimated, and workers generally have *diverse* quality depending on a worker's characteristics and contexts (e.g., location, sensors' capabilities). For example, in spectrum sensing, a typical task is to measure the transmit signal from a wireless device. Then the signal to noise ratio (SNR) received by a worker from that device determines the worker's data quality, and workers generally have distinct SNRs depending on their locations (as illustrated in Fig. 4.1). To exploit the potential of crowdsourcing,

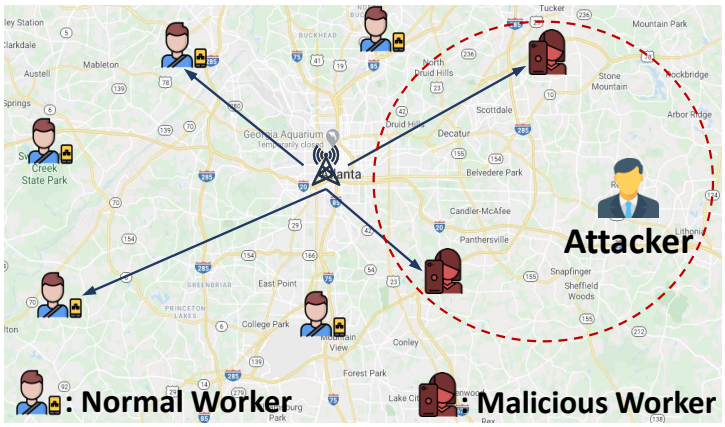

Figure 4.1: In spectrum crowdsensing, a worker's quality for a task depends on her location with respect to the wireless device to be observed. Normal workers report their true data, while malicious workers controlled by an attacker report malicious data.

it is imperative for the crowdsourcing requester to *know* workers' data quality. In particular, with the knowledge of workers' quality, the requester can assign the tasks to the workers with higher quality rather than with lower quality, so that the accuracy of collected data can be substantially improved. Thus, it is beneficial to achieve *quality-aware crowdsourcing*.

A worker's quality is often unknown to the worker and also the requester (e.g., in spectrum sensing, the location of the wireless device to be observed is unknown). In this case, the requester can learn workers' quality based on the data collected from them. In a dynamic setting of crowdsourcing where tasks are assigned to and performed by workers sequentially (e.g., in spectrum crowdsensing, tasks can arrive over time, where each task is to measure signals in a particular time slot), the requester can carry out quality learning on the fly, while making use of the learned quality information to perform task assignment and data aggregation. Such online quality learning can improve data accuracy and cost-effectiveness of crowdsourcing and is essential for the practical deployment of crowdsourcing services.

However, crowdsourcing is vulnerable to *data poisoning attacks*, where an attacker controls malicious workers to report manipulated data to the requester, typically with the goal of reducing the requester's aggregated data accuracy. Due to the random nature of workers' data and unknown ground truths of tasks, it is difficult for the requester to distinguish a malicious worker from a normal worker according to their data. It is important to note that, in order to reduce the aggregated data accuracy, malicious workers' data should deviate from normal

workers' data, so that malicious workers' quality should be lower than that of normal workers. Thus motivated, quality learning can be leveraged to find malicious workers whose quality is lower than that of the other workers, and thus exclude these malicious workers in task assignment and data aggregation.

In this chapter, we study data poisoning attacks on dynamic crowdsourcing where tasks are assigned to and performed by workers sequentially. In the presence of malicious workers, we exploit online quality learning to not only find normal workers with high quality, but more importantly to find malicious workers with low quality, as a defense mechanism against the data poisoning attack. In particular, we seek fundamental understandings of *whether and how much harm* can be made by malicious data when online quality learning is used as a defense. The design and analysis of the attacker's data poisoning strategy is highly non-trivial. First of all, the attacker needs to judiciously determine the malicious noise added to her reported data: too much noise can result in low estimated quality which prevents malicious workers' data from being used, while too little noise directly reduces the harm on the aggregated data accuracy. Therefore, the attacker should strike a *desired balance* between those two effects, in order to increase the overall harm of the attack. Moreover, the design of the attack strategy is complicated by the fact that the attacker *lacks information* of normal workers' data and also their quality. Furthermore, the effect of the attack strategy depends on both normal and malicious workers' estimated quality, which is *time-varying and inaccurate* according to the online quality learning algorithm. To further mitigate the attack, we also study the median and maximum influence of estimation based data aggregation as two defense mechanisms.

The main contributions of this chapter are summarized as follows:

- We consider malicious data attacks on dynamic crowdsourcing where tasks are assigned and performed sequentially, and propose online quality learning as a defense mechanism against the attack. We first focus on the asymptotic setting of online quality learning where workers' quality is accurately learned by the requester. In this case, we characterize the conditions under which the attack can effectively reduce the requester's aggregated data accuracy. Our analysis copes with intricate coupling between malicious workers' actual quality (which determines their data accuracy) and their quality estimated by

the requester (which determines whether they are assigned to a task). The results show that the malicious noise variance needs to be within a certain range for the attack to be effective. Based on these conditions, we analyze the harm of effective attacks.

- Based on the results in the asymptotic setting, we then study the attack in the general non-asymptotic setting, where workers' quality is inaccurately estimated over time. We first characterize the conditions under which the attack is effective. Then we analyze the online quality learning algorithm under effective attacks, which reveals that the regret can be increased substantially from $\mathcal{O}((\log T)^2)$ (upper bound) to $\Omega(T)$ (lower bound). Our analysis takes into account workers' dynamic and inaccurate estimated quality based on the online quality learning algorithm. Our findings provide useful insights on the impacts of data poisoning attacks when online quality learning is used as a defense mechanism.

- To further mitigate the attack, we propose median and maximum influence of estimation based data aggregation for the online quality learning algorithm, and analyze their performance in the asymptotic setting.

- We evaluate the proposed attack strategies and defense mechanisms using simulations based on real-world data, which demonstrate the efficacy of the attacks and defenses, and the impacts of various parameters on their performance.

The remainder of this chapter is organized as follows. Section 4.2 reviews related work. In Section 4.3, we describe a data crowdsourcing system based on online quality learning, and present malicious data attacks against this system. In Section 4.4, we propose an online quality learning algorithm and analyze its regret. In Section 4.5 and Section 4.6, we study malicious data attacks in the asymptotic regime and non-asymptotic regime of the online quality learning algorithm. In section 4.7, we propose median and maximum influence of estimation based data aggregation defenses and analyze their performance in the asymptotic regime. Simulation results are presented in Section 4.8. Section 4.9 concludes this chapter.

4.2 Related Work

4.2.1 Quality-Aware Crowdsourcing

The quality of workers' data in crowdsourcing has been studied in a few works [29, 49, 50, 43, 51]. One interesting line of works [50, 43, 51] in this direction have studied truthful mechanisms for crowdsourcing where workers have private participating costs. A few recent works [24, 64, 53] have designed truthful mechanisms for quality elicitation in quality-aware crowdsourcing. Some other works have focused on learning data quality of workers, e.g., by exploiting the correlation of their data [29, 30], or assigning tasks on the fly [49, 52]. Data poisoning attacks have been studied in [65] for crowdsourcing using offline quality learning that does not consider task assignment. It also assumes that normal workers' data are known to the attacker. Different from these works, this paper studies data poisoning in crowdsourcing based on online quality learning that learns workers' quality on the fly while assigning tasks based on workers' estimated quality. Moreover, we consider a more practical setting where the attacker does not know normal workers' data, which is more challenging.

4.2.2 Online Learning Algorithms

Online learning algorithms are generally concerned with learning some unknown parameters from past observations on the fly, while making use of the learned information to take better actions in the future. These algorithms have been used in a wide range of applications, such as the dynamic control of systems [66]. A classic branch of these algorithms are for the multi-armed bandit (MAB) problems [67]. Online data poisoning for MAB has been studied in [68], in which the attacker hijacks the behavior of MAB algorithms by manipulating selected arms' rewards. Compared to MAB algorithms such as [68, 69, 70], a key difference of the online quality learning algorithm studied here is that the learner cannot observe workers' data quality which measures the data accuracy, because the ground truth of the task is unknown to the learner. A few works [49, 52] have studied online quality learning for crowdsourcing, for which a central issue studied is the tradeoff between exploration (i.e., learning the unknown parameter) and exploitation (i.e., utilizing the learned information). Different from these works,

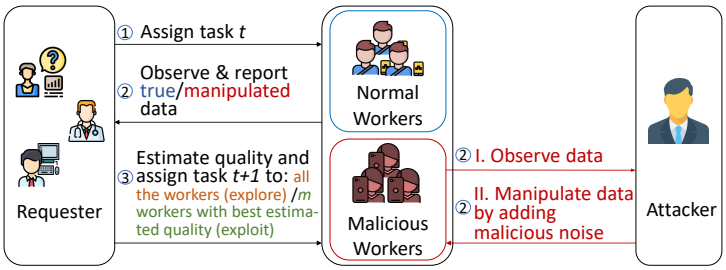

Figure 4.2: Structure and procedure of the crowdsourcing system based on online quality learning under malicious data attacks.

this paper studies malicious data attacks on crowdsourcing with the online quality learning based defense, which has not been studied before.

4.2.3 Data Poisoning Attacks and Defenses

Data poisoning is a widely used attack for machine learning [71, 72, 73, 74]. There are a few works studying data poisoning attacks on MAB algorithms [69, 70, 68], where the attacker manipulates the rewards generated from the bandit environment so that a target arm is pulled with a high probability. Online data poisoning attacks on MAB algorithms have been studied in [70, 68], in which the attacker eavesdrops on the decision of the bandit algorithm and makes an attack by manipulating the reward. Data poisoning attacks have also been studied for crowdsourcing [65, 75, 76, 77], where colluding workers aim to degrade the performance of crowdsourcing by reporting malicious data. Besides the data poisoning attack, there are also a few works on defenses against these attacks in crowdsourcing. The work most related to ours is [78], where data poisoning attacks and defenses are studied in an offline manner. Compared to these works, this paper focuses on data poisoning attacks in dynamic crowdsourcing with the online quality learning based defense. The online learning setting makes the problem studied here very different from prior works.

4.3 System Model and Problem Formulation

In this section, we first describe a dynamic crowdsourcing system, and propose an online quality learning algorithm as a defense mechanism against malicious data attacks. Then we present

Table 4.1: Main Notation

Symbol	Description
N	Number of normal workers
M	Number of malicious workers
m	Number of workers being selected in exploitation
$\mathcal{E}(t)$	Set of exploration time steps up to time step t
$D_i(t)$	Random data observed by worker i at time t
$D(t)$	Malicious workers' data before adding noise
$D'(t)$	Malicious workers' data after adding noise
$X(t)$	Ground truth at time t
$X'(t)$	Estimated ground truth by learner at time t without the attack
$X_a(t)$	Estimated ground truth by learner at time t with the attack
$X^*(t)$	Estimated ground truth by attacker with malicious workers' data at time t
p_i	Data quality of worker i
$\hat{p}_i$	Estimated quality of worker i using the actual ground truth without the attack
$\tilde{p}_i$	Estimated quality of worker i using the estimated ground truth without the attack
$\tilde{p}'_i$	Estimated quality of worker i using the estimated ground truth with the attack
$\hat{p}^*$	Estimated quality of malicious worker using the actual ground truth without the attack
$\tilde{p}^*$	Estimated quality of malicious workers using the estimated ground truth without the attack
$\tilde{p}^{*\prime}$	Estimated quality of malicious workers using the estimated ground truth with the attack

malicious data attacks on the above system. The crowdsourcing system under the attack is illustrated in Fig. 4.2.

4.3.1 Online Quality Learning Based Dynamic Crowdsourcing

Dynamic Crowdsourcing.

We consider a crowdsourcing requester recruiting a set of workers $\mathcal{N} = \{1, 2, \ldots, N\}$ to perform a sequence of crowdsourcing tasks. The crowdsourcing system operates in discrete time steps and a crowdsourcing task is allocated to and performed by workers in each time step.

Data Quality. At each time step t, a crowdsourcing task is considered, and the requester aims to estimate an unknown variable $X(t)$ by assigning a number of workers to observe $X(t)$. The variable $X(t)$ takes a continuous real value within a bounded range which is known to the requester, i.e., $X(t) \in [X_a, X_b]$, and $X_b - X_a \triangleq \Delta X$. If worker i is selected to perform the task, she observes random data $D_i(t)$, which is the sum of the interested variable $X(t)$ and an independent noise $W_i(t)$, i.e.,

$$D_i(t) = X(t) + W_i(t),$$

where $W_i(t)$'s mean is 0, and its variance $p_i \in [\underline{p}, \overline{p}]$ is unknown to the requester with $\overline{p} - \underline{p} \triangleq \Delta p$.

The quality of worker i captures the accuracy of its data compared to the ground truth $X(t)$ and is quantified by the variance p_i of $W_i(t)$. The quality of a worker is an intrinsic coefficient that captures the worker's capability for the task. Note that a *lower* p_i means *higher* quality.

Requester's Utility. The requester's crowdsourcing utility for task t is the estimation loss of her estimate $X'(t)$ compared to the ground truth $X(t)$, which is quantified by the mean squared error (MSE):

$$U(S(t)) = -E_{X'(t)|X(t),\{p_i\}_{i\in S(t)}}[(X'(t) - X(t))^2] = -\frac{\sum_{i\in S(t)} p_i}{|S(t)|^2}, \tag{4.1}$$

where $S(t)$ is the set of selected workers at time t. Note that the utility $U(S(t))$ is affected by the quality of the workers selected to perform task t.

Online Quality Learning Algorithm.

In each time step, the algorithm performs one of two functions: exploration or exploitation. In the exploration steps, to address the lack of ground truths, tasks are repeatedly assigned to all the workers in order to obtain a good estimate of the ground truth of the tasks. Specifically:

(1) In each exploration step: the requester first obtains the data from all workers and makes an estimate $X'(t)$ of the interested variable $X(t)$ using the simple average rule [1]. Then the requester estimates each worker's quality $\tilde{p}_i$ based on the estimated ground truth $X'(t)$. A worker's estimated quality is given by

$$\tilde{p}_i = \frac{\sum_{t \in \mathcal{E}(t)} \left(D_i(t) - X'(t)\right)^2}{|\mathcal{E}(t)|},$$

where $\mathcal{E}(t)$ is the set of exploration time steps up to time step t.

(2) In each exploitation step: the requester assigns a new task to the m workers who have the best estimated worker quality over the past exploration steps and makes an estimate of $X(t)$.

In each time step t, whether to perform exploration or exploitation is determined by a threshold function $h(t)$: if $|\mathcal{E}(t)| < h(t)$, then the algorithm explores; otherwise, the algorithm exploits. The threshold function $h(t)$ balances the accuracy of quality learning and the number of performed tasks by controlling the learning algorithm either performs exploration or exploitation. The detailed online quality learning algorithm and its regret analysis are given in Section 4.4.

4.3.2 Malicious Data Attack

Given the crowdsourcing system, we consider an attacker who controls a set of malicious workers $\mathcal{M} = \{1, 2, \ldots, M\}$, which along with the set of normal workers $\mathcal{N}$ forms the set of workers recruited by the requester. In each time step, the attacker estimates the ground truth $X(t)$ as $X^*(t)$ using the data observed by malicious workers. The quality of $X^*(t)$ is

$$p^* = \frac{\sum_{i \in \mathcal{M}} p_i}{M^2}.$$

[1] The requester may use other data aggregation rules (e.g., weighted average, median) to estimate the interested variable, which will be discussed in Section 4.7.

The attacker knows p^* as she knows malicious workers' quality. After obtaining $X^*(t)$, the attacker adds an artificial noise $\Delta d(t)$ to it and let all malicious workers report data as

$$D'(t) = X^*(t) + \Delta d(t).$$

where the noise $\Delta d(t)$ has a bounded range of an interval ΔD, and the mean and variance of $\Delta d(t)$ are 0 and a, respectively. The attacker's objective is to to determine the variance a of the noise $\Delta d(t)$, in order to reduce the requester's utility as much as possible, i.e., make as much harm as possible. In this chapter, we focus on the attacker's strategy that uses the same malicious noise level for all malicious workers and the same noise variance over time. This is a relatively simple but reasonable strategy: without knowing normal workers' data and quality, it is difficult for a strategic attacker to increase the harm by using more complex attack strategies (such as a time-varying noise variance). Moreover, the attack strategy above leads to non-trivial analysis, and provides useful insights for the attacks and defenses.

4.4 Online Quality Learning without Malicious Data Attack

In this section, we present the detailed online quality learning algorithm for dynamic crowdsourcing, and, as a benchmark, analyze its regret when the malicious data attack is absent.

The online learning algorithm here is an extension of that in our earlier work [52]. The major difference here is that *multiple different* tasks rather than the same single task (as in [52]) are assigned to workers in exploration time steps, which is more efficient in utilizing workers. We denote the set of tasks that have been performed in exploration steps up to time t as $K(t)$ and the set of time steps that task $k \in K(t)$ is assigned as $N_k(t)$. The algorithm perform exploration when either one of the following events occurs:

$$\mathcal{E}_1 : |K(t)| \leq B_1(t),$$
$$\mathcal{E}_2 : \exists k \in K(t), s.t., |N_k(t)| \leq B_2(t),$$

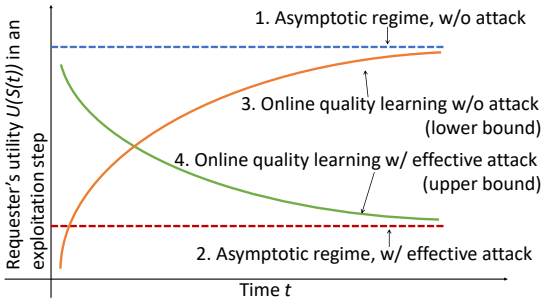

Figure 4.3: Comparison of main results.

where

$$B_1(t) = \frac{1}{(\frac{\epsilon}{|S|} - 2\epsilon\Delta X)^2} \log t,$$
$$B_2(t) = \frac{1}{\epsilon^2} \log t,$$

$|S| = m$ is the number of selected workers in exploitation steps, and $0 < \epsilon < \frac{\Delta_{min}}{2}$ is a bounded constant. $\mathcal{E}_1$ denotes the event that an insufficient number of exploration tasks have been assigned up to time t, and $\mathcal{E}_2$ denotes the event that there exists an exploration task(s) that has been assigned for insufficient times in exploration steps. From the above equations, we have that the threshold function is given by $h(t) \triangleq B_1(t)B_2(t)$. The online quality learning algorithm is described in Algorithm 5 in detail. In particular, in each exploitation step, we assign the task to the optimal set of workers based on workers' estimated quality.

Next we analyze the accumulated regret for the requester's utility.

Theorem 4.1 The regret of the online quality learning algorithm can be bounded uniformly in time (the gap between Line 1 and Line 3 in Fig. 4.3):

$$R(T) \leq \frac{\sum_{i=1}^{N} p_{i\in A'} \log^2 T}{N^2\epsilon^2 \left(\frac{\epsilon}{|S|} - 2\epsilon\Delta X\right)^2} + \Delta_{\max} \sum_{t=1}^{T} \sum_{S\subseteq\mathcal{N}} |S| \left(\frac{3}{t^2}\right),$$

where A' is the worker set that gives the largest requester's utility other than the optimal worker set.

Algorithm 5: Online quality learning for dynamic crowdsourcing

Initialization at $t = 1$: denote the first task as k, set $K(t) = \{k\}$ and $N_k(t) = \{1\}$;
At each time step t when task t arrives;
if $|K(t)| \leq B_1(t)$ *or* $\exists k \in K(t), s.t., |N_k(t)| \leq B_2(t)$ **then**
    The algorithm explores ;
    if $\exists k \in K(t), s.t., |N_k(t)| \leq B_2(t)$ **then**
        Randomly select one task in $K(t)$, denote it by k;
        Assign k to all workers $\mathcal{N}$;
        $N_k(t) = N_k(t) \cup \{t\}$;
        else
            Assign a new task to all workers $\mathcal{N}$, denote it by k;
            $K(t) = K(t) \cup \{k\}$;
    Estimate the ground truth as $X'(t) = \frac{\sum_{i=1}^{N} D_i(t)}{N}$;
    Update the estimation of ground truth as

$$X'_k(t) = \frac{\sum_{t \in N_k(t)} X'(t)}{|N_k(t)|};$$

    Update each worker's estimated quality as

$$\tilde{p}_i = \frac{\sum_{k \in K(t)} (D_i(\hat{t}) - X'_k(t))^2}{|K(t)|},$$

    where $\hat{t}$ is the time that k is last performed;
    else
        The algorithm exploits;
        Allocate a new task to the optimal workers set based on the updated estimated quality $\tilde{p}_i$;

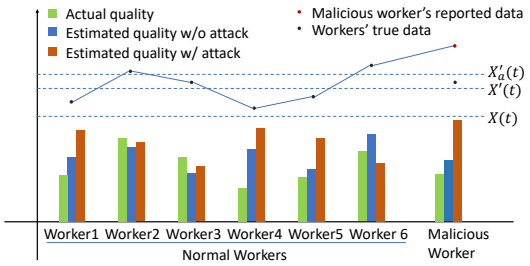

Figure 4.4: 1 malicious worker, 7 workers in total.

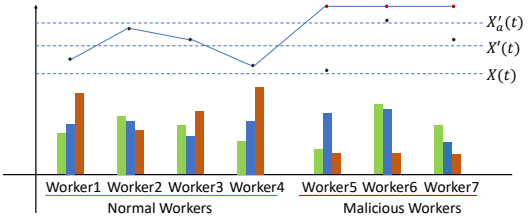

Figure 4.5: 3 malicious workers, 7 workers in total.

4.5 Malicious Data Attack with Accurate Quality Learning

In this section, we focus on malicious data attacks in the asymptotic setting of the online quality learning algorithm (i.e., when there have been infinitely many exploration time steps), where data quality of workers is learned accurately. This setting serves as a basis and also a benchmark for the general non-asymptotic setting studied in the next section. We first investigate the conditions under which the attack strategy is effective in reducing the requester's aggregated data accuracy. Based on these conditions, we then analyze the harm of effective attacks.

4.5.1 Effective Attack Conditions

To make harm to the requester, the attack strategy should have two objectives as follows.

- Malicious workers' estimated quality in exploration steps should be high in relative to that of normal workers, so that they can be selected in exploitation steps and thus their data are used by the requester.
- Meanwhile, malicious workers' actual quality should be low so that the requester's aggregated data accuracy is low.

Based on the objectives above, we provide two conditions of the attack strategy under which it can reduce the requester's data accuracy compared to the case without the attack (we say the attack is "effective" in this case).

1) Each malicious worker's *estimated* quality is *better* than the threshold worker's estimated quality, i.e., $E(\tilde{p}^{*\prime}) \leq E(\tilde{p}'_{th})$, where the threshold worker is the normal worker who has the worst quality among those selected workers in exploitation steps.

2) Each malicious worker's *actual* quality is worse than the *worst* worker in the optimal worker set, i.e., $p^* + a \geq p_m$, where a is the variance of the malicious noise $\Delta d(t)$ and p_m is the quality of mth best normal worker.

Recall that m is the number of selected workers in an exploitation step. Note that p_m is a normal worker's actual quality and is unknown to the attacker. We can see that as malicious

workers have the same estimated quality and the same actual quality after adding the malicious noise, condition 1) is a necessary condition for the attack to be effective, and condition 2) is a sufficient condition for the attack to be effective.

In Fig. 4.4 and 4.5, we use two examples to illustrate how malicious data affects workers' estimated quality in one exploration step, when the number of malicious workers varies. In both figures, most of the normal workers' estimated quality with the malicious data attack is worse than that without the attack. Fig. 4.4 shows that the malicious worker's estimated quality when she reports malicious data is worse than when she reports true data, while Fig. 4.5 shows the opposite. This is because when malicious workers' number is much less than normal workers', the malicious noise's influence on the estimated ground truth is too small, such that the malicious workers' estimated quality is reduced after adding the malicious noise. As a result, the attack in Fig. 4.4 does not satisfy condition 1). We can see that Fig. 4.5 satisfies both conditions 1) and 2).

Next we characterize the conditions of the malicious noise under which the conditions of the attack strategy above are satisfied, and show that it is a effective attack. We show the design steps in the proof of the theorem.

Theorem 4.2 In the asymptotic setting, the malicious data attack is effective if the malicious noise variance satisfies the following bounds:

$$a \leq \begin{cases} \frac{(N+M-2)p_{th}}{N-M} - p^*, & \text{if } M < N, \\ \infty, & \text{otherwise,} \end{cases} \tag{4.2}$$

and

$$a \geq p_m - p^*, \tag{4.3}$$

where $p_{th} = p_{\max\{m-M,1\}}$, and $p_{\max\{m-M,1\}}$ is the quality of the $\max\{m-M,1\}$th best normal worker.

Note that the attacker does not know normal workers' actual quality but knows its probability distribution. Thus she can infer a worker's expected quality and use it to calculate the bounds of a.

To ensure that there exists some a that satisfies (4.2) and (4.3) when $M < N$, the bounds of a should meet

$$\frac{(N+M-2)p_{th}}{(N-M)} \geq p_m. \tag{4.4}$$

Next, we show that the above inequality holds with a guaranteed probability. For ease of exposition, We assume that each worker's quality is uniformly distributed over $[\underline{p}, \overline{p}]$. Our result can be easily extended to other distributions.

Proposition 4.1 When the following condition is satisfied:

$$\frac{N+M-2}{N-M}(E(p_{th}) - \varphi) \geq E(p_m) + \varphi, \tag{4.5}$$

where $E(p_i) = \underline{p} + \frac{i}{N}\Delta p$, the condition in (4.4) holds with a probability at least $1 - 2\exp\frac{-2\varphi^2}{\Delta p^2}$ (where φ is any positive constant).

When the condition in (4.4) does not hold, the malicious data attack fails for any malicious noise since there is no such value of a that satisfies both conditions 1) and 2).

Proposition 4.2 When there are more normal workers than malicious workers (i.e., $N > M$), the upper bound of a increases with the number of malicious workers M.

Proposition 4.2 directly follows from (4.2).

Remark 4.1 The above result shows that when $N > M$, as the number of malicious workers increases, the attacker can use higher noise to make harm. When $M > N$, according to (4.2), the attacker can use arbitrarily high noise to make harm. This is because, with more malicious workers, the attacker can control the learning algorithm by dominating estimated ground truths such that malicious workers always have higher estimated quality than normal workers, so that it can use higher noise without violating condition 1).

4.5.2 Effective Attack Analysis

Since under the considered attack strategy, all malicious workers are selected in exploitation steps when the attack is effective, the selected worker set at time t consists of all the malicious

workers and $\max\{m-M,1\}$ normal workers with the best quality. We sort normal workers' quality in ascending order as $w_1, w_2, \ldots, w_N$, then we have the regret in exploitation steps:

$$\begin{aligned} Rx(T) &= \sum_{t\in e(T)} \left(U\left(S^*\right) - U(S_a)\right) \\ &= \sum_{t\in e(T)} \left(-\frac{\sum_{i=1}^{m} w_i}{m^2} + \frac{\sum_{i\in\mathcal{M}} (p^*+a) + \sum_{i=1}^{\max\{m-M,1\}} w_i}{m^2} \right) \\ &= \sum_{t\in e(T)} \left(\frac{M\left(p^*+a\right) - \sum_{i=\max\{m-M,1\}+1}^{m} w_i}{m^2} \right), \end{aligned} \tag{4.6}$$

where S_a is the selected worker set with effective attacks, and $e(T)$ is the set of exploitation steps up to time T.

Theorem 4.3 In the asymptotic setting, compared to the online learning algorithm without attacks (Fig. 4.3. Line 1), the online learning algorithm with effective attacks (Fig. 4.3. Line 2) has an expected regret of $\Omega(T)$.

Proof: The regret consists of the regret in exploration and in exploitation: $R(T) = Re(T) + Rx(T)$, and we simply bound the regret in exploration as $Re(T) > 0$. We also have $|e(T)| \geq T - h(T)$ from the definition of $|e(T)|$.

Then from (4.3) and (4.6), we can bound the regret as

$$R(T) \geq |e(T)| \left(\frac{Mw_m - \sum_{i=\max\{m-M,1\}+1}^{m} w_i}{m^2} \right) = \Omega(T).$$

□

Remark 4.2 We note from the above analysis that the regret in exploitation steps is given by (4.6). It is obvious that as the variance a increases, the regret increases. This implies that the attacker can increase the requester's estimation loss by increasing a. Therefore, this harm is maximized when a takes the upper bound value in (4.2). Thus we obtain the maximum harm by substituting the upper bound in (4.2) into (4.6).

Remark 4.3 We have from Proposition 4.2 that the upper bound of a increases as the number of malicious workers M increases. Therefore, based on (4.6), with more malicious workers, the attacker can increase the harm by using a higher noise variance a. At the same time, we see that the regret increases as M increases. This means that more harm can be made with more malicious workers even if a stays the same.

Remark 4.4 Besides the malicious noise and the number of malicious workers, another factor that affects the regret is the number of selected workers m in exploitation steps. From (4.6), we can see that the regret decreases as m increases because more normal workers are selected. As a result, the proportion of malicious workers in the selected worker set decreases, so that the influence of the attack is reduced.

4.6 Malicious Data Attack with Online Quality Learning

In this section, based on the results for the asymptotic setting in the previous section, we study the attack strategy in the general non-asymptotic setting of the online quality learning algorithm.

4.6.1 Effective Attack Conditions

To achieve effective attacks in the non-asymptotic setting, the attack strategy should satisfy the following conditions:

1) Malicious workers' *estimated quality* is *better* than that of the threshold worker, i.e., $\tilde{p}^{*\prime} \leq \tilde{p}'_{th}$.

2) Malicious workers' *actual* quality is worse than the the *worst* worker in the optimal worker set, i.e., $p^* + a \geq p_m$.

We can see that the difference of the conditions above compared to those in the asymptotic setting in Section 4.5 is that the inaccurately estimated quality instead of the accurately estimated quality of a malicious worker should be better than that of the threshold worker. The inaccurate and dynamic nature of the estimated quality poses a non-trivial challenge for the design and analysis of an effective attack strategy, as will be shown in the rest of this section.

We first provide the conditions under which an attack strategy is effective. Then we present the major steps in the proof of the result. Note that the proof has non-trivial differences from that of Theorem 1, due to the inaccurate and dynamic estimated quality of workers.

Theorem 4.4 The malicious data attack is effective with a probability at least $\mathcal{Q}$ if the malicious noise variance satisfies the following bounds:

$$a \leq \begin{cases} \frac{(N+M-2)p_{th}}{N-M} - p^* - \frac{N^2+M^2}{N^2-M^2}\epsilon_0 - \\ \frac{(N+M)^2}{N^2-M^2}(\epsilon_1 + 2\epsilon_2 + \epsilon_3 + \epsilon_4 + \epsilon_5), & \text{if } M < N, \\ \infty, & \text{otherwise,} \end{cases} \tag{4.7}$$

and

$$a \geq p_m - p^*, \tag{4.8}$$

where $p_{th} = p_{\max\{m-M,1\}}$, ϵ_0, ϵ_1, ϵ_2, ϵ_3, ϵ_4, and ϵ_5 are any positive constants, and

$$\mathcal{Q} \triangleq \max\{\omega_1 + \omega_2 - 1, 0\}, \tag{4.9}$$

where

$$\begin{aligned} \omega_1 \triangleq & 1 - \exp\left(-\frac{2|K(t)|\epsilon_0^2}{\Delta D^2}\right) - \exp\left(-\frac{2|K(t)|\epsilon_1^2}{\Delta X^2}\right) \\ & - \exp\left(-\frac{2|K(t)|\epsilon_2^2}{\Delta X^2 \Delta D^2}\right) - \exp\left(-\frac{2|K(t)|\epsilon_3^2}{\Delta p^2}\right) \end{aligned}$$

and

$$\begin{aligned} \omega_2 \triangleq & 1 - \exp\left(-\frac{2|K(t)|\epsilon_0^2}{\Delta D^2}\right) - \exp\left(-\frac{2|K(t)|\epsilon_2^2}{\Delta X^2 \Delta D^2}\right) \\ & - \exp\left(-\frac{2|K(t)|\epsilon_4^2}{\Delta p^2}\right) - \exp\left(-\frac{2|K(t)|\epsilon_5^2}{\Delta X^2}\right). \end{aligned}$$

To ensure that the upper bound of a is higher than its lower bound when $M < N$, the bounds of a should satisfy the condition below:

$$\frac{(N+M-2)p_{th}}{N-M} - \frac{N^2+M^2}{N^2-M^2}\epsilon_0 - \frac{(N+M)^2}{N^2-M^2}(\epsilon_1 + 2\epsilon_2 + \epsilon_3 + \epsilon_4 + \epsilon_5) \geq p_m. \tag{4.10}$$

Next, we show that the above inequality holds with a guaranteed probability.

Proposition 4.3 When the following condition is satisfied:

$$\frac{(N+M-2)(E(p_{th})-\varphi)}{N-M} - \frac{N^2+M^2}{N^2-M^2}\epsilon_0 \\ - \frac{(N+M)^2}{N^2-M^2}(\epsilon_1+2\epsilon_2+\epsilon_3+\epsilon_4+\epsilon_5) \geq E(p_m)+\varphi,$$

the condition in (4.10) holds with a probability at least $1-2\exp\frac{-2\varphi^2}{\Delta p^2}$.

The proof of Proposition 4.3 is omitted since it's similar to the proof of Proposition 4.1.

Proposition 4.4 When there are more normal workers than malicious workers (i.e., $N > M$), the upper bound of the variance a increases with the number of malicious workers M.

The proof of Proposition 4.4 is omitted here since it directly follows from (4.7). We can also make similar comments on Proposition 4.4 as those on Proposition 4.2 in Remark 4.1.

Proposition 4.5 When (4.10) holds, the lower bound $\mathcal{Q}$ of the probability that the malicious attack is effective increases with time and converges to 1. The upper bound of a converges to that in the asymptotic setting.

Proposition 4.5 directly follows from the definition of $\mathcal{Q}$.

Remark 4.5 The above results shows that as the number of tasks performed in exploration $|K(t)|$ increases with time, the lower bound $\mathcal{Q}$ of the probability that the attack is effective increases and eventually converges to 1. This is because as time goes, the estimated quality becomes more accurate, such that the state (including a worker's estimated quality, the set of selected workers in an exploitation step, and the requester's utility in the exploitation step) of the online quality learning algorithm becomes closer and converges to the state in the asymptotic setting in Section 4.5.

4.6.2 Effective Attack Analysis

Next we compare the regret of the online learning algorithm with effective attack in the non-asymptotic state to that in the asymptotic state in Section 4.5 (Theorem 4.5). Then we compare the regret of the online learning algorithm with effective attack to that without attacks in the asymptotic state (Theorem 4.6).

Theorem 4.5 With effective attacks, the online learning algorithm (Fig. 4.3. Line 4) has an expected regret of $\mathcal{O}(1)$ compared to in the asymptotic setting (Fig. 4.3. Line 2).

Proof: Since the regret in exploration steps is the same in both the asymptotic and the non-asymptotic states, we only need to compare the regret in exploitation steps. The regret gap is given by

$$G(T) = \sum_{t \in e(T)} (U(S(t)) - U(S_a)) \leq \Delta_{\max} E(\sum_{t \in e(T)} \mathbf{1}_{S(t) \neq S_a}), \tag{4.11}$$

where $\Delta_{\max} \triangleq \max_{S(t)}[U(S^*) - U(S(t))]$.

For ease of exposition, we use $|S|$ to express $|S(t)|$ in the rest of the proof. According to [52], we can further bound the gap by

$$G(T) \leq \Delta_{\max} \sum_{t=1}^{T} \sum_{S(t)} \sum_{i \in S(t)} \Pr(|\tilde{p_i^{'}} - E(\tilde{p_i^{'}})| > \frac{\epsilon}{|S|}). \tag{4.12}$$

Lemma 4.1 Each term in (4.12) can be bounded as

$$\Pr(|\tilde{p_i^{'}} - E(\tilde{p_i^{'}})| > \frac{\epsilon}{|S|}) \leq \frac{8}{t^2}, \forall i \in \mathcal{N} \cup \mathcal{M}.$$

Therefore, we can bound the gap between requester's utility in the asymptotic state and in the non-asymptotic state as

$$G(T) \leq \Delta_{\max} \sum\nolimits_{t=1}^{T} \sum\nolimits_{S(t)} |S| \frac{8}{t^2} = \mathcal{O}(1). \tag{4.13}$$

□

Remark 4.6 In the proof of Theorem 4.5, we can see from (4.13) that the upper bound on the regret gap in an exploitation step decreases with time and converges to 0. This is because as time goes, a worker's estimated quality becomes more accurate, and converges to the exact quality learned in the asymptotic setting (note that this is not the worker's actual quality but the quality learned from workers' data in the requester's view). As a result, the set of selected workers is more likely to be the same as that in the asymptotic setting (which with effective attacks includes all malicious workers). Therefore, the regret gap in an exploitation step also decreases and converges to 0 as time goes. Moreover, this decrease with time is sufficiently fast, such that the total utility gap $G(T)$ over all exploitation steps up to time T is in the order of $\mathcal{O}(1)$, which means it is upper bounded by a constant independent of time.

Theorem 4.6 Compared to the online learning algorithm in the asymptotic setting without attacks (Fig. 4.3. Line 1), the online learning algorithm with effective attacks (Fig. 4.3 Line 4) has an expected gap of $\Omega(T)$.

Remark 4.7 From Theorem 4.4, we can see that all malicious workers are selected in exploitation steps with a probability greater than $\mathcal{Q}$. Thus we can have

$$\begin{aligned} R(T) \geq & \sum_{t \in e(T)} \left(U(S^*) - U(S(t))\right) \geq \Delta_{\min} E\Big(\sum_{t \in e(T)} \mathbf{1}_{S(t) \neq S^*}\Big) \\ \geq & \Delta_{\min}(\mathcal{Q} - 2\exp\left(-2\varphi^2\right))(T - h(T)), \end{aligned}$$

where

$$\begin{aligned} \Delta_{\min} \triangleq & \min_{S(t)} \left[U\left(S^*\right) - U(S(t))\right] = U\left(S^*\right) - U(S_a) \\ = & U\left(S^*\right) + (M(p^* + a) + \sum_{i=1}^{\max\{m-M,1\}} w_i)/m^2. \end{aligned}$$

It is obvious that as a increases, $\Delta_{\min}$ increases. We can also see that the lower bound of the regret increases with $\Delta_{\min}$. This means that the attacker can make more harm by increasing a (as long as a is no greater than its upper bound).

Remark 4.8 As in Fig. 4.3, compared to the offline optimal strategy (Line 1), the upper bound on the regret of the requester's utility is increased from $\mathcal{O}(\log^2 T)$ (Line 3) to $\Omega(T)$ (Line 4),

as a result of effective attacks. This is essentially due to that in the asymptotic setting, the requester's utility in an exploitation step is reduced by a constant (from Line 1 to Line 2). The above observation shows that malicious attacks can make substantial harm, despite that workers' quality is estimated with errors for online quality learning.

4.7 Data Aggregation Defenses

The online quality learning algorithm serves as a defense mechanism against the data poisoning attack, as it learns workers' quality and finds malicious workers with low quality. Note that the data aggregation rule used in the online quality learning algorithm has substantial impacts on the quality learning results (as it affects the estimated ground truth and thus the estimated quality). In this section, we integrate median and maximize influence estimation (MIE) based data aggregation for the online quality learning algorithm, and analyze their effects in the asymptotic setting. We will evaluate their performance in the non-asymptotic setting in Section 4.8.

4.7.1 Median

In the online quality learning algorithm, instead of using simple average based data aggregation, the requester takes the median value of submitted data as the estimated value of the ground truth as in (2), i.e., the estimated ground truth at time t is

$$X'(t) = Median\left\{D_i(t), i \in S(t)\right\}. \tag{4.14}$$

Theorem 4.7 With the median-based data aggregation, the malicious data attack is effective with any malicious noise variance a if the number of malicious workers is greater than that of normal workers (i.e., $M \geq N$).

Proof: It is intuitive that when the number of malicious workers is greater than that of normal workers, i.e. $M \geq N$, the estimated ground truth (median) constantly equals the malicious data since all malicious workers submit the same data value. Thus, malicious workers' empirical variances are always the lowest among all workers and estimated quality is always the highest. Therefore, the attacker can use an arbitrarily high malicious noise to make harm. □

When the number of malicious workers is less than that of normal workers ($M < N$), it is difficult to characterize the impacts of malicious data on the estimated ground truths $X'(t)$ and estimated quality of workers. We will use simulation result to analyze this case in Section 7.6.

4.7.2 Maximize Influence of Estimation

Next we study the malicious data attack under the maximize influence of estimation (MIE) based defense proposed in [78]. The general idea of MIE defense is to find the workers with the maximum influence on the estimated ground truth and exclude them when selecting workers in exploitation steps. MIE uses a greedy influential worker selection algorithm to detect the influential worker set which contains the workers with the maximum influence on the estimated ground truth. We follow the assumption in [78] that the requester knows the number of malicious workers. We first present the greedy influential worker selection algorithm in Algorithm 6. Then we analyze the effectiveness of the attack under the MIE defense in the asymptotic setting.

Algorithm 6: Greedy influential worker selection

```
Input: Workers' data in exploration D_i(t) for i ∈ N ∪ M, t ∈ E(t), number of
  malicious workers M;
Initialize the influential worker set A = ∅;
while |A| < M do
    Select i = arg max_{i∈M\A} φ(i, K(t));
    A ← A ∪ {i};
return Influential worker set A.
```

The influence of worker i is defined as follows:

$$\varphi(i, K(t)) \triangleq \frac{\sum_{t\in\mathcal{E}(t)} \left(\frac{\sum_{j\in\mathcal{N}\cup\mathcal{M}} D_j(t)}{N+M} - \frac{\sum_{j\in\mathcal{N}\cup\mathcal{M}\setminus\{i\}} D_j(t)}{N+M-1} \right)^2}{|\mathcal{E}(t)|} \tag{4.15}$$

Theorem 4.8 Under the MIE defense, the malicious data attack is effective if the malicious noise variance satisfies the bounds given by (4.2) and (4.3).

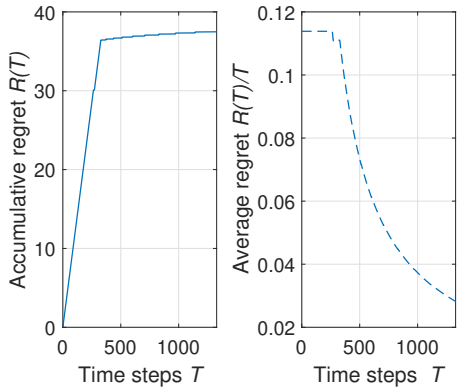

Figure 4.6: Regret of the multi-task-explore online learning algorithm.

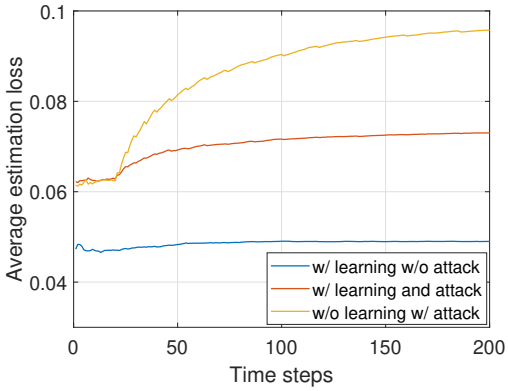

Figure 4.7: Impact of quality learning and attack on data accuracy with accurate quality learning.

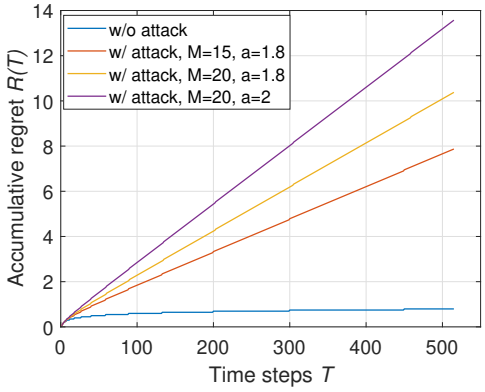

Figure 4.8: Impact of the number of malicious workers M with accurate quality learning.

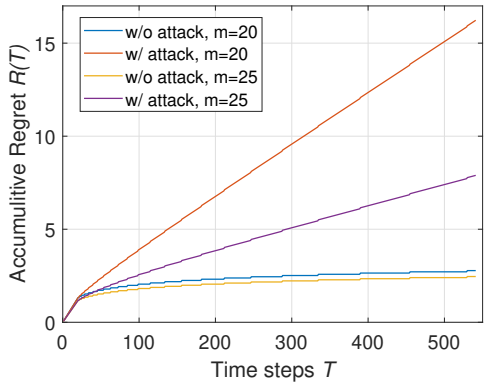

Figure 4.9: Impact of the number of selected workers m in exploitation with accurate quality learning.

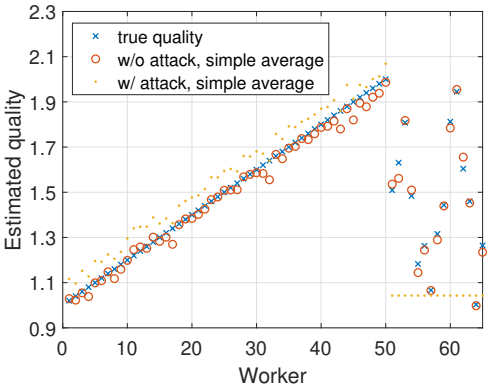

Figure 4.10: Learned quality with online quality learning.

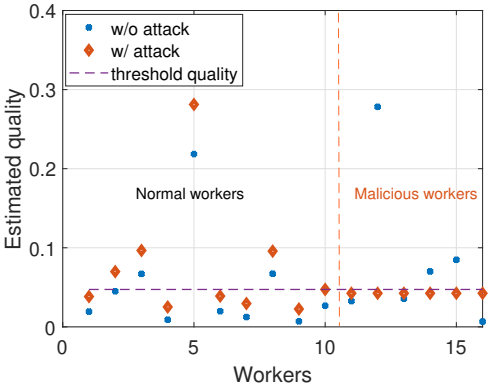

Figure 4.11: Learned quality with online quality learning. (real-world data)

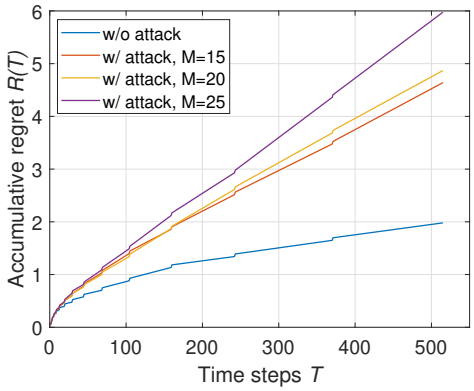

Figure 4.12: Impact of the number of malicious workers M with online quality learning.

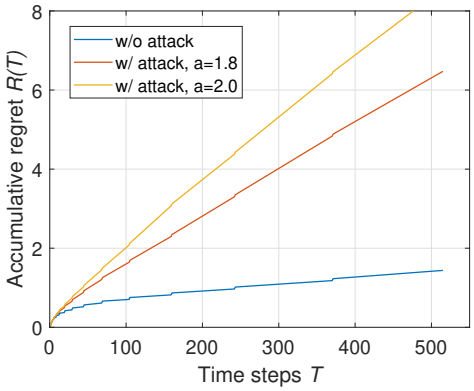

Figure 4.13: Impact of the variance of added noise a with online quality learning.

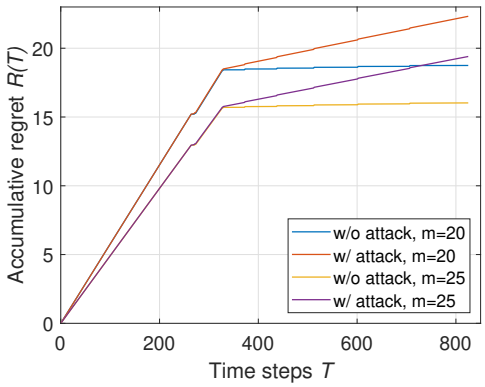

Figure 4.14: Impact of the number of selected workers in each exploitation m with online quality learning.

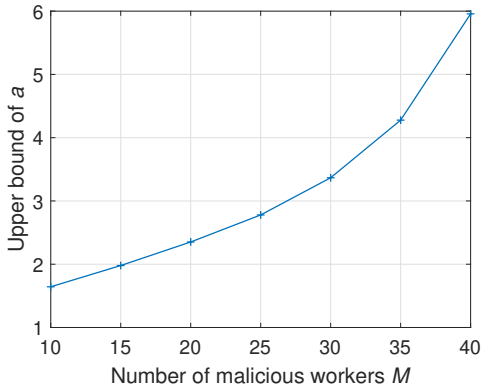

Figure 4.15: Impact of the number of malicious workers M on the upper bound of a.

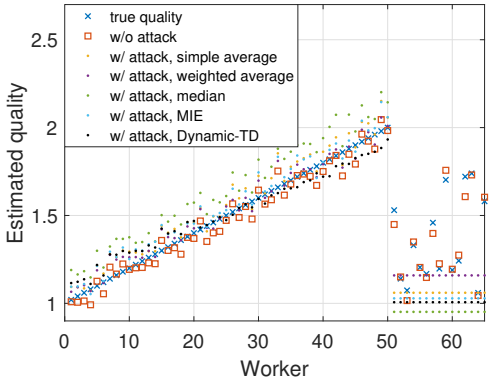

Figure 4.16: Learned quality with online quality learning under different defenses.

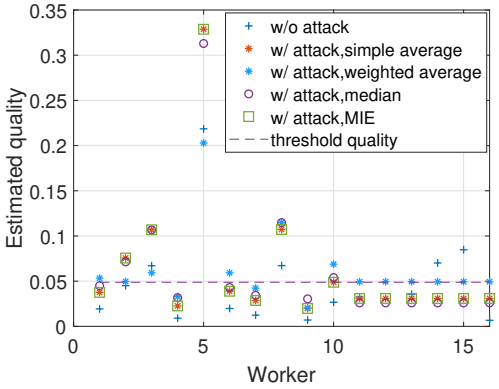

Figure 4.17: Learned quality with online quality learning under different defenses. (real-world data)

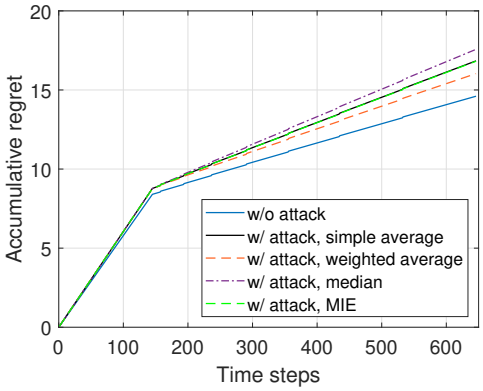

Figure 4.18: Impact of different defenses on the accumulative regret.

4.8 Performance Evaluation

In this section, we use simulations based on both synthetic data and real-world data to evaluate the performance of the attack strategies and defense mechanisms. In the synthetic dataset, the quality of normal workers is linearly distributed over $[1, 2]$, and the quality of malicious workers is uniformly distributed over $[1, 2]$. There are 500 tasks of which the ground truth values $X(t)$ are uniformly distributed over $[1, 2]$. The data of worker i for task t follows a normal distribution $D_i(t) \sim \mathcal{N}(X(t), p_i)$, where p_i is worker i's quality. The real-world data is Weather, which contains the weather data on 30 major USA cities from 18 websites every 45 minutes on a day in Mar, 2010. We use 224 data samples from each of 16 websites, respectively, as 224 tasks and 16 workers.

4.8.1 Online Quality Learning Without Attack

We first evaluate the online quality learning algorithm proposed in Section 4.4. The crowdsourcing system parameters using synthetic data are set as follows: $N = 30$, $m = 20$, $\epsilon = 0.4$. Fig. 4.6 illustrates the algorithm's accumulative regret and average regret versus time steps. We can see that the accumulative regret increases as a logarithmic function with time steps and the average regret converges to zero. These observations conform our theoretical results.

4.8.2 Attack with Accurate Quality Learning

We evaluate the impact of online quality learning and the malicious data attack on crowdsourcing data accuracy. We study the requester's estimation loss under three circumstances: online quality learning without the attack, online quality learning with the attack, and no quality learning with the attack (the requester randomly select workers in exploitation). The crowdsourcing system parameters using synthetic data are set as follows: $N = 30$, $M = 15$, $m = 25$, $\epsilon = 0.4$. The variance of additive noise a takes the value of its upper bound when the attack is applied. We can see in Fig. 4.7 that, as online quality learning is used, the requester's estimation loss is larger with the attack, which demonstrates that the attack makes harm. However, we can also observe that, when the system is attacked, the requester's estimation loss with online quality learning is always smaller than that without online quality learning, which shows that online quality learning can mitigate the attack.

We evaluate the impact of the number of malicious workers M, the variance of additive noise a, and the number of selected workers m in each exploitation step on the accumulative regret in the asymptotic regime of the online quality learning algorithm. The crowdsourcing system parameters using synthetic data are set as follows: $N = 30$, $M = 15$, $m = 20$, $\epsilon = 0.4$. Fig. 4.8 shows the regret when the attacker takes different attack strategies. We can observe that the regret increases as either the number of malicious workers M or the variance of additive noise a increases, which agrees with Remarks 4.2 and 4.3. In Fig. 4.9, we can see that the regret always decreases as the number of selected workers m increases. This is because the utility loss decreases when more workers work on a task. We also see that the accumulative regret with the attack is always higher than that without the attack.

4.8.3 Attack with Online Quality Learning

We first investigate the impact of the data poisoning attack on the estimated quality of workers. The crowdsourcing system parameters using synthetic data are set as follows: $\mathcal{N} = \{1, 2, \ldots, 50\}$, $\mathcal{M} = \{51, 52, \ldots, 65\}$, $m = 20$, $\epsilon = 0.55$, $\epsilon_0 = 0.2$, $\epsilon_3 = \epsilon_4 = 0.1$, $a = 2$. The crowdsourcing system parameters using synthetic data are set as follows: $\mathcal{N} = \{1, 2, \ldots, 10\}$,

$\mathcal{M} = \{11, 12, \ldots, 16\}$, $m = 10$, $\epsilon = 0.4$, $a = 7$. Both simulation results using synthetic data (Fig. 4.10) and real-world data (Fig. 4.11) show that malicious workers' estimated quality is good enough to be selected in exploitation steps. For both settings, we compare the data accuracies of tasks with and without the malicious data attack, respectively. The data accuracy is calculated as

$$Accuracy = \frac{\sum_k |X'_k - X_k|}{\sum_k X_k},$$

where k is the task index, X'_k and X_k are the estimated ground truth and the actual ground truth of task k, respectively. For synthetic data simulation, the data accuracies with and without the malicious data attack are **0.47** and **0.14** respectively. Since the actual ground truths of tasks are not provided in the real-world dataset, we use the average of all workers' data as the "actual ground truth". For real-world data simulation, the data accuracies with and without the malicious data attack are **0.007** and **0.004** respectively. We can see that the data accuracies decrease with the attacks. Although both data accuracies with and without the malicious data attack seem good. However, this is because we do not have the actual ground truth, i.e., X_k, for the real-world data and use the average of all workers' data as the "actual ground truth". Since X'_k and X_k are obtained from the same dataset with different numbers of data samples, the data accuracies seem good and close. If we calculate the data accuracy using actual ground truth, the data accuracy values can be worse than the presented results. Moreover, our result shows that the data accuracy with the malicious attack is almost twice of that without the malicious attack. Thus, this result demonstrates that the malicious data attack is effective.

We also compare the accumulative regret of the online learning algorithm while the attack strategy takes a various number of malicious workers M (Fig. 4.12), a different variance of the added noise a (Fig. 4.13), and a various number of selected workers m in each exploitation steps (Fig. 4.14), respectively, using synthetic data. The crowdsourcing system parameters are set as follows: $N = 30$, $M = 20$, $m = 25$, $\epsilon = 0.55$, $\epsilon_0 = 0.2$, $\epsilon_3 = \epsilon_4 = 0.1$, $a = 2$. In Fig. 4.12, we compare the accumulative regret of the online learning algorithm while the number of malicious workers M varies. We can observe that the accumulative regret is linear and increases as the number of malicious workers increases with the attack. In Fig. 4.13, we

can see that the attacker can make more harm by increasing the variance of added noise a. In Fig. 4.14, we observe that the regret reduces as more workers are selected in each exploitation step. From the three figures we mentioned above, we can conclude that the attack strategy is effective, and the attack can make more harm to the online learning algorithm by recruiting more malicious workers and/or increasing the variance of added noise (as long as a is lower than its upper bound). We also study the upper bound of added noise a of an effective attack that is shown in Fig. 4.15. It shows that the upper bound increases as the number of malicious workers increases.

4.8.4 Data Aggregation Defenses

We evaluate the impact of malicious attacks when different defenses are used for the crowdsourcing system. We compare the estimated quality and regret when using simple average aggregation, weighted average aggregation, median aggregation, MIE in the online quality learning algorithms, respectively. The estimated ground truth using weighted average aggregation is $X'(t) = \frac{1}{\tilde{p}_i} D_i(t) / \sum_{i \in S(t)} \frac{1}{\tilde{p}_i}$. The crowdsourcing system parameters are set as the same as evaluating the malicious data attack with online quality learning. Fig. 4.16 and Fig. 4.17 shows the estimated quality after learning when performing different defenses using synthetic data and real-world data. We observe from both figures that with the malicious data attack, compared to using simple average aggregation, using weighted average aggregation can mitigate the attack, using MIE defense gives the same result as the simple average aggregation, and using median aggregation can aggravate the attack. In Fig. 4.16, We also compare the estimated quality when using defenses in this chapter and the defense (Dynamic-TD) proposed in [79]. We observed that the result of Dynamic-TD is close to that of the MIE defense in this chapter. The same results are shown in Fig. 4.18, where we compare the accumulative regret when using different defenses. The reasons for the above results are as follows. 1) Since the weighted aggregation rule uses the reciprocal of workers' estimated quality as weights, as the learned quality converges to the actual quality, estimated ground truths of tasks are more accurate compared to using simple average aggregation. Thus, with more accurate estimated

ground truths, learned quality is closer to the actual quality than using simple average aggregation. 2) For MIE defense, a worker's influence is determined by the deviation from her data to the average of other workers' data, which uses the same idea as the online quality learning algorithm estimates workers' quality. Thus, the order of workers' influences is the same as the order of workers' estimated quality, such that malicious workers are still selected in exploitation steps. 3) When using the median aggregation rule, since malicious workers' data are the same, the median shifts towards malicious workers' data. Thus, malicious workers' estimated quality is better when using median aggregation than when using simple average aggregation. More importantly, the estimated ground truths are more inaccurate compared to using simple average aggregation. Hence, using median aggregation can aggravate the attack.

4.9 Conclusion

In this chapter, we explore the malicious data attack on the online quality learning algorithm in data crowdsourcing. We first study the design of the malicious data attack in the asymptotic regime and non-asymptotic regime of the online quality learning algorithm and discuss the impacts of various parameters on the effect of the attack. We show that the requester's accumulative regret of online quality learning can be increased from $O(\log^2 T)$ to $\Omega(T)$ due to the attack. We use simulation to evaluate the proposed malicious data attack and the impact of various parameters.

4.10 Appendix

4.10.1 Proof of Theorem 4.1

We use several steps to prove the above theorem.

Step 1: The regret consists of two parts: regret in exploration and regret in exploitation:

$$R(T) = Re(T) + Rx(T).$$

Step 2: Using union bound we have the following bound for $Re(T)$ and $Rx(T)$ respectively at time T:

The regret from the exploration steps is bounded by

$$Re(T) \leq -U(A')D_1(T)D_2(T),$$

where $D_1(T)D_2(T)$ is the upper bound of the number of exploration steps.

According to [52], the regret from the exploitation steps is bounded by

$$Rx(T) \leq \Delta_{\max} \sum_{t=1}^{T} \sum_{S \subseteq \mathcal{N}} \sum_{i \in S} \Pr(|\tilde{p}_i - p_i| > \frac{\epsilon}{|S|}). \tag{4.16}$$

Step 3: We further bound the regret from exploitation steps. Consider each term in (4.16),

$$\begin{aligned}
&\Pr(|\tilde{p}_i - p_i| > \frac{\epsilon}{|S|}) = \\
&\underbrace{\Pr(|\tilde{p}_i - p_i| > \frac{\epsilon}{|S|} | X'_k(t) - X(t) \leq \epsilon)}_{\textbf{Term } 1} \Pr(X'_k(t) - X(t) \leq \epsilon) \\
&+ \Pr(|\tilde{p}_i - p_i| > \frac{\epsilon}{|S|} | X'_k(t) - X(t) > \epsilon) \underbrace{\Pr(X'_k(t) - X(t) > \epsilon)}_{\textbf{Term } 2},
\end{aligned}$$

where k and $X(t)$ are the task performed at time step t and its ground truth.

For **Term 1**, under the condition that $X'_k(t) - X(t) \leq \epsilon$, we have the difference of the estimated quality when compare worker's data with the ground truth $\hat{p}_i$ and with the estimated ground truth $\tilde{p}_i$ is

$$\begin{aligned}
|\hat{p}_i - \tilde{p}_i| &= |\frac{\sum_{k \in K(t)} (D_i(t) - X(t))^2 - (D_i(t) - X'_k(t))^2}{|K(t)|}| \\
&= |\frac{\sum_{k \in K(t)} (2D_i(t) - X(t) - X'_k(t))(X'_k(t) - X(t))}{|K(t)|}| \\
&\leq 2\Delta X \epsilon.
\end{aligned} \tag{4.17}$$

Thus, from (4.17) and Hoeffding's inequality, we have

$$
\begin{aligned}
&\textbf{Term 1}\ \Pr(X'_k(t) - X(t) \leq \epsilon) \\
&\leq \Pr(|\tilde{p}_i - \hat{p}_i| + |\hat{p}_i - p_i| > \frac{\epsilon}{|S|} | X'_k(t) - X(t) \leq \epsilon) \times \Pr(X'_k(t) - X(t) \leq \epsilon) \\
&\leq \Pr(|\hat{p}_i - p_i| > \frac{\epsilon}{|S|} - 2\Delta X \epsilon) \\
&\leq 2\exp\left(\frac{-2(\frac{\epsilon}{|S|} - 2\Delta X \epsilon)^2 |K(t)|}{\Delta p^2}\right) \leq \frac{2}{t^2}.
\end{aligned}
$$

For **Term 2**, we have

$$
\Pr(X'_k(t) - X(t) > \epsilon) \leq \exp\left(\frac{-2\epsilon^2 |N_k(t)|}{\Delta X^2}\right) \leq \frac{1}{t^2}.
$$

Summing up, we can bound (4.16) as

$$
Rx(T) \leq \Delta_{\max} \sum_{t=1}^{T} \sum_{S \subseteq \mathcal{N}} |S| \frac{3}{t^2}.
$$

4.10.2 Proof of Theorem 4.2

First we can see that (4.3) follows from condition 2). Next we use 3 steps (Steps 1, 2, and 3) to show that (4.2) achieves condition 1).

Step 1: We need to find the threshold that can ensure all malicious workers to be selected in exploitation, which determines the upper bound for malicious workers' estimated quality. First, the expected estimated quality of a normal worker i with the attack is

$$
\begin{aligned}
E\left(\tilde{p}'_i\right) &= \sum_{t \in K(t)} E\left(\left(D_i(\hat{t}) - X_a(t)\right)^2\right) / |K(t)| \\
&= p_i \frac{N + M - 2}{N + M} + \frac{\sum_{j \in \mathcal{N} \cup \mathcal{M}} p_j}{(N + M)^2} + a\left(\frac{M}{N + M}\right)^2.
\end{aligned}
\tag{4.18}
$$

From the equation above we have that $E(\tilde{p}'_i) < E(\tilde{p}'_j)$, if $p_i < p_j$. This means that all malicious workers can be selected in exploitation steps if their estimated quality is better than that of the $\max\{m - M, 1\}$th best normal worker. Thus we set the threshold worker as the

$\max\{m-M,1\}$th best normal worker. We express condition 1) as:

$$E(\tilde{p}^{*\prime}) \leq E(\tilde{p}'_{th}) = E(\tilde{p}'_{\max\{m-M,1\}}). \tag{4.19}$$

Step 2: Similar as (4.18), for a malicious worker, we have

$$E\left(\tilde{p}^{*\prime}\right) = (p^* + a)\frac{N-M}{N+M} + \frac{\sum_{j\in\mathcal{N}\cup\mathcal{M}} p_j}{(N+M)^2} + a\left(\frac{M}{N+M}\right)^2. \tag{4.20}$$

Step 3: By substituting (4.18) and (4.20) into (4.19), we have that the attack meets condition 1) when a is bounded by (4.2).

4.10.3 Proof of Proposition 4.1

Since workers' quality follows the uniform distribution, the expected quality of the ith best worker is $E(p_i) = \underline{p} + \frac{i}{N}\Delta p$.

From Hoeffding's inequality, we know that event $\{\zeta_1 : p_{th} < E(p_{th}) - \varphi\}$ is true with a probability at most $\exp(\frac{-2\varphi^2}{\Delta p^2})$, and so is event $\{\zeta_2 : p_m > E(p_m) + \varphi\}$. Thus from De Morgan's laws and Boole's inequality, we have

$$\Pr(\zeta_1' \cap \zeta_2') = 1 - \Pr(\zeta_1 \cup \zeta_2) \geq 1 - 2\exp(-2\varphi^2/\Delta p^2),$$

where ζ_1' and ζ_2' are the complement events of ζ_1 and ζ_2, respectively. Therefore, when (4.5) is true, the condition in (4.4) holds with a probability at least $1 - 2\exp(\frac{-2\varphi^2}{\Delta p^2})$.

4.10.4 Proof of Theorem 4.4

First we can see that (4.8) follows from condition 2). Next we use 3 steps (Steps 1, 2, and 3) to show that (4.7) achieves condition 1).

Step 1: Similar as Theorem 4.2, we express condition 1) as:

$$\tilde{p}^{*\prime} \leq \tilde{p}'_{\max\{m-M,1\}} = \tilde{p}'_{th}, \forall i \in \mathcal{M}. \tag{4.21}$$

Step 2: To achieve (4.21), we give the following two lemmas.

Lemma 4.2 For a normal worker's estimated quality based on the estimated ground truth $\tilde{p}_i'$ and her actual quality p_i, we have that

$$\begin{aligned}\tilde{p}_i' \geq & \frac{N+M-2}{N+M}p_i + \frac{\sum_{j\in\mathcal{N}\cup\mathcal{M}} p_j}{(N+M)^2} + \\ & (\frac{M}{M+N})^2(\frac{a}{|N_k(t)|} - \epsilon_0) - \epsilon_1 - 2\epsilon_2\frac{M}{N+M} - \epsilon_3\end{aligned} \tag{4.22}$$

holds with a probability greater than ω_1.

Lemma 4.3 For a malicious worker's estimated quality based on the estimated ground truth $\tilde{p}^{*\prime}$ and her actual quality p^*, we have that

$$\begin{aligned}\tilde{p}^{*\prime} \leq & \frac{N-M}{N+M}p^* + \frac{\sum_{j\in\mathcal{N}\cup\mathcal{M}} p_j}{(N+M)^2} + \\ & (\frac{N}{N+M})^2(\frac{a}{N_k(t)} + \epsilon_0) + 2\epsilon_2\frac{N}{N+M} + \epsilon_4 + \epsilon_5\end{aligned} \tag{4.23}$$

holds with a probability greater than ω_2.

Step 3: By substituting (4.22) and (4.23) into (4.21), and from $|N_k(t)| \geq 1$, we have when the noise variance a is bounded by (4.7), the attack strategy is effective with a probability greater than $\mathcal{Q}$.

4.10.5 Proof of Theorem 4.6

From Theorem 4.3 and 4.5, we have

$$\begin{aligned}R(T) &\geq \sum\nolimits_{t\in e(T)} (U(S^*) - U(S(t))) \\ &= \sum\nolimits_{t\in e(T)} ((U(S^*) - U(S_a)) - (U(S(t)) - U(S_a))) \\ &= \Omega(T) - \mathcal{O}(1) = \Omega(T)\end{aligned}$$

4.10.6 Proof of Theorem 4.8

According to (4.15), worker i's influence is

$$\begin{aligned}
&\varphi(i, K(t)) \\
=&\frac{\sum_{t\in\mathcal{E}(t)}\left(\frac{\sum_{j\in\mathcal{N}\cup\mathcal{M}} D_j(t)}{N+M} - \frac{\sum_{j\in\mathcal{N}\cup\mathcal{M}\setminus\{i\}} D_j(t)}{N+M-1}\right)^2}{|\mathcal{E}(t)|} \\
=&\frac{1}{|\mathcal{E}(t)|}\sum_{t\in\mathcal{E}(t)}\left(\frac{(N+M)(\sum_{j\in\mathcal{N}\cup\mathcal{M}} D_j(t) - \sum_{j\in\mathcal{N}\cup\mathcal{M}\setminus\{i\}} D_j(t))}{(N+M)(N+M-1)}\right. \\
&\left. -\frac{\sum_{j\in\mathcal{N}\cup\mathcal{M}} D_j(t)}{(N+M)(N+M-1)}\right)^2 \\
=&\frac{1}{|\mathcal{E}(t)|}\sum_{t\in\mathcal{E}(t)}\left(\frac{D_i(t) - \frac{\sum_{j\in\mathcal{N}\cup\mathcal{M}} D_j(t)}{N+M}}{N+M-1}\right)^2 \\
=&\frac{\sum_{t\in\mathcal{E}(t)}\left(D_i(t) - X_a(t)\right)^2}{|\mathcal{E}(t)|(N+M-1)^2}.
\end{aligned}$$

Thus, we have worker i's expected influence is

$$\begin{aligned}
E(\varphi(i, K(t))) =&\frac{\sum_{t\in\mathcal{E}(t)}\left(D_i(t) - X_a(t)\right)^2}{|\mathcal{E}(t)|(N+M-1)^2} \\
=&\frac{E(\tilde{p}_i')}{(N+M-1)^2}.
\end{aligned} \tag{4.24}$$

We can see that a worker's expected influence is proportional to her expected estimated quality from the online quality learning algorithm. Thus, the worker selection result according to the influence is the same as that according to the estimated quality. We have proved in Theorem 4.2 that the malicious data attack is effective if the variance of the malicious noise satisfies the upper bound and lower bound given by (4.2) and (4.3). Hence, the malicious data attack is effective by adding the same noise under the MIE defense.

4.10.7 Proof of Lemma 4.2

From Hoeffding's inequality and (4.18), we have

$$\begin{aligned}\hat{p}_i(t) - \tilde{p}_i(t) &\leq E(\hat{p}_i(t) - \tilde{p}_i(t)) + \epsilon_1 \\ &\leq p_i - E(\tilde{p}_i(t)) + \epsilon_1 \\ &\leq \frac{2}{N+M} p_i - \frac{\sum_{j \in \mathcal{N} \cup \mathcal{M}} p_j}{(N+M)^2} + \epsilon_1\end{aligned} \tag{4.25}$$

with a probability greater than $1 - \exp\left(-\frac{2|K(t)|\epsilon_1^2}{\Delta X^2}\right)$.

We also have the following two inequalities:

1) with a probability greater than $1 - 2\exp(-\frac{2|K(t)|\epsilon_0^2}{\Delta D^2})$,

$$\left|\frac{\sum_{k \in K(t)} (\frac{\sum_{t \in N_k(t)} \frac{M}{M+N} \Delta d(t)}{|N_k(t)|})^2}{|K(t)|} - \frac{a}{|N_k(t)|}\right| \leq \epsilon_0; \tag{4.26}$$

2) with a probability greater than $1 - \exp(-\frac{2|K(t)|\epsilon_2^2}{\Delta X^2 \Delta D^2})$,

$$\begin{aligned}&\frac{\sum_{t \in K(t)} \frac{\sum_{t \in N_k(t)} \Delta d(t)}{|N_k(t)|} (D_i(t) - X'(t))}{|K(t)|} \\ &\leq \frac{\sum\limits_{t \in K(t)} E(\frac{\sum_{t \in N_k(t)} \Delta d(t)}{|N_k(t)|} (D_i(t) - X'(t)))}{|K(t)|} + \epsilon_2 \leq \epsilon_2.\end{aligned} \tag{4.27}$$

Next we bound the difference of the estimated quality before and after the attack. From (4.26), (4.27), and the definition of workers' estimated quality, we have

$$\begin{aligned}\tilde{p}_i - \tilde{p}_i' &= \frac{\sum\limits_{k \in K(t)} (D_i(\hat{t}) - X_k'(t))^2 - (D_i(\hat{t}) - X_a(t))}{|K(t)|} \\ &= \frac{\sum\limits_{k \in K(t)} (2D_i(\hat{t}) - X_k'(t) - X_a(t))(X_a(t) - X_k'(t))}{|K(t)|} \\ &= \frac{\sum_{k \in K(t)} 2(\frac{\sum_{t \in N_k(t)} \frac{M}{M+N} \Delta d(t)}{|N_k(t)|})(D_i(t) - X'(t))}{|K(t)|} - \frac{\sum_{k \in K(t)} (\frac{\sum_{t \in N_k(t)} \frac{M}{M+N} \Delta d(t)}{|N_k(t)|})^2}{|K(t)|} \\ &\leq (\frac{M}{M+N})^2 (\epsilon_0 - \frac{a}{|N_k(t)|}) + 2\epsilon_2 \frac{M}{N+M}\end{aligned} \tag{4.28}$$

with a probability greater than $1 - \exp\left(-\frac{2|K(t)|\epsilon_0^2}{\Delta D^2}\right) - \exp\left(-\frac{2|K(t)|\epsilon_2^2}{\Delta X^2 \Delta D^2}\right)$. From Hoeffding's inequality, we bound the difference of a worker's actual quality p_i and its estimated quality using the actual ground truth $\hat{p}_i$ without the attack:

$$\Pr(p_i - \hat{p}_i \leq \epsilon_3) \geq 1 - \exp(-\frac{2|K(t)|\epsilon_3^2}{\Delta p^2}). \tag{4.29}$$

Thus, from (4.25), (4.28), and (4.29), we can have that, for a normal worker, (4.22) is true with a probability greater than ω_1.

4.10.8 Proof of Lemma 4.3

First, similar as (4.25), we have

$$\tilde{p}^* - \hat{p}^* \leq -\frac{2M}{M+N}p^* + \frac{\sum_{j \in \mathcal{N} \cup \mathcal{M}} p_j}{(N+M)^2} + \epsilon_5 \tag{4.30}$$

with a probability greater than $1 - \exp\left(-\frac{2|K(t)|\epsilon_5^2}{\Delta X^2}\right)$.

Next we bound the difference of the estimated quality before and after the attack similar as (4.28):

$$\begin{aligned}
&\tilde{p}^{*\prime} - \tilde{p}^* \\
&= \frac{\sum_{t \in K(t)}((D(\hat{t}) - X_a(t))^2 - (D'(\hat{t}) - X'_k(t))^2)}{|K(t)|} \\
&\leq (\frac{N}{M+N})^2(\frac{a}{|N_k(t)|} + \epsilon_0) + 2\epsilon_2 \frac{N}{N+M}
\end{aligned} \tag{4.31}$$

with a probability greater than $1 - \exp\left(-\frac{2|K(t)|\epsilon_0^2}{\Delta D^2}\right) - \exp\left(-\frac{2|K(t)|\epsilon_2^2}{\Delta X^2 \Delta D^2}\right)$. We further bound $\hat{p}^*$ using Hoeffding's inequality as follows:

$$\Pr(\hat{p}^* - p^* \leq \epsilon_4) \geq 1 - \exp(-\frac{2|K(t)|\epsilon_4^2}{\Delta p^2}). \tag{4.32}$$

Thus, from (4.30), (4.31), and (4.32) we have, for a malicious worker, (4.23) is true with a probability greater than ω_2.

4.10.9 Proof of Lemma 4.1

First, for a normal worker, we have the following two inequalities: 1) from (4.26) and (4.27), we have

$$|\tilde{p}_i^{'} - \tilde{p}_i| \leq (a+\epsilon_0)(\frac{M}{N+M})^2 + 2\frac{M}{N+M}\epsilon_2 \triangleq \beta_1 \tag{4.33}$$

with a probability greater than $1-2\exp(-\frac{2|K(t)|\epsilon_0^2}{\Delta D^2})-2\exp(-\frac{2|K(t)|\epsilon_2^2}{\Delta X^2 \Delta D^2})$, note that this inequality also works for malicious workers; 2) from (4.25), we have

$$|\tilde{p}_i - \hat{p}_i| \leq \frac{2}{N+M}p_i + \frac{\sum_{i\in\mathcal{N}\cup\mathcal{M}} p_i}{(N+M)^2} + \epsilon_1 \triangleq \gamma_1 \tag{4.34}$$

with a probability greater than $1-2\exp(-\frac{2|K(t)|\epsilon_1^2}{\Delta X^2})$. Considering each term in (4.12), for a normal worker,

$$\begin{aligned}
&\Pr(|\tilde{p}_i^{'} - E(\tilde{p}_i^{'})| > \frac{\epsilon}{|S|}) \\
&\leq \underbrace{\Pr(|\tilde{p}_i^{'} - E(\tilde{p}_i^{'})| > \frac{\epsilon}{|S|} \mid |\tilde{p}_i^{'} - \tilde{p}_i| \leq \beta_1, |\tilde{p}_i - \hat{p}_i| \leq \gamma_1)}_{\textbf{Term } 1} \\
&\quad \Pr(|\tilde{p}_i^{'} - \tilde{p}_i| \leq \beta_1, |\tilde{p}_i - \hat{p}_i| \leq \gamma_1) \\
&+ \Pr(|\tilde{p}_i^{'} - E(\tilde{p}_i^{'})| > \frac{\epsilon}{|S|} \mid |\tilde{p}_i^{'} - \tilde{p}_i| > \beta_1, |\tilde{p}_i - \hat{p}_i| \leq \gamma_1) \\
&\quad \underbrace{\Pr(|\tilde{p}_i^{'} - \tilde{p}_i| > \beta_1, |\tilde{p}_i - \hat{p}_i| \leq \gamma_1)}_{\textbf{Term } 2} \\
&+ \Pr(|\tilde{p}_i^{'} - E(\tilde{p}_i^{'})| > \frac{\epsilon}{|S|} \mid |\tilde{p}_i - \hat{p}_i| > \gamma_1) \underbrace{\Pr(|\tilde{p}_i - \hat{p}_i| > \gamma_1)}_{\textbf{Term } 3}.
\end{aligned} \tag{4.35}$$

From (4.18), (4.33), and (4.34), we have that, for **Term 1**,

$$\begin{aligned}
\textbf{Term 1} &\leq \Pr(|\tilde{p}_i^{'} - \tilde{p}_i| + |\tilde{p}_i - \hat{p}_i| + |\hat{p}_i - p_i| + |p_i - E(\tilde{p}_i^{'})| > \frac{\epsilon}{|S|} \mid \\
&\quad |\tilde{p}_i^{'} - \tilde{p}_i| \leq \beta_1, |\tilde{p}_i - \hat{p}_i| \leq \gamma_1) \\
&\leq \Pr(|\hat{p}_i - p_i| > \frac{\epsilon}{|S|} - 2p_i - 2\frac{\sum_{i\in\mathcal{N}\cup\mathcal{M}} p_i}{(N+M)^2} - 2\frac{M}{N+M}\epsilon_2 - \\
&\quad (2a+\epsilon_0)(\frac{M}{N+M})^2 - \epsilon_1 \mid |\tilde{p}_i^{'} - \tilde{p}_i| \leq \beta_1, |\tilde{p}_i - \hat{p}_i| \leq \gamma_1).
\end{aligned} \tag{4.36}$$

Then from Hoeffding's inequality, we can have the bound

$$\begin{aligned}
&\textbf{Term 1}\ \Pr(|\tilde{p}_i^{'} - \tilde{p}_i| \le \beta_1, |\tilde{p}_i - \hat{p}_i| \le \gamma_1) \\
&\le \Pr(|\hat{p}_i - p_i| > \frac{\epsilon}{|S|} - 2p_i - \frac{2\sum_{i\in\mathcal{N}\cup\mathcal{M}} p_i}{(N+M)^2} - \frac{2M\epsilon_2}{N+M} - (2a+\epsilon_0)(\frac{M}{N+M})^2 - \epsilon_1) \\
&\le 2\exp(-2|K(t)|(\frac{\epsilon}{|S|} - 2\epsilon\Delta X)^2) = \frac{2}{t^2}, \qquad (4.37)
\end{aligned}$$

where

$$2(p_i + \frac{\sum\limits_{i\in\mathcal{N}\cup\mathcal{M}} p_i}{(N+M)^2} + \frac{M\epsilon_2}{N+M}) + \frac{(2a+\epsilon_0)M^2}{(N+M)^2} + \epsilon_1 \ge \epsilon_1 \ge 2\epsilon.$$

From (4.33), we have that, for **Term 2**,

$$\begin{aligned}
&\textbf{Term 2} \le \Pr(|\tilde{p}_i^{'} - \tilde{p}_i| > \beta_1) \\
&\le 1 - (1 - 2\exp(-\frac{2|K(t)|\epsilon_0^2}{\Delta D^2}) - 2\exp(-\frac{2|K(t)|\epsilon_2^2}{\Delta X^2 \Delta D^2})) \qquad (4.38) \\
&\le 2\exp(-\frac{2|K(t)|\epsilon_0^2}{\Delta D^2}) + 2\exp(-\frac{2|K(t)|\epsilon_2^2}{\Delta X^2\Delta D^2}) \le \frac{4}{t^2},
\end{aligned}$$

where $|\frac{\epsilon}{|S|} - 2\epsilon\Delta X| \le \min\{\epsilon_0, \epsilon_2\}$. From (4.34), we have that, for **Term3**,

$$\textbf{Term 3} \le 2\exp(-\frac{2|K(t)|\epsilon_1^2}{\Delta X^2}) \le \frac{2}{t^2}. \qquad (4.39)$$

Thus, from (4.37), (4.38), and (4.39), we have that, for a normal worker,

$$\Pr(|\tilde{p}_i^{'} - E(\tilde{p}_i^{'})| > \frac{\epsilon}{|S|}) \le \frac{8}{t^2}.$$

Next we study the bound of each term in (4.12) for a malicious worker. Similarly, we first give the following inequality: from (4.30), we have

$$|\tilde{p}^* - \hat{p}^*| \le \frac{2M}{N+M}p^* + \frac{\sum_{i\in\mathcal{N}\cup\mathcal{M}} p_i}{(N+M)^2} + \epsilon_5 \triangleq \gamma_2 \qquad (4.40)$$

with a probability greater than $1 - 2\exp(-\frac{2|K(t)|\epsilon_5^2}{\Delta X^2})$. For a malicious worker, each term in (4.12) can be bounded similarly as in (4.35). Thus, similar to (4.37), (4.38), and (4.39), we

bound the three terms. From (4.20), (4.33), and (4.40), we have

$$\textbf{Term 1}\ \Pr(|\tilde{p}^{*'} - \tilde{p}^{*}| \leq \beta_1, |\tilde{p}^{*} - \hat{p}^{*}| \leq \gamma_2) \leq \frac{2}{t^2}, \tag{4.41}$$

$$\textbf{Term 2} \leq \Pr(|\tilde{p}^{*'} - \tilde{p}^{*}| > \beta_1) \leq \frac{4}{t^2}, \tag{4.42}$$

$$\textbf{Term 3} \leq 2\exp\left(-\frac{2|K(t)|\epsilon_5^2}{\Delta X^2}\right) \leq \frac{2}{t^2}. \tag{4.43}$$

Thus, from (4.41), (4.42), and (4.43), we have

$$\Pr(|\tilde{p}^{*'} - E(\tilde{p}^{*'})| > \frac{\epsilon}{|S|}) \leq \frac{8}{t^2}. \tag{4.44}$$

Chapter 5

Quality-Aware Adaptive Computation and Device Selection for Cost-Effective Wireless Federated Learning.

5.1 Introduction

Federated learning (FL) [80] is an emerging and promising ML paradigm, which performs the training of ML models in a distributed manner. Instead of transmitting data from a potentially large number of devices to a central server in the cloud for training, FL allows the data to remain at devices (such as smartphone), and trains a global ML model on the server by collecting and aggregating model updates locally computed on each device based on her local data. One significant advantage of using FL is to preserve the privacy of individual devices' data. Moreover, since only local ML model updates instead of local data are sent to the server, the communication costs can be greatly reduced. Furthermore, FL can exploit substantial computation capabilities of ubiquitous smart devices, which are often under-utilized. In particular, when FL is used in a wireless edge network, data samples generated at individual wireless devices can be exploited via local computation and global aggregation based on distributed ML. As a result, *wireless federated learning* (WFL) can achieve collaborative intelligence in wireless edge networks. A general consensus is that WFL can support intelligent control and management of wireless communications and networks (such as in [7, 8, 9, 10, 11]), and can enable many AI applications based on wireless networked systems, including connected and autonomous vehicles, collaborative robots, multi-user virtual/mixed reality.

As is standard, learning accuracy is a key performance metric for FL. The accuracy of the trained ML model in FL depends heavily on which devices participate in the training process and the quality of their local model updates. Specifically, stochastic gradient descent (SGD) is a

popular method for ML training that is widely studied in the literature (e.g., in [12, 14]). When SGD is used for FL, the quality of a local model update in each iteration can be measured by the variance of the gradient, which depends on the *mini-batch size* used to compute the gradient. A key observation is that the quality of local updates (determined by the mini-batch size) can be treated as a *design parameter* and used as a control knob to be adapted across devices and over time. Such quality-aware computation can substantially improve the learning accuracy of WFL.

In this chapter, we study quality-aware computation for WFL, aiming to maximize the learning accuracy, while taking into account the costs and constraints of devices' computation and communication resources. In particular, we investigate how to adaptively select participating devices and determine their mini-batch sizes over the training process. To this end, several significant challenges need to be addressed: 1) The quality (determined by the mini-batch size) of local stochastic gradient updates can be heterogeneous across devices and varying over the training process, which has non-trivial impacts on the accuracy of the final learnt model and also devices' computation costs. 2) The unique features of wireless edge networks, including time-varying wireless channels and computation costs of devices, should be taken into account. To achieve a desired tradeoff between learning accuracy and communication and computation costs, participating devices of FL in each round and their local updates' quality should be determined based on their impacts on the eventual training loss, as well as devices' channel conditions and computation costs.

The main contributions of this chapter are summarized as follows:

- We propose quality-aware computation for FL in wireless edge networks, which controls the *quality* of devices' local model updates via the mini-batch sizes. Our goal is to improve the learning accuracy of FL while taking into account the computation and communication costs of participating devices.

- We characterize performance bounds on the training loss as a function of devices' local updates' quality (quantified by the variances of local stochastic gradients) over the training process, for both cases when devices have IID data and have non-IID data. Our

findings rigorously show that a *more recent* local update has a larger impact on the training loss, which implies that it is beneficial to use larger mini-batch sizes in a later round (given the total mini-batch size of devices over all rounds).

- With the obtained insights, we develop channel-aware adaptive algorithms that select participating devices and determine their mini-batch sizes for each round of the FL algorithm, based on their impacts on the training loss as well as their wireless channel conditions and computation costs. We characterize the optimal device selection (for the case of homogeneous computation capabilities) and the optimal mini-batch sizes. Our results show that it is more beneficial to select more participating devices (for the case of IID data) and use larger mini-batch sizes in a *later* round. For the case of IID data and single local iteration per round, we show that the proposed greedy device selection algorithm can achieve a guaranteed approximation ratio, by exploiting the non-monotone submodular property of the problem.

- We evaluate the proposed quality-aware adaptive algorithms using both simulations and testbed-based experiments for the popular MNIST-based hand-written digit recognition. The results demonstrate that these algorithms outperform existing methods in terms of learning accuracy and cost-effectiveness.

The remainder of this chapter is organized as follows. Section 5.2 reviews related work. In Section 5.3, we present a framework of quality-aware computation for WFL. Under this framework, we characterize performance bounds for the training loss in Section 5.4. Based on the training loss bounds, in Section 5.5, we investigate adaptive device selection and mini-batch size design. Simulation results are provided in Section 5.6. Conclusion is discussed in Section 5.7.

5.2 Related Work

Distributed Machine Learning. With rapid advances in ML and AI technologies, distributed ML has also received substantial research activities in the past decade [81, 82, 83, 84, 14, 15]. In many of the existing studies, participating devices are owned and operated by a single

organization (e.g., used by Google or Facebook on computer clusters in their data centers), and are powerful computers interconnected by high-speed networks (often wired networks). There have recently been some works on algorithm design for distributed ML while taking into account computation and/or communication costs (in terms of delay, energy consumption, bandwidth use, etc) of carrying out distributed learning. In particular, communication-efficient distributed learning has garnered a lot of attention [85, 86, 87, 88, 89, 90, 91, 92]. Many of these works used data compression to reduce the size of local models and thus the communication workloads [86, 89, 91, 93]. Some other works [84, 94] studied the optimal communication frequency of local model updates. However, most of the studies above do not consider the setting of *wireless edge networks*.

Wireless Federated Learning. FL has emerged as a disruptive computing paradigm for ML by democratizing the learning process to potentially many individual devices. For WFL, wireless edge networks have salient features, including heterogeneous and time-varying computing and communication resources that need to be accounted for. Recent studies on FL made effort to take into account these issues [95, 96, 97, 98, 99, 100, 101, 102]. For example, Tran et al [95] studied FL in wireless networks for devices with heterogeneous computation and communication capabilities. In [103], Tu et al studied computation offloading based distributed learning where devices have diverse computing and communication resources. Xu et al [98] used numerical experiments to show that devices participating in a later round in the training process of FL would have more effect on learning accuracy than in an earlier round. Besides, [104] showed that geometrically increasing mini-batch size in the training process can achieve better convergence performance in centralized ML. However, all these existing works do not explore mini-batch size to *control the quality* of local model updates in the context of WFL. Note that the analysis of mini-batch sizes' impacts on the training loss for FL is significantly different from the case of centralized ML (e.g., in [104]), due to devices' non-IID data, heterogeneous and time-varying mini-batch sizes, and multiple local iterations in the FL setting. In this paper, we characterize training loss bounds as a function of mini-batch sizes for the general

case of *multiple* local iterations per round for both IID and non-IID cases. Moreover, we devise adaptive device selection and mini-batch design algorithms that can achieve a performance guarantee compared to the optimal solution.

5.3 Quality-Aware Computation for Wireless Federated Learning

In this section, we present the system model of quality-aware computation for FL in wireless edge networks.

Consider the setting where the distributed learning process of FL is carried out by a set of wireless devices. The server of FL can reside in the cloud or at the edge (e.g., access point or base station of a wireless network), and the devices are connected to the FL server via wireless links. A device incurs a computation cost (measured by the computation time, energy consumption, etc) for computing a local model update, which depends on the computation capability of the device and the mini-batch size used to compute the update. Let $c_{p,t}^i$ be device i's cost of computing her local update using one data sample in round t. Besides the computation cost, a device also incurs communication cost for communicating local updates to the server (measured by the communication time, energy consumption, etc), which depends on the device's wireless channel condition. Let $c_{m,t}^i$ be device i's communication cost in round t.

Consider the following FL problem:

$$\min_{\mathbf{w}} \ F(\mathbf{w}) \triangleq \sum_{i=1}^{N} \frac{D_i}{D} F_i(\mathbf{w}),$$

where $F_i(\mathbf{w})$ is defined by

$$F_i(\mathbf{w}) \triangleq \frac{1}{D_i} \sum_{m=1}^{D_i} f_i(\mathbf{w}; \xi_m^i),$$

$f_i(\cdot)$ is the per-sample loss function of device i, N is the number of devices, $\mathcal{D}_i = \{\xi_1^i, \xi_2^i, \ldots, \xi_{D_i}^i\}$ is device i's local dataset for updating the model parameter, and $D \triangleq \sum_{i=1}^{N} D_i$.

In each round of FL, K out of N devices are selected from the device set $\mathcal{N}$ to compute local updates, communicate their local updates to the server, and receive the updated global model from the server.

In round t,[1] each selected device i receives the global model $\mathbf{w}_{t-1}$ from the server, sets $\mathbf{w}_{t,0}^i = \mathbf{w}_{t-1}$, and then performs H local iterations of SGD. In the hth local iteration, device i computes the average gradient $g_{t,h-1}^i$ of the loss function using a mini-batch of D_t^i data samples randomly drawn from her local dataset $\mathcal{D}_i$. Then device i updates her local model as

$$\mathbf{w}_{t,h}^i = \mathbf{w}_{t,h-1}^i - \eta g_{t,h-1}^i,$$

where

$$g_{t,h-1}^i \triangleq \frac{1}{D_t^i} \sum_{j=1}^{D_t^i} \nabla f(\mathbf{w}, \xi_{t,h}^{i,j}),$$

η is the step size, and $\xi_{t,h}^{i,j}$ is the jth data sample randomly drawn from device i's local dataset $\mathcal{D}_i$. After H local iterations, device i sends her local update $\mathbf{w}_{t,H}^i$ for round t to the server. For ease of exposition, we assume that devices' number of local iterations H does not change over rounds, and the mini-batch size D_t^i for a local iteration of device i in round t does not change over the H local iterations in round t (but can change over rounds). Our results in this chapter can be extended to the case where these assumptions are relaxed.

At the end of round t, the server aggregates K devices' local models and updates the global model as

$$\mathbf{w}_t = \sum_{i=1}^{K} \frac{D_t^i}{D_t} \mathbf{w}_{t,H}^i,$$

where $D_t \triangleq \sum_{i=1}^K D_t^i$.

Due to the randomness of data sampling for computing the update in SGD, the computed gradient of a device $g_{t,h}^i$ deviates from the expected gradient $E[g_{t,h}^i]$, and thus slows down the convergence of the global model. The *quality* of a device's local update is captured by the variance of the update, given by

$$q_i \triangleq E\left[\left\|g_{t,h}^i - \bar{g}_{t,h}^i\right\|^2\right],$$

[1] In this chapter, we use t as the index of communication rounds and h as the index of local iterations. The subscript $\{t, h\}$ denotes the hth local iteration in round t.

where $\bar{g}_{t,h}^i \triangleq E[g_{t,h}^i]$. Assume that the loss function f satisfies $E\left\|\nabla f_i\left(\mathbf{w}_t, \xi_m^i\right) - \nabla F_i\left(\mathbf{w}_t\right)\right\|^2 \leq \sigma^2$, $\forall t$. It can be shown that [105]

$$E\left[\left\|g_{t,h}^i - \bar{g}_{t,h}^i\right\|^2\right] \leq \frac{\sigma^2}{D_t^i}.$$

Note that a local update's quality is determined by the *mini-batch size* D_t^i used to compute the local update, such that a local update computed with larger mini-batch sizes has higher quality. In this chapter, we focus on the SGD method where a mini-batch of data samples are randomly drawn *with replacement* from a device's local dataset, for the sake of technical tractability. Such SGD based on sampling with replacement has been widely studied as a popular method in the literature (e.g., in [14, 105, 13, 15]), as it allows for tractable theoretical analysis and thus provides performance guarantees for ML algorithms. On the other hand, this method is different from the version of SGD that is often used in practice, where a mini-batch of data samples are randomly drawn *without replacement* from the local dataset (a.k.a. the epoch-based SGD). This without-replacement based SGD method has been much less studied [106, 107]), although it can achieve better empirical performance than with replacement.

5.4 Training Loss Bound

In this section, under the framework presented in the previous section, we study the learning accuracy of FL, measured in terms of the training loss. We will first characterize performance bounds on the training loss as a function of devices' mini-batch sizes. Based on this result, we then discuss the impacts of mini-batch sizes and other system parameters on the training loss.

5.4.1 The Case of IID Data

We first analyze the training loss for the case when devices have IID local data.

Before we discuss the training loss bound, we define a virtual sequence $\bar{\mathbf{w}}_{t,h}$ that is given by

$$\bar{\mathbf{w}}_{t,h} = \sum_{i \in S_t} \frac{D_t^i}{D_t} \mathbf{w}_{t,h}^i, \forall t, h, \tag{5.1}$$

where S_t is the set of devices selected to participate in round t, and $D_t = \sum_{i \in S_t} D_t^i$ is the total mini-batch size of the selected devices in one local iteration. Note that $\bar{\mathbf{w}}_{t,h}$ is not accessible when the participating devices have not completed H local iterations (i.e., $h < H$), and $\mathbf{w}_t = \bar{\mathbf{w}}_{t,H}$.

Theorem 5.1 Suppose F is L-smooth and μ-strongly convex, and $\|\nabla F_i(\mathbf{w}_t)\|^2 \leq B^2$ and $E\|\nabla f_i(\mathbf{w}_t, \xi_m^i) - \nabla F_i(\mathbf{w}_t)\|^2 \leq \sigma^2$, $\forall i, t$. Suppose that the step size $\eta \leq \frac{1}{2L}$. Then the training loss for the case of IID data is bounded above by:

$$E[F(\mathbf{w}_T) - F(\mathbf{w}^*)] \leq \frac{L}{2}(1-\mu\eta)^{TH}\|\mathbf{w}_0 - \mathbf{w}^*\|^2 + \frac{L}{2}\sum_{t=1}^{T}\sum_{h=1}^{H}\left((1-\mu\eta)^{TH-(t-1)H-h}\left(\frac{\eta^2\sigma^2}{D_t} + 4L\eta^3(H-1)^2\left(\frac{\sigma^2|S_t|}{D_t} + B^2\right)\right)\right). \tag{5.2}$$

Note that the assumption of bounded gradient $E\|\nabla F_i(\mathbf{w}_t)\|^2 \leq B^2$ is common in convergence analyses for FL [14, 15]. The detailed proofs for theorems are given in the Appendix.

Remark 5.1 Theorem 5.1 shows that the training loss bound consists of two terms. The first term decreases geometrically with the total number of local iterations TH, and is due to that SGD in expectation makes progress towards the optimal solution. The second term of the bound is caused by the randomness of data sampling in SGD for computing local updates, which depends on the total mini-batch size D_t of participating devices in each round t. Observe that a larger total mini-batch size D_t reduces the training loss. Also note that as there are multiple local iterations and only one update aggregation in a round, the randomness of data sampling accumulates over the local iterations. This implies that reducing the local iteration number H (which is equivalent to increasing the communication frequency) can improve the training loss.

Remark 5.2 In the second term in (5.2), $\frac{\sigma^2}{D_t}$ is the upper bound of the variance of the (virtual) global model update $\bar{\mathbf{w}}_{t,h}$, which is determined by the variances of participating devices' local model updates. Thus the coefficient $(1-\mu\eta)^{TH-(t-1)H-h}\eta^2$ of the variance bound $\frac{\sigma^2}{D_t}$ of the global update quantifies the impact of mini-batch sizes on the training loss. Note that this

coefficient increases with the round index t as $1 - \mu\eta < 1$. Therefore, the mini-batch sizes in a later round have a larger impact on the training loss (the second term in (5.2)) than in an earlier round. This observation has an important implication: Given a total number of mini-batch sizes of all participating devices over all rounds $\sum_{t=1}^{T} D_t$, an increasing total mini-batch size D_t over rounds results in a lower training loss than a constant or decreasing total mini-batch size.

Remark 5.3 Compared to the case where devices perform a single local iteration in each round [108], the result given in Theorem 5.1 includes an additional term which is the accumulative error due to multiple local iterations. First note that when devices perform a single local iteration per round (i.e., $H = 1$), the training loss bound given in Theorem 5.1 is the same as in Theorem 1 in [108]. When devices perform multiple local iterations per round, the additional error compared to the previous case (the last term in (5.2)) increases with the number of rounds T. This is because, in the single iteration case, a device i updates its local model in each round based on the global model obtained from aggregating all participating devices' local models in the previous round (whose variance is determined by the total mini-batch size D_t). However, when a device performs H number of local iterations, there are $H - 1$ times when it updates its local model based on its local model only (whose variance is determined by the mini-batch size D_t^i which is less than D_t). Thus, with less communications, FL with multiple local iterations suffers a larger training loss due to more randomness of local updates.

5.4.2 The Case of Non-IID Data

In many practical situations, devices' local data is non-IID due to various reasons (e.g., location, device, user behavior, etc). Note that in the case when devices have non-IID data, we have $E[F_i(\mathbf{w}^*)] \neq E[F_i(\mathbf{w}_i^*)]$, $\forall i \in \mathcal{N}$, where $\mathbf{w}^*$ and $\mathbf{w}_i^*$ are the models that minimize the loss function when using all devices' data and only device i's data, respectively. Furthermore, it has been proved that when devices perform different total numbers of local iterations, the final global model is inconsistent with the solution to the FL problem, i.e., the aggregated model is biased. One way to solve this problem is selecting devices randomly and re-weight devices according to their participating probabilities. In this subsection, we study the training loss

bounds for the non-IID data case when each device participates with a probability a_t^i $(0 < a_t^i \leq 1)$ in each round t.

First, we give the adjusted aggregation weight that ensures unbiased model aggregation.

Lemma 5.1 When each device participates with a probability a_t^i, the aggregated model is unbiased, i.e., $E_{S_t}[\mathbf{w}_t] = \tilde{\mathbf{w}}_t$, if the server updates the global model as

$$\mathbf{w}_t = \mathbf{w}_{t-1} + \sum_{i \in S_t} \frac{p_i}{a_t^i} (\mathbf{w}_{t,H}^i - \mathbf{w}_{t-1}), \tag{5.3}$$

where p_i, $\forall i \in \mathcal{N}$, is the original aggregation weight, and $\tilde{\mathbf{w}}_t$ is the global model with full device participation in round t.

The next result presents training loss bounds for the non-IID data case.

Theorem 5.2 Suppose F is L-smooth and μ-strongly convex, and $E \left\| \nabla f_i \left(\mathbf{w}_t, \xi_m^i \right) - \nabla F_i \left(\mathbf{w}_t \right) \right\|^2 \leq \sigma_i^2$, and $E \left\| \nabla f_i \left(\mathbf{w}_t, \xi_m^i \right) \right\|^2 \leq C^2$, $\forall i, t$. Suppose that the step size $\eta \leq \frac{1}{2L}$. Then the training loss for the case of non-IID data is bounded above by

$$\begin{aligned} & E[F(\mathbf{w}_T) - F(\mathbf{w}^*)] \\ \leq & \frac{L}{2} (1 - \mu\eta)^{TH} E \left\| \mathbf{w}_0 - \mathbf{w}^* \right\|^2 + \frac{L\eta^2}{2} \sum_{t=1}^{T} \sum_{h=1}^{H} \Big[(1 - \mu\eta)^{TH-(t-1)H-h} \\ & \sum_{i \in \mathcal{N}} ({p_i}^2 \frac{\sigma_i^2}{D_t^i} + p_i (6Ld_i + 2(H-1)^2 C^2) + p_i^2 \frac{1 - a_t^i}{a_t^i} C^2 \Big], \end{aligned} \tag{5.4}$$

where D_t^i is device i's mini-batch size if she is selected to participate in round t, and $d_i \triangleq E[F_i(\mathbf{w}^*)] - E[F_i(\mathbf{w}_i^*)]$ quantifies the heterogeneity degree of the data held by device i [15].

Remark 5.4 Theorem 5.2 shows that, similar to the case of IID data, the training loss for the case of non-IID data is also affected by mini-batch sizes (as shown in the second term in (5.4)), and the training loss reduces when the mini-batch sizes increase (due to that the variances of local stochastic gradients decrease). Also note that as the coefficient $(1 - \mu\eta)^{TH-(t-1)H-h}$ of a local model's variance $\frac{\sigma_i^2}{D_t^i}$ increases with the round index t, the mini-batch size in a later round has a larger impact on the training loss than in an earlier round, similar to the observation in the IID data case.

Compared to the IID data case, one key difference of the result here is that it depends not merely on the total mini-batch size of devices in a round, but on each device's individual mini-batch size in the round, as well as devices' local datasets' weights $\{p_i\}$, participation probability $\{a_t^i\}$, and the degrees of heterogeneity $\{d_i\}$. We observe that, to ensure the convergence of unbiased FL model, each device should have a participation probability larger than 0. Otherwise, the training loss bound tends to infinity even with large number of rounds T. Furthermore, similar to the observation on mini-batch sizes, as the coefficient $(1-\mu\eta)^{TH-(t-1)H-h}$ increases with the round index t, the participation probabilities in a later round has a larger impact on the training loss. This observation has an important implication: Given a devices expected total participating rounds $\sum_{t=1}^{T} a_t^i$, $\forall i \in \mathcal{N}$, an increasing participation probability a_t^i over rounds results in a lower training loss than a constant or decreasing participation probability.

From the definition of d_i, we can see that for the case of IID data, the degree of heterogeneity of each device's data is 0. In the case of non-IID data, as devices' data are not drawn from the same distribution, we have $E[F_i(\mathbf{w}^*)] \neq E[F_i(\mathbf{w}_i^*)]$, $\forall i \in \mathcal{N}$, and $d_i \neq 0$. Due to the non-zero d_i, the second term in (5.4) does not converge to 0 as the mini-batch sizes increase. Moreover, as the coefficients of a local model's variance $\frac{\sigma_i^2}{D_t^i}$ is p_i, the mini-batch size of a device with a higher weight p_i has a larger impact on the training loss than with a lower weight p_i.

5.5 Cost-Effective Channel-Aware Adaptive device Selection and Mini-Batch Size Design

In the previous section, we study the impacts of devices' mini-batch sizes used to compute local model updates on the training loss in various settings, by characterizing training loss bounds as a function of mini-batch sizes. In this section, we study how to select participating devices and their mini-batch sizes in each round to minimize the training loss bound, while taking into account devices' communication and computation costs. Note that we consider continuous-valued mini-batch sizes D_t^i in our theoretical analysis, which can be converted back to the nearest integer values when used in practice.

We aim to minimize the system cost which consists of the FL training loss and devices' total communication and computation cost. The optimization problem can be formulated as

$$\begin{aligned} &\min_{\mathcal{S},D} \gamma E[F(\mathbf{w}_T) - F(\mathbf{w}^*)] + (1-\gamma) \sum_{t=1}^{T} \sum_{i \in S_t} (H c_{p,t}^i D_t^i + c_{m,t}^i), \\ &s.t.\ D_t^i \leq D_B^i, \forall i, t, \end{aligned} \tag{5.5}$$

where $\mathcal{S} = \{S_1, \cdots, S_T\}$ consists of the sets of participating devices in all rounds, $D = \{D_t^i | \forall i \in S_t, \forall t\}$ is the set of mini-batch sizes of participating devices in all rounds, $c_{p,t}^i$ is device i's unit computation cost in round t, $c_{m,t}^i$ is the communication cost of device i in round t, D_t^i is the mini-batch size for one local iteration that is assigned to device i in round t[2], $\gamma \in (0, 1]$ is the weight that balances the training loss and the cost, and D_B^i is the maximum mini-batch size that device i can compute in one local iteration. Note that problem (5.5) involves multi-objective optimization of learning accuracy and learning cost: By controlling the weight γ, any Pareto-optimal solution of these two objectives can be reached by solving problem (5.5). A variant formulation of problem (5.5) is a constrained optimization problem, where the objective function is the training loss bound while devices' total communication and computation cost is subject to a constraint (or vice versa). The solution of this variant problem can be derived from that of problem (5.5).

5.5.1 The Case of IID Data

We first focus on problem (5.5) for the case of IID data, aiming to minimize the training loss upper bound given by (5.2), plus the communication and computation costs in (5.5). Since the first term of the bound is not related to devices' mini-batch sizes D_t^i, it suffices to find the mini-batch sizes that minimize the second term. Thus, we can rewrite (5.5) as

[2]Recall that we assume that the mini-batch size D_t^i for a local iteration of device i in round t does not change over the H local iterations, but can change over communication rounds.

$$
\begin{aligned}
&\min_{\mathcal{S},D} \sum_{t=1}^{T} \mathcal{J}_1(S_t, D_t) \\
&\quad = \sum_{t=1}^{T} \left(\sum_{h=1}^{H} \left(\frac{\gamma L \eta^2 \sigma^2}{2} (1-\mu\eta)^{TH-(t-1)H-h} \left(\frac{1}{D_{t,h}} + 4L\eta(H-1)^2 \left(\frac{|S_t|}{D_{t,h}} + \frac{B^2}{\sigma^2} \right) \right) \right) \right. \\
&\qquad \left. + (1-\gamma) \sum_{i \in S_t} (H c_{p,t}^i D_t^i + c_{m,t}^i) \right), \\
&s.t.\ D_t^i \leq D_B^i, \forall i, t.
\end{aligned}
\tag{5.6}
$$

We can see that the problem can be decomposed into the minimization problems in each round. Thus, we aim to find the optimal S_t and D_t for a single round t. Moreover, since each minimization problem in a round is a mixed integer programming problem, we decompose it using the concept of generalized Benders decomposition. The problem is decomposed into two subproblems, namely mini-batch size design and device selection. First, given the selected device set in round t, the mini-batch size design problem is solved. Further, based on the optimal mini-batch size design, we solve the device selection problem.

The Case of Single Local Iteration ($H = 1$).

To obtain some useful insights, we first study the case where devices perform a single local iteration in each round. In this case, the error caused by multiple local iterations does not exist. The results for this setting will serve as the basis for the case of multiple local iterations.

Optimal Mini-Batch Size. Given the selected device set, the optimal mini-batch sizes can be obtained as follows. Given the selected device set, the total communication cost is the sum of the communication costs of selected devices, and can be seen as a constant. With the same mini-batch size, a device with a lower computation cost contributes more than with a higher computation cost. Hence, the optimal mini-batch sizes are determined in the ascending order of devices' computation costs until the minimum value of $\mathcal{J}_1$ is reached. It can be shown that $\mathcal{J}_1$ is a convex function of D_t^i when other devices' mini-batch sizes are given. The following result characterizes devices' optimal mini-batch sizes.

Proposition 5.1 Let devices in S_t be ordered as $c_{p,t}^1 \leq c_{p,t}^2 \leq \cdots \leq c_{p,t}^{|S_t|}$. Then the optimal mini-batch sizes of devices are determined iteratively in this order, where the ith device's

optimal mini-batch size is given by

$$D_t^{i*}(S_t) = \min\{D_B^i, \max\{\eta\sigma\sqrt{\frac{L\gamma\omega_t}{2(1-\gamma)c_{p,t}^i}} - \sum_{j=1}^{i-1} D_t^{j*}, 0\}\}, \tag{5.7}$$

where $\omega_t = (1-\mu\eta)^{T-t}$.

Note that the optimal mini-batch size is a function of the Lipschitz constant L of the loss function, which can be estimated given the specific loss function. Proposition 5.1 shows that a device's optimal mini-batch size is larger in a later round. This is because the coefficient ω_t in the training loss bound increases with the round index t. As a result, the optimal total mini-batch size in a round also increases with the round index.

Device Selection. With the optimal mini-batch size design, the device selection problem in round t can be rewritten as

$$\min_{S_t} \ \mathcal{J}_1(S_t, D_t^*(S_t)) = \gamma\left(\frac{L\omega_t\eta^2\sigma^2}{2\sum_{i\in S_t} D_t^{i*}(S_t)}\right) + (1-\gamma)\sum_{i\in S_t}(c_{p,t}^i D_t^{i*}(S_t) + c_{m,t}^i). \tag{5.8}$$

To obtain some useful insights, we first study the optimal device selection when devices have homogeneous computation capabilities (i.e., the same computation unit cost c_p and maximum mini-batch size D_B). In this case, it follows from Proposition 5.1 that the optimal mini-batch sizes are given by $D_t^{i*}(S_t) = \min\{D_B, \max\{\eta\sigma\sqrt{\frac{L\gamma\omega_t}{2(1-\gamma)c_p}} - (i-1)D_B, 0\}\}, \forall i$. We can see that device selection does not affect the training loss and the total computation cost but only the total communication cost. Thus, the optimal device selection in round t is given as follows.

Proposition 5.2 For the case of homogeneous computation capabilities, the optimal device selection algorithm selects devices in the increasing order of their communication costs (i.e., $c_{m,t}^1 \leq c_{m,t}^2 \leq \cdots c_{m,t}^N$), with each selected device's mini-batch size set to be as much as possible (up to the maximum mini-batch size D_B), until the total mini-batch size of selected devices reaches the optimal level $\eta\sigma\sqrt{\frac{L\gamma\omega_t}{2(1-\gamma)c_p}}$.

Based on the result above, we can see that the optimal set of selected devices for problem (5.8) increases with the round number t. Therefore, in a later round of the FL algorithm, it is optimal to not only use larger mini-batch sizes, but also to select more devices.

Next we consider the general case where devices have heterogeneous computation unit costs and maximum mini-batch sizes. First note that problem (5.8) can be cast as the maximization problem of $-\mathcal{J}_1(S_t, D_t^*(S_t))$. Then we have the following property of the problem, of which the proof is given in the appendix.

Lemma 5.2 $-\mathcal{J}_1(S_t, D_t^*(S_t))$ is a negative non-monotone submodular function.

Note that it is difficult to solve a negative non-monotone submodular maximization problem with performance guarantee (e.g., with an approximation ratio). To overcome this challenge, we transform the above problem into a non-negative non-monotone submodular maximization problem.

First, we find the maximum value of $\mathcal{J}_1(S_t, D_t^*(S_t))$. From the optimal mini-batch size design, we can see that there exists some device j, such that for any device $j' \in S_t$ that has $c_{p,t}^{j'} \geq c_{p,t}^{j}$, we have $D_t^{j'*}(S_t) = 0$. The objective function $\mathcal{J}_1$ decreases as devices' optimal mini-batch sizes are determined iteratively according to Proposition 5.1 until device j, and does not change when the optimal mini-batch sizes of devices after j are determined. Therefore, given a selected device set S_t, $\mathcal{J}_1$ is maximized when only one device's mini-batch size is non-zero. Thus, for any $S \subseteq \mathcal{N}$, the maximum value of $\mathcal{J}_1$ is given by

$$\mathcal{J}_{1,\max} = \frac{\gamma L \eta^2 \sigma^2}{2} + (1-\gamma)(c_{p,\max} D_{B,\max} + N c_{m,\max}),$$

where $c_{p,\max} = \max\{c_{p,t}^i | \forall i, t\}$, $D_{B,\max} = \max\{D_B^i | \forall i\}$, and $c_{m,\max} = \max\{c_{m,t}^i | \forall i, t\}$.

Based on the maximum value of $\mathcal{J}_1(S_t, D_t^*(S_t))$, we define a new function $\mathcal{G}(S_t)$ and rewrite the device selection problem in round t as

$$\max_{S_t} \mathcal{G}(S_t) \triangleq \begin{cases} -\mathcal{J}_1(S_t, D_t^*(S_t)) + \mathcal{J}_{1,\max}, & \text{if } S_t \neq \emptyset \\ 0, & \text{if } S_t = \emptyset. \end{cases}$$

Note that in each round t of the FL algorithm, at least one device is selected (i.e., $|S_t| > 0$) with a positive mini-batch size (i.e., $D_t > 0$). This is because when $S_t = \emptyset$, the global model is not updated so that round t should not be counted as a round of the FL algorithm. Thus, we can define that $\mathcal{G}(\emptyset) \triangleq 0$. We can see that the above two problems are equivalent since $\mathcal{J}_{1,\max}$

Algorithm 7: Device selection for the case of IID data and single local iteration per round

$X_0 \leftarrow \emptyset, Y_0 \leftarrow \mathcal{N}$;
for $i = 1, 2, \cdots, N$ **do**
  $a_i \leftarrow \mathcal{G}\left(X_{i-1} \cup \{i\}\right) - \mathcal{G}\left(X_{i-1}\right)$;
  $b_i \leftarrow \mathcal{G}\left(Y_{i-1} \setminus \{i\}\right) - \mathcal{G}\left(Y_{i-1}\right)$;
  if $a_i \geq b_i$ **then**
    $X_i \leftarrow X_{i-1} \cup \{i\}, Y_i \leftarrow Y_{i-1}$;
    else
      $X_i \leftarrow X_{i-1}, Y_i \leftarrow Y_{i-1} \setminus \{i\}$;
$S_t \leftarrow X_N$;
return S_t.

is a constant. Next, we focus on finding the solution that maximizes $\mathcal{G}(S_t)$. From Lemma 5.2, it follows directly that $\mathcal{G}(S_t)$ is a non-negative non-monotone submodular function.

Next, we can apply the DeterministicUSM Algorithm [109] to solve the device selection problem, which is given in Algorithm 7. The approximation ratio of DeterministicUSM is given by the following lemma.

Lemma 5.3 [109] Algorithm DeterministicUSM is a $\frac{1}{3}$-approximation algorithm for maximizing function $\mathcal{G}(S_t)$.

Given the result above, we then show the approximation ratio of Algorithm 7 for our problem, under the optimal mini-batch size design given in Proposition 5.1. The proof of the following result is given in the appendix.

Theorem 5.3 Under the optimal mini-batch size design, with the device selection given by Algorithm 1, the system loss is bounded above by

$$\sum_{t=1}^{T} \mathcal{J}_1(S_t, D_t^*(S(t))) \leq \frac{1}{3}\mathcal{OPT} + \mathcal{O}(T), \tag{5.9}$$

where $\mathcal{OPT}$ is the system loss of the optimal device and mini-batch size selection.

The Case of Multiple Local Iteration ($H > 1$).

Following the results for the case of single local iteration per round, we first study devices' optimal mini-batch size in each round. It can be shown that $\mathcal{J}_1$ is a convex function of D_t^i when

Algorithm 8: Device selection for the case of IID data and multiple local iterations per round

$u \leftarrow \arg_i \min \mathcal{J}_1(\{i\}, D_t^*(\{i\}))$;
$k \leftarrow 1$,$X_0 \leftarrow \{u\}$,$Y_0 \leftarrow \mathcal{N}$;
for $k < N$ **do**
  $i' \leftarrow \arg_i \min M_1(X_{k-1}, i)$;
  if $M_1(X_{k-1}, i') < 0$ **then**
    $X_k \leftarrow X_{k-1} \cup \{i'\}$,$Y_k \leftarrow Y_{k-1} \setminus \{i'\}$;
    $k \leftarrow k + 1$;
    else
      Break **for**;
$S_t \leftarrow X_{k-1}$;
return S_t.

other devices' mini-batch size are given. Devices' optimal mini-batch size for the case of multiple local iterations can be given as follows.

Proposition 5.3 Let devices in S_t be ordered as $c_{p,t}^1 \leq c_{p,t}^2 \leq \cdots \leq c_{p,t}^{|S_t|}$. Then the devices' optimal mini-batch sizes are determined iteratively in this order, where the ith device's optimal mini-batch size for one local iteration is given by

$$D_t^{i*}(S_t) = \min \left\{ D_B^i, \max\{0, \eta\sigma \sqrt{\frac{L\gamma\phi_t \left(1 + 4L\eta(H-1)^2|S_t|\right)}{2(1-\gamma)c_{p,t}^i H}} - \sum_{j=1}^{i-1} D_t^{j*} \} \right\},$$

where $\phi_t = \sum_{h=1}^{H} (1 - \mu\eta)^{(T-t+1)H-h}$.

Similar to Proposition 5.1, Proposition 5.3 implies that a device's optimal mini-batch size is larger in a later round. We can also show that a device's optimal mini-batch size increases as the number of local iterations H increases. This is because a local update's quality can be improved by using a larger mini-batch size, and thus reduce the error caused by performing multiple local iterations.

Since we aim to find the selected device set S_t for a single round t, for ease of expression, we omit the subscript t in the device selection algorithm. Given the optimal mini-batch size design in Proposition 5.3, we select devices greedily according to their marginal contribution

which is defined by

$$M_1(S,i) = \mathcal{J}_1(S \cup \{i\}, D_t^*(S \cup \{i\})) - \mathcal{J}_1(S, D_t^*(S)), \tag{5.10}$$

where S is the selected device set. The detailed device selection algorithm in each round is given in Algorithm 8. It can be shown that the objective function given the optimal mini-batch size is not a submodular function. Therefore, it is in general difficult to find an approximation ratio for Algorithm 8.

5.5.2 The Case of Non-IID Data

Next we study the optimal device selection and mini-batch size design when devices' data are non-IID. Since devices are selected randomly in non-IID case, we aim to find the optimal participation probability $\{a_t^i\}, \forall i, t$, instead of the deterministic device selection, and minimize the expected system cost. According to the training loss upper bound given in (5.4), the problem can be formulated as

$$\begin{aligned}
&\min_{\boldsymbol{a},D} \sum_{t=1}^{T} E[\mathcal{J}_2(S_t, D_t)] \\
=&\sum_{t=1}^{T} \Bigg(\frac{L\gamma\eta^2}{2} \Bigg(\sum_{h=1}^{H} (1-\mu\eta)^{(T-t+1)H-h} \\
&\sum_{i\in\mathcal{N}} (p_i^2 \frac{\sigma_i^2}{D_t^i} + 2p_i(H-1)^2C^2 + 6Lp_id_i + p_i^2 \frac{1-a_t^i}{a_t^i} C^2) \Bigg) \\
&+(1-\gamma) \sum_{i\in\mathcal{N}} a_t^i (Hc_{p,t}^i D_t^i + c_{m,t}^i) \Bigg), \\
&s.t.\ D_t^i \le D_B^i, \forall i, t,
\end{aligned} \tag{5.11}$$

where $\boldsymbol{a} = \{a_t^i | \forall i, t\}$.

We first solve the optimal mini-batch size D given devices' participation probability $\boldsymbol{a}$, then determine the devices' optimal participation probability $\boldsymbol{a}$. It can be shown that the objective function above is a convex function of D_t^i given devices' participation probability. By utilizing the first order condition, the following result characterizes a device's optimal mini-batch size in a round.

Proposition 5.4 Given devices' participation probability $\boldsymbol{a}$, the optimal mini-batch size for device i in round t is given by

$$D_t^{i*}(\boldsymbol{a}) = \min\{D_B^i, \sigma_i \eta p_i \sqrt{\frac{L\gamma\phi_t}{2(1-\gamma)a_t^i H c_{p,t}^i}}\}. \tag{5.12}$$

Remark 5.5 Proposition 5.4 shows that, similar to the case of IID data, a device's optimal mini-batch size increases with the round index t. Furthermore, the optimal mini-batch size increases as the device's participation probability a_t^i in the round decreases. This means that, when a device participates with a small probability a_t^i, she should be allocated a larger mini-batch size to remedy for the bias caused when she does not participate, so as to ensure that the final global model is unbiased.

Given the optimal mini-batch size design in Proposition 5.4, we substitute $D_t^{i*}(\boldsymbol{a})$ into the object function $E[\mathcal{J}_2(S_t, D_t)]$. Then, we can rewrite the problem as

$$\begin{aligned}
&\min_{\boldsymbol{a}} \sum_{t=1}^{T} E[\mathcal{J}_2(S_t, \{D_t^{i*}(\boldsymbol{a})\})] \\
=&\sum_{t=1}^{T}\sum_{i\in\mathcal{N}} \Bigg(\frac{1}{2} p_i \eta \sigma_i \sqrt{2a_t^i L \phi_t \gamma (1-\gamma) H c_{p,t}^i} \\
&+ L\gamma\eta^2\phi_t \left(\frac{1}{2}\frac{1-a_t^i}{a_t^i} p_i^2 C^2 + 2p_i(H-1)^2 C^2 + 3Lp_i d_i \right) + (1-\gamma) a_t^i c_{m,t}^i \Bigg).
\end{aligned}$$

We can show that the above problem is convex in $a_t^i, \forall i, t$. Thus, the optimal participating probability a_t^{i*} is unique and can be obtained based on the first order condition.

Proposition 5.5 The optimal participating probability a_t^{i*} of device i in round t satisfies

$$2a_t^{i*2}(1-\gamma)c_{m,t}^i + a_t^{i*\frac{3}{2}} p_i \eta \sigma_i \sqrt{2L\phi_t\gamma(1-\gamma)Hc_{p,t}^i} - L\gamma\eta^2\phi_t p_i^2 C^2 = 0.$$

Although the optimal participating probability's closed form is complicated, we can get some useful insights from the above equation.

Figure 5.1: Experimental testbed of wireless federated learning consisting of a laptop as the server connected with two smartphones as devices via a WiFi router.

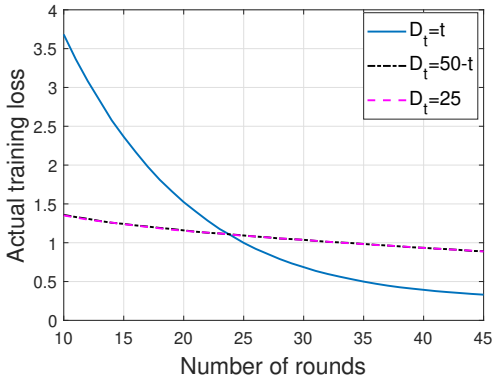

Figure 5.2: Impact of the mini-batch size on the training loss.(Simulation)

Remark 5.6 From Proposition 5.5 and 5.4, we can see that when a device has larger computation or communication costs, her optimal participating probability a_t^{i*} is smaller, and her optimal mini-batch size D_t^{i*} is larger. This result is intuitive, since a smaller a_t^{i*} can reduce the computation and communication costs, and a larger D_t^{i*} can reduce the training loss. We also observe that $a_t^{i*} \neq 0, \forall i, t$, which ensures that the FL model is unbiased.

5.6 Performance Evaluation

In this section, we conduct both simulations and testbed-based experiments to validate the theoretical findings and evaluate the proposed quality-aware adaptive algorithms. We first describe the simulation and experiment setups, and then we present the evaluation results and their analyses.

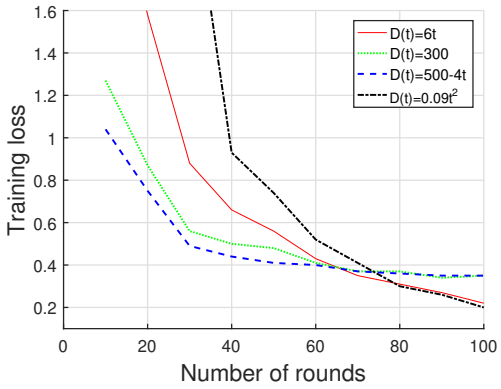

Figure 5.3: Impact of the mini-batch size on the training loss.(Experiment)

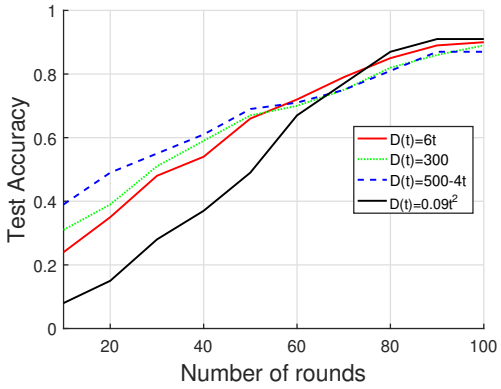

Figure 5.4: Impact of the mini-batch size on the test accuracy.(Experiment)

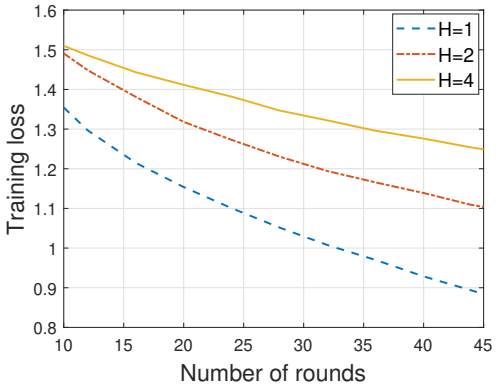

Figure 5.5: Impact of the number of local iterations on the training loss.(Simulation)

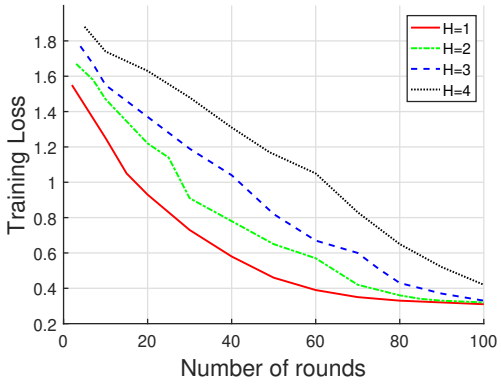

Figure 5.6: Impact of the number of local iterations on the training loss.(Experiment)

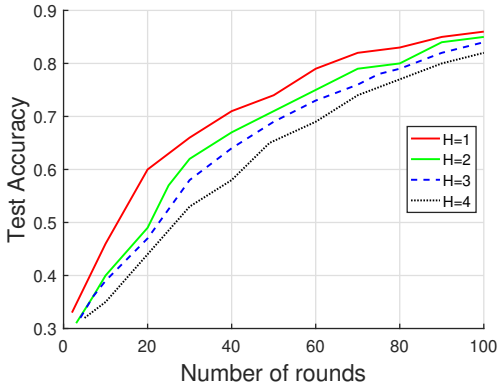

Figure 5.7: Impact of the number of local iterations on the test accuracy.(Experiment)

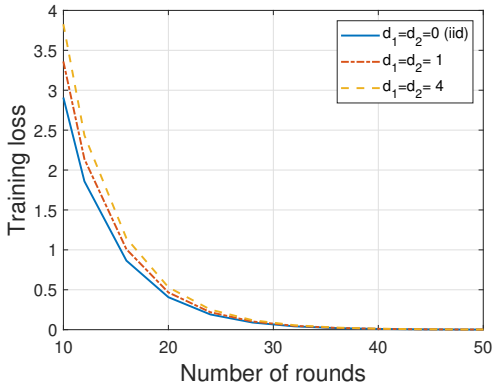

Figure 5.8: Impact of the degree of non-IID of data on the training loss.(Simulation)

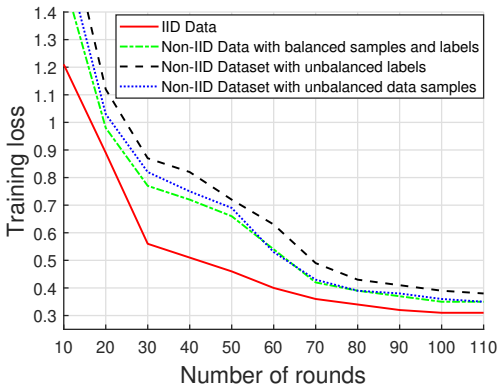

Figure 5.9: Impact of non-IID data on the training loss.(Experiment)

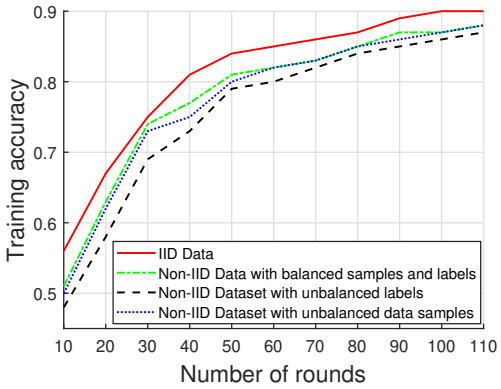

Figure 5.10: Impact of non-IID data on the test accuracy.(Experiment)

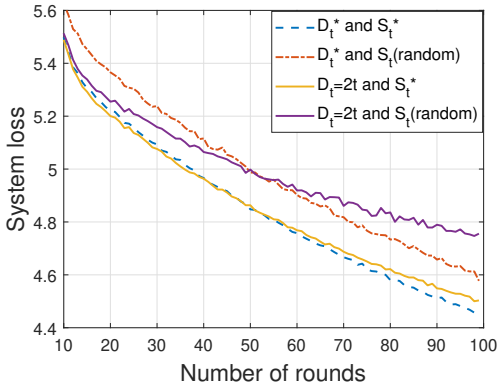

Figure 5.11: Channel-aware adaptive algorithm. (Homogeneous c_p)

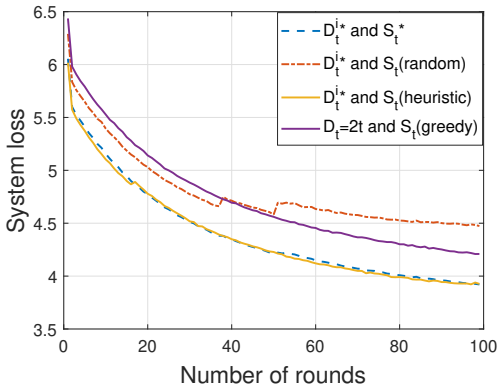

Figure 5.12: Channel-aware adaptive algorithm. (Heterogeneous c_p and single local iteration $H = 1$)

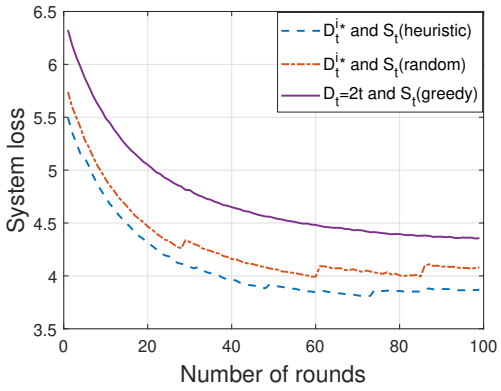

Figure 5.13: Channel-aware adaptive algorithm. (Heterogeneous c_p and multiple local iterations $H = 2$)

5.6.1 Evaluation Setup

Platforms. We build a testbed of wireless FL in edge networks consisting of a laptop as as the FL server connected with two smartphones as devices (as illustrated in Fig. 5.1). The server and devices are connected via a wireless router based on WiFi. During the experiments, the wireless router is placed 4.2 meters away from all devices. For simulations, we also implement a simulated system consisting of a virtual server and a number of virtual devices.

Datasets and Models. We use a real dataset for experiments on the wireless FL testbed and synthetic datasets for simulations on Matlab. For real data experiments, we use the widely used MNIST dataset [110] which contains 60000 pictures as training data, and 10000 pictures as test data. Each training element is a 28*28 pixel handwritten digit picture which represents numbers from 0 to 9. For synthetic data simulations, we generate 10000 data samples according to the linear model, i.e., $y = \mathbf{w}^T\mathbf{x}$, and use the mean square error function as the loss function, i.e., $f(\mathbf{w}, \xi) = \frac{1}{2}\|y - \mathbf{w}^T\mathbf{x}\|^2$, where $\xi = (\mathbf{x}, y)$ is a data sample.

5.6.2 Evaluation Results

Impact of Mini-Batch Size Distribution. We compare the training loss while using time-invariant, descending and ascending mini-batch sizes to update the global model over rounds when devices perform single local iteration. The average mini-batch size over all rounds are the same for above three distributions to achieve fair comparison. We conduct simulations for 50 global rounds, with the mean of mini-batch size over all rounds as 25. For the experiment, we divide 2000 data samples (randomly sampled from 60000 data samples) among 2 smart phones in a non-IID fashion with each device containing a balanced number of 1000 samples of 2 digits labels. We use time-invariant, descending and ascending mini-batch sizes to update the global model over rounds with the mean of the mini-batch size over all rounds as 300.

Fig. 5.2 and Fig. 5.3 show that although the average mini-batch sizes over time of three distributions are the same, different distributions of the mini-batch size result in different training loss at the end of training for both cases of IID data and non-IID data. The case of ascending mini-batch size has the worst learning accuracy in beginning rounds and results in the best

learning accuracy in ending rounds. We can see that in Fig. 5.2, the training loss while using time-invariant and descending mini-batch sizes are similar to each other. This is because that the training model used in the simulation is simple, and the training performance can be good by using just a few data samples to update the model in one round. From Fig. 5.4, we observe that the model accuracies for three distributions also show the expected result. With the same mean of mini-batch size, the setting of ascending mini-batch size achieves the highest model accuracy in ending rounds. This conforms the result from Theorem 5.1 that the update in a later round has a larger impact on the learning accuracy. The results in the experiment also show that, for real applications, the ascending mini-batch size should be used to improve the learning accuracy.

Impact of the Number of Local Iterations. We compare the training loss while devices perform uniform numbers of multiple local iterations ($H \in \{1, 2, 3, 4\}$). For the simulation, we simulate for 50 local iterations in total, with the mini-batch size in each local iteration set as 25. For the experiment, we divide 2000 data samples (randomly sampled from 60000 data samples) among 2 smartphones in a non-IID fashion with each device holds 1000 samples of 2 digits labels.

From Fig. 5.5 and Fig. 5.6, we can see that when devices perform single local iteration ($H = 1$), the system suffers the lowest training loss. The training loss increases as the number of local iterations increases. Fig. 5.7 shows the test accuracy for different number of local iterations. The result shows that the test accuracy when devices perform single local iteration is always the highest. From the figures we can see that the experiment results are consistent with the result given in Theorem 5.1. FL suffers a larger error caused by the randomness of data with less communication rounds.

Impact of Non-IID Datasets. We evaluate the impact of the degree of non-IID data on the training loss while devices perform single local iteration. For the simulation, we use three kinds of set ups for devices' datasets. The first set up is that devices' data follows IID (i.e., $d_i = 0$). The second set up is that devices' data is unbalanced (non-IID), which contains two kinds of datasets and their heterogeneity degrees are set as $d_1 = d_2 = 1$. The third set up is

also unbalanced, which contains two kinds of datasets with higher heterogeneity degrees that are set as $d_1 = d_2 = 4$.

For the experiment, we use four kinds of set ups for devices' datasets which are set as follows: 1) we divide 3000 data samples in an IID fashion where each device holds a balanced number of 1500 data samples of 10 digits labels (which means either smart phone holds 150 data samples for each label); 2) we divide 3000 data samples (randomly sampled from 60000 data samples) among 2 smart phones in a non-IID fashion with each device holding 1500 data samples of 5 digits labels (which means for each label, either smart phone holds 300 data samples); 3) we set two edge devices hold the same amount of data samples in a non-IID fashion with one device holds data samples of 5 digit labels and another device holds data samples of 4 digit labels, respectively; 4) we set one device holds 2000 data samples and another device holds 200 data samples, respectively, in a non-IID fashion with each device holds data samples of 5 digit labels.

From Fig. 5.8, we can see that the case of IID data always has the lowest training loss, and for the two settings with non-IID datasets, the greater the degree of non-IID, the higher the training loss that FL suffers, which conforms to the result given in Theorem 5.2. From Fig. 5.9 and Fig 5.10, we can see that the case of IID data always has the lowest training loss and the highest test accuracy. In the cases of non-IID data, the case that devices' data samples and labels are balanced has the best training loss and test accuracy, and the case that devices' data samples are unbalanced has the worst training loss and test accuracy. This implies that the degree of heterogeneous is higher when both devices' data and labels are unbalanced compared to that when only devices' labels are unbalanced.

Cost-Effective Channel-Aware Adaptive Algorithm. We also evaluate the cost-effective channel-aware adaptive device selection and mini-batch size design algorithms proposed in Section 5.5 for the case of IID data.

We first evaluate the case where devices perform single local iteration and have homogeneous computation capabilities. We compare the system loss of 4 device selection and mini-batch size designs with the same average mini-batch sizes over time: 1) the optimal device

selection (Proposition 5.2) and mini-batch size design (Proposition 6.4); 2) random device selection and the optimal mini-batch size design (Proposition 6.4); 3) the optimal device selection (Proposition 5.2) and linearly increasing mini-batch size; 4) random device selection and linearly increasing mini-batch size. We observe from Fig. 5.11 that the proposed channel-aware adaptive algorithm results in the lowest system loss, which confirms the effectiveness of the proposed algorithm.

Then, we evaluate the case where devices perform single local iteration and have heterogeneous computation capabilities. We compare the system loss of 4 device selection and mini-batch size designs with the same average mini-batch sizes over time: 1) the device selection given by Algorithm 1 and optimal mini-batch size design (Proposition 6.4); 2) random device selection and the optimal mini-batch size design (Proposition 6.4); 3) the heuristic device selection where devices are selected according to their marginal contribution (Algorithm 2 with $H = 1$) to the system loss and the optimal mini-batch size design (Proposition 6.4); 4) greedy device selection where devices with the lowest communication cost are selected and linearly increasing mini-batch size. From Fig. 5.12, we observe that the proposed channel-aware adaptive algorithm and the heuristic device selection algorithm result in the lowest system loss. The result confirms the effectiveness of the proposed algorithm. Besides, we can see that although the approximation ratio of the heuristic algorithm is hard to find, it can achieve the same performance as the proposed channel-aware adaptive algorithm under some conditions.

Last, we evaluate the case where devices perform multiple local iterations ($H = 2$) and have heterogeneous computation capabilities. We compare the system loss of the last 3 device selection and mini-batch size designs for the case where devices perform single local iterations. From Fig. 5.13, we can see that the proposed channel-aware adaptive algorithm results in the lowest system loss, which confirms the effectiveness of the proposed algorithm.

5.7 Conclusion

In this chapter, we study performance bounds on the learning accuracy of FL as a function of devices' mini-batch sizes in each local iteration. The results show that the impact of devices' quality increases with the learning process. We also develop cost-effective dynamic distributed

learning algorithms which optimize the learning accuracy by adaptively selecting devices and their mini-batch sizes. Simulations based on both synthetic data and real data are demonstrated to evaluate the proposed algorithms.

5.8 Appendix

5.8.1 Proof of Theorem 5.1

From (5.1), We have

$$\left\|\bar{\mathbf{w}}_{t,H} - \mathbf{w}^*\right\|^2 \tag{5.13}$$

$$= \left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^* - \eta g_{t,H-1} - \eta\bar{g}_{t,H-1} + \eta\bar{g}_{t,H-1}\right\|^2 \tag{5.14}$$

$$= \underbrace{\left\|\bar{\mathbf{w}}_{t,H-1} - \eta\bar{g}_{t,H-1} - \mathbf{w}^*\right\|^2}_{A_1} + \underbrace{\left\|\eta\bar{g}_{t,H-1} - \eta g_{t,H-1}\right\|^2}_{A_2} \tag{5.15}$$

$$+ \underbrace{2\left\langle\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^* - \eta\bar{g}_{t,H-1}, \eta\bar{g}_{t,H-1} - \eta g_{t,H-1}\right\rangle}_{A_3}. \tag{5.16}$$

Note that $E[A_3] = 0$. We next focus on bounding A_1. Since F is L-smooth and μ-strongly convex, we have

$$\begin{aligned}
A_1 &= \left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*\right\|^2 + \eta^2\bar{g}_{t,H-1}^2 - 2\eta\left\langle\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*, \bar{g}_{t,H-1}\right\rangle \\
&= \left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*\right\|^2 + \eta^2\bar{g}_{t,H-1}^2 - 2\eta\sum_{i\in S_t}\frac{D_{t,H}^i}{D_{t,H}}\left\langle\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*, \bar{g}_{t,H-1}^i\right\rangle \\
&\leq \left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*\right\|^2 + 2\eta^2 L\left(F\left(\bar{\mathbf{w}}_{t,H-1}\right) - F\left(\mathbf{w}^*\right)\right) \\
&- 2\eta\sum_{i\in S_t}\frac{D_{t,H}^i}{D_{t,H}}\left\langle\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}_{t,H-1}^i + \mathbf{w}_{t,H-1}^i - \mathbf{w}^*, \bar{g}_{t,H-1}^i\right\rangle, \\
&\leq \left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*\right\|^2 + 2\eta L\sum_{i\in S_t}\frac{D_{t,H}^i}{D_{t,H}}\left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}_{t,H-1}^i\right\|^2 \\
&\quad + 2\eta\sum_{i\in S_t}\frac{D_{t,H}^i}{D_{t,H}}\left(\left(\eta L - \frac{1}{2}\right)\left(F\left(\mathbf{w}_{t,H-1}^i\right) - F(\mathbf{w}^*)\right) - \frac{\mu}{2}\left\|\mathbf{w}_{t,H-1}^i - \mathbf{w}^*\right\|^2\right).
\end{aligned}$$

Next, we bound

$$
\begin{aligned}
& E\left\|\bar{\mathbf{w}}_{t,H-1}-\mathbf{w}_{t,H-1}^{i}\right\|^{2} \\
=& E\left\|\mathbf{w}_{t,H-1}^{i}-\mathbf{w}_{t,1}-\left(\bar{\mathbf{w}}_{t,H-1}-\mathbf{w}_{t,1}\right)\right\|^{2} \\
\leq & E\left\|\mathbf{w}_{t,H-1}^{i}-\mathbf{w}_{t,1}\right\|^{2} \\
=& \eta^{2} E\|\sum_{h=1}^{H-1} g_{t,h}^{i}\|^{2} \\
\leq & 2\eta^{2}\left[E\|\sum_{h=1}^{H-1}(g_{t,h}^{i}-\bar{g}_{t,h})\|^{2}+E\|\sum_{h=1}^{H-1}\bar{g}_{t,h}\|^{2}\right] \\
\leq & 2(H-1)\eta^{2}\left[\sum_{h=1}^{H-1} E\|(g_{t,h}^{i}-\bar{g}_{t,h})\|^{2}+\sum_{h=1}^{H-1} E\|\bar{g}_{t,h}\|^{2}\right] \\
\leq & 2(H-1)^{2}\eta^{2}(\frac{\sigma^{2}}{D_{t}^{i}}+B^{2}),
\end{aligned}
\tag{5.17}
$$

where the last inequality follows from the definition of a user's quality.

For $\eta \leq \frac{1}{4L}$ and by the convexity of $a\left(F(\mathbf{w})-F(\mathbf{w}^{*})\right)+b\left\|\mathbf{w}-\mathbf{w}^{*}\right\|^{2}$ for $a, b \geq 0$, we have

$$
\begin{aligned}
& E\left\|\bar{\mathbf{w}}_{T,H}-\mathbf{w}^{*}\right\|^{2} \\
\leq & \left(1-\mu\eta\right)\left\|\bar{\mathbf{w}}_{T,H-1}-\mathbf{w}^{*}\right\|^{2}-\frac{\eta}{2}\left(F\left(\bar{\mathbf{w}}_{T,H-1}\right)-F\left(\mathbf{w}^{*}\right)\right) \\
& +\frac{\eta^{2}\sigma^{2}}{D_{T}}+2\eta L\sum_{i\in S_{T}}\frac{D_{T}^{i}}{D_{T}}\left\|\bar{\mathbf{w}}_{T,H-1}-\mathbf{w}_{T,H-1}^{i}\right\|^{2} \\
\leq & \left(1-\mu\eta\right)\left\|\bar{\mathbf{w}}_{T,H-1}-\mathbf{w}^{*}\right\|^{2}+\frac{\eta^{2}\sigma^{2}}{D_{T}}+4L\eta^{3}\sum_{i\in S_{T}}(H-1)^{2}(\frac{\sigma^{2}}{D_{T,H}}+\frac{D_{T}^{i}}{D_{T,H}}B^{2}) \\
\leq & \left(1-\mu\eta\right)^{TH}\left\|\mathbf{w}_{0}-\mathbf{w}^{*}\right\|^{2}+\sum_{t=1}^{T}\sum_{h=1}^{H}\Big(\left(1-\mu\eta\right)^{TH-(t-1)H-h} \\
& \left(\frac{\eta^{2}\sigma^{2}}{D_{t}}+4L\eta^{3}\sum_{i\in S_{t}}(H-1)^{2}(\frac{\sigma^{2}}{D_{t}}+\frac{D_{t}^{i}}{D_{t}}B^{2})\right)\Big).
\end{aligned}
$$

Thus we have

$$
\begin{aligned}
&E[F(\mathbf{w}_T) - F(\mathbf{w}^*)] \\
&\leq \frac{L}{2}(1-\mu\eta)^{TH}\|\mathbf{w}_0 - \mathbf{w}^*\|^2 + \frac{L}{2}\sum_{t=1}^{T}\sum_{h=1}^{H}\Big((1-\mu\eta)^{TH-(t-1)H-h} \\
&\quad\left(\frac{\eta^2\sigma^2}{D_t} + 4L\eta^3(H-1)^2(\frac{\sigma^2|S_t|}{D_t} + B^2)\right)\Big).
\end{aligned}
$$

5.8.2 Proof of Lemma 5.1

By taking expectation over the selected device set S_t, we have

$$
\begin{aligned}
E_{S_t}[\mathbf{w}_t] =& \mathbf{w}_{t-1} + E_{S_t}\left[\sum_{i\in\mathcal{N}} \mathbf{1}\{i \in S_t\}\frac{p_i}{a_t^i}(\mathbf{w}_{t,H}^i - \mathbf{w}_{t-1})\right] \\
=& \mathbf{w}_{t-1} + \sum_{i\in\mathcal{N}} a_t^i \frac{p_i}{a_t^i}(\mathbf{w}_{t,H}^i - \mathbf{w}_{t-1}) \\
=& \mathbf{w}_{t-1} + \sum_{i\in\mathcal{N}} p_i(\mathbf{w}_{t,H}^i - \mathbf{w}_{t-1}) \\
=& \tilde{\mathbf{w}}_t,
\end{aligned}
$$

where $\mathbf{1}\{\cdot\}$ is a indicator function.

5.8.3 Proof of Theorem 5.2

First, we bound $\|\mathbf{w}_t - \mathbf{w}^*\|^2$ as in (5.16), then we bound the terms A_1, A_2, and A_3 therein. For A_1, we have

$$
A_1 = \|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*\|^2 + \underbrace{\eta^2\|\bar{g}_{t,H-1}\|^2}_{B_1} - \underbrace{2\eta\left\langle \bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*, \bar{g}_{t,H-1}\right\rangle}_{B_2}. \tag{5.18}
$$

By the convexity of $\|\cdot\|^2$ and the L-smoothness of F_i, we have

$$
\begin{aligned}
B_1 = \eta^2\|\bar{g}_{t,H-1}\|^2 &\leq \eta^2\sum_{i\in\mathcal{N}} p_i\left\|\bar{g}_{t,H-1}^i\right\|^2 \\
&\leq 2L\eta^2\sum_{i\in\mathcal{N}} p_i(F_i(\mathbf{w}_{t,H-1}^i) - F_i(\mathbf{w}_i^*)).
\end{aligned} \tag{5.19}
$$

Next we bound B_2, we have

$$\begin{aligned}
B_2 &= -2\eta\langle\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*, \bar{g}_{t,H-1}\rangle \\
&= -2\eta \sum_{i\in\mathcal{N}} p_i \langle\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*, \bar{g}^i_{t,H-1}\rangle \\
&= -2\eta \sum_{i\in\mathcal{N}} p_i \langle\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^i_{t,H-1}, \bar{g}^i_{t,H-1}\rangle \\
&- 2\eta \sum_{i\in\mathcal{N}} p_i \langle\mathbf{w}^i_{t,H-1} - \mathbf{w}^*, \bar{g}^i_{t,H-1}\rangle.
\end{aligned}$$

We use Cauchy-Schwarz inequality and AM-GM inequality to bound the first term:

$$-\langle\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^i_{t,H-1}, \bar{g}^i_{t,H-1}\rangle \leq \frac{1}{\eta}\left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^i_{t,H-1}\right\|^2 + \eta\left\|\bar{g}^i_{t,H-1}\right\|^2. \tag{5.20}$$

Then we use the μ-strong convexity of F_i to bound the second term:

$$-\left\langle\mathbf{w}^i_{t,H-1} - \mathbf{w}^*, \bar{g}^i_{t,H-1}\right\rangle \leq -\left(F_i\left(\mathbf{w}^i_{t,H-1}\right) - F_i\left(\mathbf{w}^*\right)\right) - \frac{\mu}{2}\left\|\mathbf{w}^i_{t,H-1} - \mathbf{w}^*\right\|^2. \tag{5.21}$$

Substituting (5.19), (5.20), and (5.21) into (7.13), we have

$$\begin{aligned}
A_1 &\leq \|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*\|^2 + 2L\eta^2 \sum_{i\in\mathcal{N}} p_i (F_i(\mathbf{w}^i_{t-1}) - F_i(\mathbf{w}^*_i)) \\
&+ \sum_{i\in\mathcal{N}} p_i \left(\left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^i_{t,H-1}\right\|^2 + \eta^2\left\|\bar{g}^i_{t,H-1}\right\|^2\right) \\
&- 2\eta \sum_{i\in\mathcal{N}} p_i \left(F_i(\mathbf{w}^i_{t,H-1}) - F_i(\mathbf{w}^*) + \frac{\mu}{2}\left\|\mathbf{w}^i_{t,H-1} - \mathbf{w}^*\right\|^2\right) \\
&\leq (1-\mu\eta)\|\mathbf{w}_{t,H-1} - \mathbf{w}^*\|^2 + \sum_{i\in\mathcal{N}} p_i \left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^i_{t,H-1}\right\|^2 \\
&+ \underbrace{4L\eta^2 \sum_{i\in\mathcal{N}} p_i (F_i(\mathbf{w}^i_{t,H-1}) - F_i(\mathbf{w}^*_i)) - 2\eta \sum_{i\in\mathcal{N}} p_i \left(F_i(\mathbf{w}^i_{t,H-1}) - F_i(\mathbf{w}^*)\right)}_{C_1},
\end{aligned}$$

in which C_1 can be bounded as

$$\begin{aligned}C_1 =& 4L\eta^2 \sum_{i\in\mathcal{N}} p_i (F_i(\mathbf{w}^*) - F_i(\mathbf{w}_i^*)) - 2\eta(1-2L\eta) \sum_{i\in\mathcal{N}} p_i (F_i(\mathbf{w}_{t,H-1}^i) - F_i(\mathbf{w}^*)) \\ \leq& 4L\eta^2 \sum_{i\in\mathcal{N}} p_i d_i - 2\eta(1-2L\eta) \underbrace{\sum_{i\in\mathcal{N}} p_i (F_i(\mathbf{w}_{t,H-1}^i) - F_i(\mathbf{w}^*))}_{D}.\end{aligned}$$

Next, we bound D.

$$\begin{aligned}D =& \sum_{i\in\mathcal{N}} p_i (F_i(\mathbf{w}_{t,H-1}^i) - F_i(\bar{\mathbf{w}}_{t,H-1}) + F_i(\bar{\mathbf{w}}_{t,H-1}) - F_i(\mathbf{w}^*)) \\ \geq& \sum_{i\in\mathcal{N}} p_i \langle \nabla F_i(\bar{\mathbf{w}}_{t,H-1}), \mathbf{w}_{t,H-1}^i - \bar{\mathbf{w}}_{t,H-1} \rangle + \sum_{i\in\mathcal{N}} p_i (F_i(\bar{\mathbf{w}}_{t,H-1}) - F_i(\mathbf{w}^*)) \\ \geq& -\frac{1}{2} \sum_{i\in\mathcal{N}} p_i \left(\eta \left\| \nabla F_i(\bar{\mathbf{w}}_{t,H-1}) \right\|^2 + \frac{1}{\eta} \left\| \mathbf{w}_{t,H-1}^i - \bar{\mathbf{w}}_{t,H-1} \right\|^2 \right) + \sum_{i\in\mathcal{N}} p_i (F_i(\bar{\mathbf{w}}_{t,H-1}) - F_i(\mathbf{w}^*)) \\ \geq& -\sum_{i\in\mathcal{N}} p_i \left(\eta L \left(F_i(\bar{\mathbf{w}}_{t,H-1} - F_i(\mathbf{w}_i^*)) + \frac{1}{2\eta} \left\| \mathbf{w}_{t,H-1}^i - \bar{\mathbf{w}}_{t,H-1} \right\|^2 \right) \right. \\ & \left. + \sum_{i\in\mathcal{N}} p_i (F_i(\bar{\mathbf{w}}_{t,H-1}) - F_i(\mathbf{w}^*)). \right.\end{aligned}$$

Then we can bound C_1 as

$$\begin{aligned}C_1 \leq& 4L\eta^2 \sum_{i\in\mathcal{N}} p_i d_i - 2\eta(1-2L\eta) \left(-\sum_{i\in\mathcal{N}} p_i \right. \\ & \left. \left(\eta L \left(F_i(\bar{\mathbf{w}}_{t,H-1} - F_i(\mathbf{w}_i^*)) + \frac{1}{2\eta} \left\| \mathbf{w}_{t,H-1}^i - \bar{\mathbf{w}}_{t,H-1} \right\|^2 \right) + \sum_{i\in\mathcal{N}} p_i (F_i(\bar{\mathbf{w}}_{t,H-1}) - F_i(\mathbf{w}^*)) \right) \right. \\ \leq& 4L\eta^2 \sum_{i\in\mathcal{N}} p_i d_i - 2\eta(1-2L\eta) \left(-\sum_{i\in\mathcal{N}} p_i \left((\eta L - 1) \left(F_i(\bar{\mathbf{w}}_{t,H-1}) - F_i(\mathbf{w}^*) \right) \right) \right. \\ & \left. \left. -\eta L d_i + \frac{1}{2\eta} \left\| \mathbf{w}_{t,H-1}^i - \bar{\mathbf{w}}_{t,H-1} \right\|^2 \right) \right) \\ =& 2\eta(1-2L\eta)(\eta L - 1) \sum_{i\in\mathcal{N}} p_i \left(F_i(\bar{\mathbf{w}}_{t,H-1} - F_i(\mathbf{w}_i^*)) \right) \\ & + (4L\eta^2 + 2L\eta^2(1-2L\eta)) \sum_{i\in\mathcal{N}} p_i d_i + (1-2L\eta) \sum_{i\in\mathcal{N}} p_i \left\| \mathbf{w}_{t,H-1}^i - \bar{\mathbf{w}}_{t,H-1} \right\|^2 \\ \leq& 6L\eta^2 \sum_{i\in\mathcal{N}} p_i d_i + \sum_{i\in\mathcal{N}} p_i \left\| \mathbf{w}_{t,H-1}^i - \bar{\mathbf{w}}_{t,H-1} \right\|^2.\end{aligned}$$

Thus we can further bound A_1 as

$$\begin{aligned} &E[A_1] \\ \leq&(1-\mu\eta)\left\|\bar{\mathbf{w}}_{t,H-1}-\mathbf{w}^*\right\|^2+6L\eta^2\sum_{i\in\mathcal{N}}p_i d_i+\sum_{i\in\mathcal{N}}p_i\left\|\mathbf{w}_{t,H-1}^i-\bar{\mathbf{w}}_{t,H-1}\right\|^2 \end{aligned} \tag{5.22}$$

Next we bound A_2. Let $\hat{g}_{t,h}$ denote $\sum_{i\in\mathcal{N}}(\mathbf{w}_{t,h}^i-\mathbf{w}_{t,h-1}^i)$. We have

$$\begin{aligned} A_2=&\eta^2\left\|\bar{g}_{t,H-1}-\hat{g}_{t,H-1}+\hat{g}_{t,H-1}-g_{t,H-1}\right\|^2 \\ =&\eta^2\left\|\bar{g}_{t,H-1}-\hat{g}_{t,H-1}\right\|^2+\eta^2\left\|\hat{g}_{t,H-1}-g_{t,H-1}\right\|^2, \end{aligned} \tag{5.23}$$

where the second equality comes from Lemma 5.1.

From the definition of the quality of a device, we have

$$\eta^2 E\left\|\bar{g}_{t,H-1}-\hat{g}_{t,H-1}\right\|^2\leq\eta^2\sum_{i\in\mathcal{N}}{p_i}^2\frac{\sigma_i^2}{D_t^i}. \tag{5.24}$$

We also have

$$\begin{aligned} &E_{S_t}\left\|\hat{g}_{t,H-1}-g_{t,H-1}\right\|^2=E_{S_t}\left\|\sum_{i\in\mathcal{N}}p_i g_{t,H-1}^i-\sum_{i\in S_t}\frac{p_i}{a_t^i}g_{t,H-1}^i\right\|^2 \\ =&E_{S_t}\left\|\sum_{i\in S_t}\frac{p_i}{a_t^i}g_{t,H-1}^i\right\|^2+E\left\|\sum_{i\in\mathcal{N}}p_i g_{t,H-1}^i\right\|^2-2E_{S_t}\langle\sum_{i\in S_t}\frac{p_i}{a_t^i}g_{t,H-1}^i,\sum_{i\in\mathcal{N}}p_i g_{t,H-1}^i\rangle \\ =&E_{S_t}\left\|\sum_{i\in S_t}\frac{p_i}{a_t^i}g_{t,H-1}^i\right\|^2-E\left\|\sum_{i\in\mathcal{N}}p_i g_{t,H-1}^i\right\|^2 \\ =&E\left\|\sum_{i\in\mathcal{N}}\mathbf{1}\{i\in S_t\}\frac{p_i}{a_t^i}g_{t,H-1}^i\right\|^2-E\left\|\sum_{i\in\mathcal{N}}p_i g_{t,H-1}^i\right\|^2 \\ =&\sum_{i\in\mathcal{N}}\frac{p_i^2}{a_t^i}E\left\|g_{t,H-1}^i\right\|^2+\sum_{i\neq j}a_t^i a_t^j E\langle\frac{p_i}{a_t^i}g_{t,H-1}^i,\frac{p_j}{a_t^j}g_{t,H-1}^j\rangle-E\left\|\sum_{i\in\mathcal{N}}p_i g_{t,H-1}^i\right\|^2 \\ =&\sum_{i\in\mathcal{N}}\frac{p_i^2}{a_t^i}E\left\|g_{t,H-1}^i\right\|^2-\sum_{i\in\mathcal{N}}p_i^2E\left\|g_{t,H-1}^i\right\|^2\leq\sum_{i\in\mathcal{N}}p_i^2\frac{1-a_t^i}{a_t^i}C^2 \end{aligned} \tag{5.25}$$

Note that $E[A_3] = 0$. Combining (5.16), (7.19), (7.14), and (7.20), we have

$$
\begin{aligned}
&E\left\|\bar{\mathbf{w}}_{T,H} - \mathbf{w}^*\right\|^2 \\
&\leq (1-\mu\eta)E\left\|\bar{\mathbf{w}}_{T,H-1} - \mathbf{w}^*\right\|^2 + \eta^2 \sum_{i\in\mathcal{N}} {p_i}^2 \frac{\sigma_i^2}{D_T^i} \\
&+ \sum_{i\in\mathcal{N}} (6L\eta^2 p_i d_i + 2\eta^2 p_i (H-1)^2 C^2 + \eta^2 p_i^2 \frac{1-a_t^i}{a_t^i} C^2) \\
&\leq (1-\mu\eta)^{TH} E\left\|\mathbf{w}_0 - \mathbf{w}^*\right\|^2 + \eta^2 \sum_{t=1}^{T}\sum_{h=1}^{H} \Big[(1-\mu\eta)^{TH-(t-1)H-h} \\
&\sum_{i\in\mathcal{N}} ({p_i}^2 \frac{\sigma_i^2}{D_t^i} + 6Lp_i d_i + 2p_i(H-1)^2 C^2 + p_i^2 \frac{1-a_t^i}{a_t^i} C^2)\Big].
\end{aligned}
$$

Thus we have

$$
\begin{aligned}
&E[F(\mathbf{w}_T) - F(\mathbf{w}^*)] \leq \frac{L}{2}(1-\mu\eta)^{TH} E\left\|\mathbf{w}_0 - \mathbf{w}^*\right\|^2 + \\
&\frac{L\eta^2}{2} \sum_{t=1}^{T}\sum_{h=1}^{H} \left[(1-\mu\eta)^{TH-(t-1)H-h} \sum_{i\in\mathcal{N}} ({p_i}^2 \frac{\sigma_i^2}{D_t^i} + 6Lp_i d_i + 2p_i(H-1)^2 C^2 + p_i^2 \frac{1-a_t^i}{a_t^i} C^2)\right].
\end{aligned}
$$

5.8.4 Proof of Lemma 5.2

First, we have that $-\mathcal{J}_1(S_t, D_t^*(S_t))$ is a non-monotone function, since its first term increases with $|S_t|$ and its second term decreases with $|S_t|$.

Next, we prove that $-\mathcal{J}_1(S_t, D_t^*(S_t))$ is a submodular function. For ease of expression, we write $-\mathcal{J}_1(S_t, D_t^*(S_t))$ as $-\mathcal{J}_1(S_t)$ in the following proof.

We can see that a user's optimal data sampling size has three possible values which are 0, $\eta\sigma\sqrt{\frac{L\gamma a_t}{2(1-\gamma)c_{p,t}^i}} - \sum_{j=1}^{i-1} D_t^{i*}$, and D_B^i. From Theorem 6.4, we know that for any selected user set $S_1 \subset S_2$,

$$
\sum_{i\in S_1} D_t^{i*}(S_1) \leq \sum_{i\in S_2} D_t^{i*}(S_2). \tag{5.26}
$$

We also know that after reordering users, the optimal data sampling size of user j' is always no greater than that of user j for any $j' > j$. Thus we have, for any selected user set $S_1 \subset S_2$ and any user $x \in S_2$, there are four possible combinations of $D_t^{x*}(S_1 \cup \{x\})$ and $D_t^{x*}(S_2 \cup \{x\})$:

1) $D_t^{x*}(S_1 \cup \{x\}) = D_t^{x*}(S_2 \cup \{x\})$;

2) $D_t^{x*}(S_1 \cup \{x\}) = D_B^x$ and $D_t^{x*}(S_2 \cup \{x\}) = \eta\sigma\sqrt{\frac{L\gamma a_t}{2(1-\gamma)c_{p,t}^x}} - \sum_{i=1}^{x-1} D_t^{i*}(S_2)$;

3) $D_t^{x*}(S_1 \cup \{x\}) = D_B^x$ and $D_t^{x*}(S_2 \cup \{x\}) = 0$;

4) $D_t^{x*}(S_1 \cup \{x\}) = \eta\sigma\sqrt{\frac{L\gamma a_t}{2(1-\gamma)c_{p,t}^x}} - \sum_{i=1}^{x-1} D_t^{i*}(S_1)$ and $D_t^{x*}(S_2 \cup \{x\}) = 0$.

We also give the definition of submodular functions.

Definition 5.1 A set function f on S is submodular if and only if

$$f(S_1 \cup \{x\}) - f(S_1) \geq f(S_2 \cup \{x\}) - f(S_2)$$

for each $S_1 \subset S_2 \subseteq S$ and $x \in S \setminus S_2$.

For any round t, selected user set S_t and user $x \notin S_t$, we have

$$\begin{aligned}
&-\mathcal{J}(S_t \cup \{x\}) - (-\mathcal{J}(S_t)) \\
&= -\gamma \left(\frac{La_t\eta^2\sigma^2}{2\sum\limits_{i \in S_t \cup \{x\}} D_t^{i*}(S_t \cup \{x\}))} \right) + \gamma \left(\frac{La_t\eta^2\sigma^2}{2\sum\limits_{i \in S_t} D_t^{i*}(S_t)} \right) \\
&- (1-\gamma) \sum_{i \in S_t \cup \{x\}} (c_{p,t}^i D_t^{i*}(S_t \cup \{x\}) + c_{m,t}^i) + (1-\gamma) \sum_{i \in S_t} (c_{p,t}^i D_t^{i*}(S_t) + c_{m,t}^i) \\
&= \frac{\gamma L a_t \eta^2 \sigma^2}{2} \left(\frac{D_t^{x*}(S_t \cup \{x\})}{\sum\limits_{i \in S_t} D_t^{i*}(S_t) \sum\limits_{i \in S_t \cup \{x\}} D_t^{i*}(S_t \cup \{x\})} \right) - (1-\gamma) \left(c_{p,t}^x D_t^{x*}(S_t \cup \{x\}) + c_{m,t}^x \right)
\end{aligned} \tag{5.27}$$

Then, for each combination of $D_t^{x*}(S_1 \cup \{x\})$ and $D_t^{x*}(S_2 \cup \{x\})$, we prove that $-\mathcal{J}(S_1 \cup \{x\}) + \mathcal{J}(S_1) \geq -\mathcal{J}(S_2 \cup \{x\}) + \mathcal{J}(S_2)$.

1) $D_t^{x*}(S_1 \cup \{x\}) = D_t^{x*}(S_2 \cup \{x\})$:

From (5.26) and (5.27), we have

$$
\begin{aligned}
&- \mathcal{J}(S_1 \cup \{x\}) + \mathcal{J}(S_1) - (-\mathcal{J}(S_2 \cup \{x\}) + \mathcal{J}(S_2)) \\
&= \frac{\gamma L a_t \eta^2 \sigma^2}{2} \left(\frac{D_t^{x*}(S_1 \cup \{x\})}{\sum\limits_{i \in S_1} D_t^{i*}(S_1) \sum\limits_{i \in S_1 \cup \{x\}} D_t^{i*}(S_1 \cup \{x\})} \right) \\
&- \frac{\gamma L a_t \eta^2 \sigma^2}{2} \left(\frac{D_t^{x*}(S_2 \cup \{x\})}{\sum\limits_{i \in S_2} D_t^{i*}(S_2) \sum\limits_{i \in S_2 \cup \{x\}} D_t^{i*}(S_2 \cup \{x\})} \right) \\
&\geq 0.
\end{aligned}
$$

2) $D_t^{x*}(S_1 \cup \{x\}) = D_B^x$ and $D_t^{x*}(S_2 \cup \{x\}) = \eta\sigma\sqrt{\frac{L\gamma a_t}{2(1-\gamma)c_{p,t}^i}} - \sum_{i \in S_2} D_t^{i*}$:

We have

$$
D_B^x < \eta\sigma\sqrt{\frac{L\gamma a_t}{2(1-\gamma)c_{p,t}^x}} - \sum_{j \in S_1} D_t^{i*}(S_1)
$$

and

$$
D_B^x > \eta\sigma\sqrt{\frac{L\gamma a_t}{2(1-\gamma)c_{p,t}^x}} - \sum_{j \in S_2} D_t^{i*}(S_2).
$$

Thus we have

$$
\frac{\gamma L a_t \eta^2 \sigma^2/(1-\gamma)}{2(D_B^x + \sum\limits_{i \in S_1} D_t^{i*}(S_1))^2} \leq c_{p,t}^x \leq \frac{\gamma L a_t \eta^2 \sigma^2/(1-\gamma)}{2(D_B^x + \sum\limits_{i \in S_2} D_t^{i*}(S_2))^2}. \tag{5.28}
$$

From (5.27) and (5.28), we have

$$
\begin{aligned}
&- \mathcal{J}(S_1 \cup \{x\}) + \mathcal{J}(S_1) - (-\mathcal{J}(S_2 \cup \{x\}) + \mathcal{J}(S_2)) \\
&\geq \frac{\gamma L a_t \eta^2 \sigma^2}{2} \left[\frac{D_B^x}{\sum\limits_{i \in S_1} D_t^{i*}(S_1)(D_B^x + \sum\limits_{i \in S_1} D_t^{i*}(S_1))} - \frac{\eta\sigma\sqrt{\frac{L\gamma a_t}{2(1-\gamma)c_{p,t}^i}} - \sum_{i \in S_2} D_t^{i*}(S_2)}{\sum\limits_{i \in S_2} D_t^{i*}(S_2)(D_B^x + \sum\limits_{i \in S_2} D_t^{i*}(S_2))} - \right. \\
&\left. \frac{D_B^x}{(D_B^x + \sum\limits_{i \in S_1} D_t^{i*}(S_1))^2} + \frac{\eta\sigma\sqrt{\frac{L\gamma a_t}{2(1-\gamma)c_{p,t}^i}} - \sum\limits_{i \in S_2} D_t^{i*}(S_2)}{(D_B^x + \sum\limits_{i \in S_2} D_t^{i*}(S_2))^2} \right] \\
&= \frac{{D_B^x}^2}{\sum\limits_{i \in S_1} D_t^{i*}(S_1)(D_B^x + \sum\limits_{i \in S_1} D_t^{i*}(S_1))^2} - \frac{(\eta\sigma\sqrt{\frac{L\gamma a_t}{2(1-\gamma)c_{p,t}^i}} - \sum_{i \in S_2} D_t^{i*}(S_2))^2}{\sum\limits_{i \in S_2} D_t^{i*}(S_2)(D_B^x + \sum\limits_{i \in S_2} D_t^{i*}(S_2))^2} \geq 0.
\end{aligned}
$$

3) $D_t^{x*}(S_1 \cup \{x\}) = D_B^x$ and $D_t^{x*}(S_2 \cup \{x\}) = 0$:

Since $D_t^{x*}(S_2 \cup \{x\}) = 0$, we have $-\mathcal{J}(S_2 \cup \{x\}) + \mathcal{J}(S_2) = -(1-\gamma)c_{m,t}^x$. Thus, we have

$$
\begin{aligned}
&- \mathcal{J}(S_1 \cup \{x\}) + \mathcal{J}(S_1) - (-\mathcal{J}(S_2 \cup \{x\}) + \mathcal{J}(S_2)) \\
&= \frac{\gamma L a_t \eta^2 \sigma^2}{2} \left[\frac{D_B^x}{\sum\limits_{i \in S_t} D_t^{i*}(S_t) \sum\limits_{i \in S_t \cup \{x\}} D_t^{i*}(S_t \cup \{x\})} \right] - (1-\gamma) \left(c_{p,t}^x D_B^x \right) \\
&\geq \frac{\gamma L a_t \eta^2 \sigma^2}{2} \left[\frac{\eta\sigma \sqrt{\frac{L\gamma a_t}{2(1-\gamma)c_{p,t}^i}} - \sum\limits_{i \in S_1} D_t^{i*}(S_1)}{\sum\limits_{i \in S_t} D_t^{i*}(S_t) \sum\limits_{i \in S_t \cup \{x\}} D_t^{i*}(S_t \cup \{x\})} \right] \\
&- (1-\gamma) \left[c_{p,t}^x (\eta\sigma \sqrt{\frac{L\gamma a_t}{2(1-\gamma)c_{p,t}^i}} - \sum_{i \in S_1} D_t^{i*}(S_1)) \right] = 0.
\end{aligned}
$$

4) $D_t^{x*}(S_1 \cup \{x\}) = \eta\sigma \sqrt{\frac{L\gamma a_t}{2(1-\gamma)c_{p,t}^i}} - \sum_{i \in S_1} D_t^{i*}(S_1)$ and $D_t^{x*}(S_2 \cup \{x\}) = 0$:

We omit the proof of this combination since it is similar with that of combination 2).

5.8.5 Proof of Theorem 5.3

We have for any round t,

$$\mathcal{G}(S_t) \geq \frac{1}{3}\mathcal{G}(\mathcal{OPT}_t),$$

where $\mathcal{OPT}_t$ is the optimal user set in round t.

Then, we have for any round t,

$$\mathcal{J}_1(S_t, D_t^*(S_t)) + \mathcal{J}_{1,\max} \geq \frac{1}{3}(\mathcal{J}_1(\mathcal{OPT}_t, D_t^*(\mathcal{OPT}_t)) + \mathcal{J}_{1,\max}).$$

Thus, the system loss over T rounds is bounded by

$$
\begin{aligned}
&\sum_{t=1}^{T} \mathcal{J}_1(S_t, D_t^*(S_t)) \geq \sum_{t=1}^{T} (\frac{1}{3}\mathcal{J}_1(\mathcal{OPT}_t, D_t^*(\mathcal{OPT}_t)) + \frac{2}{3}\mathcal{J}_{1,\max}) \\
&\geq \frac{1}{3}\mathcal{OPT} + \mathcal{O}(T).
\end{aligned}
$$

Chapter 6

Quality-Aware Distributed Computation for Cost-Effective Non-Convex and Asynchronous Wireless Federated Learning.

6.1 Introduction

One significant advantage of using FL is to preserve the privacy of individual users' data. Moreover, since only the local ML model parameters, instead of the local data, are sent to the server, the communication costs can be greatly reduced. Furthermore, FL can exploit ubiquitous smart devices with substantial computing capabilities, which are often under-utilized. In particular, when FL is used in a wireless edge network, the data samples generated at individual wireless devices can be exploited via local computation and global aggregation based on distributed ML. As a result, *wireless federated learning* (WFL) can achieve collaborative intelligence in wireless edge networks. A general consensus is that WFL can support intelligent control and management of wireless communications and networks (such as in [7, 8, 9]), and can enable many AI applications based on wireless networked systems.

As is standard, learning accuracy is a key performance metric for FL. The accuracy of the trained machine learning model in FL depends heavily on which users participate in the learning process and the *quality* of their local model updates. Specifically, when distributed stochastic gradient descent (SGD) is used for FL, the quality of a local stochastic gradient in each iteration can be measured by the variance of the gradient, which depends on the *mini-batch size* used to compute the gradient. It is important to observe that the quality of local updates can be treated as a design parameter and used as a *control "knob"* (via the mini-batch size) to be adapted across users and over time. Such quality-aware distributed computation can substantially improve the learning accuracy of WFL.

In this chapter, we study quality-aware distributed computation for WFL, with the focuses on *non-convex* problems and *asynchronous* algorithms. The training problem of many practical ML models (e.g., deep neural networks) involves a non-convex loss function. Such a non-convex optimization problem is more difficult to solve than the convex version, due to suboptimal local minima. In addition, asynchronous learning algorithms are usually more efficient than their synchronous counterparts in utilizing users' computing capabilities, as it allows devices to keep computing without waiting for the global update received from the server as in the synchronous algorithms. This benefit of asynchronous learning is more so in a wireless setting, as there can be strong communication stragglers due to heterogeneous and time-varying wireless channels.

Our goal is to minimize the training loss while taking into account costs and constraints of computation and communication resources. In particular, we investigate how to adaptively determine participating users' mini-batch sizes over the learning process. To this end, several significant challenges need to be addressed: 1) The quality (determined by the mini-batch size) of local stochastic gradient updates can be heterogeneous across users and time-varying, and it is non-trivial to quantify the impacts of local updates' quality on the accuracy of the final learnt model over the learning process. 2) The non-convex and asynchronous settings of FL require new analysis different from their convex and synchronous counterparts. 3) The unique features of wireless edge networks, including time-varying wireless channels, should be taken into account. To achieve a desired tradeoff between the training loss and the training cost, local updates' quality should be determined based on the impacts of local updates on the training loss as well as users' wireless channel conditions and computation costs.

The main contributions are summarized as follows:

- We propose quality-aware distribute computation for FL in wireless edge networks, which controls the *quality* of users' local model updates via the mini-batch sizes used to compute the updates, for non-convex problems and asynchronous algorithms. Our goal is to minimize the training loss as well as users' computation and communication costs in the training process.

- We characterize performance bounds on the training loss as a function of users' local updates' quality (and thus the mini-batch sizes) over the training process, for both non-convex and asynchronous settings. Our findings reveal that the impact of a user's local update's quality on the training loss 1) increases with the stepsize used that local update for non-convex learning, and 2) increases when there are more other users' local updates which are coupled with that local update for asynchronous learning, depending on the update delays.

- Based on the obtained insights above, we develop channel-aware adaptive algorithms that determine users mini-batch sizes over the training process for both non-convex and asynchronous learning, based on the impacts of local updates' quality on the training loss as well as users' wireless channel conditions (which determine the update delays) and computation costs. We characterize the optimal mini-batch sizes, which shows that it is optimal to use larger mini-batch sizes when the local updates' impacts are larger. For the non-convex setting, we also develop a greedy algorithm that selects participating users, which achieve an approximation ratio by exploiting the non-monotone submodular property of the problem.

- We evaluate the proposed quality-aware adaptive algorithms using simulations. The results demonstrate that these algorithms outperform existing schemes in terms of the training loss.

The remainder of this chapter is organized as follows. Section 6.2 reviews related work. In Section 6.3, we describe quality-aware distributed computation for federated learning. In Section 6.4 and Section 6.5, we study learning accuracy bounds, and dynamic user selection and mini-batch size design based on the training loss bounds, respectively. Simulation results are provided in Section 6.6.

6.2 Related Work

Wireless Federated Learning. FL has emerged as a disruptive computing paradigm for ML by democratizing the learning process to potentially many individual users using their end

devices. For WFL, the computing and networking environments have salient features, including heterogeneous and time-varying computing and communication resources that need to be accounted for. Recent studies on FL have made effort to take into account these issues [95, 97, 100, 111, 98, 99, 101, 102]. For example, Tran et al [95] studied FL in wireless networks for devices with different computing and communication capabilities. In [103], Tu et al studied computation offloading based distributed learning where devices have different computing and communication resources. However, all these works have not exploited mini-batch sizes to *control the quality* of users' local model updates, and have not considered the impacts of diverse and dynamic local updates' quality on learning accuracy. A very recent work [112] has studied quality-aware distributed computation for WFL with convex problems and synchronous algorithms. However, it has not considered the non-convex and asynchronous settings which are very different and is the focus of this chapter.

Non-Convex and Asynchronous Distributed Learning. With rapid recent advances in ML/AI, distributed ML has also seen substantial research activities in the past decade [81, 83, 14]. Many prior works have studied various settings of distributed ML [113, 114, 115, 116, 117, 118], including for non-convex problems and asynchronous algorithms. As many ML optimization problems are non-convex, the convergence of non-convex distributed learning has been studied [114, 115, 116]. Along a different avenue, asynchronous distributed learning [117, 118, 119, 120, 121] has received more attention due to its high efficiency for large-scale distributed learning. However, most existing works on non-convex and asynchronous distributed learning have focused on the impacts of learning rate and delay on the convergence of the learning algorithm. In this chapter, we theoretically study the impacts of local updates' quality (quantified by the variance and determined by the mini-batch size) on learning accuracy, and the mini-batch size design, which is very different from the prior works.

6.3 Quality-Aware Distributed Computation for Wireless Federated Learning

In this section, we present the system model of quality-aware distributed computation for FL in a wireless edge network.

Quality-Aware Distributed Computation for FL. Consider the setting where the distributed learning process of FL is carried out by a set of wireless users. The server of FL can reside in the cloud or at the edge (e.g., access point or base station of a wireless network), and the users are connected to the FL server via wireless links.

We consider the following FL problem:

$$\min_{\mathbf{w}} \; F(\mathbf{w}) \triangleq \sum_{i=1}^{N} \frac{D_i}{D} F_i(\mathbf{w}), \tag{6.1}$$

where $F_i(\mathbf{w})$ is the prediction loss of the model parameter $\mathbf{w}$ based on user i's local dataset, N is the number of users, $\mathcal{D}_i = \{\xi_1^i, \xi_2^i, \ldots, \xi_{D_i}^i\}$ is user i's local dataset for updating the model parameter, and $D \triangleq \sum_{i=1}^{N} D_i$. User i's local loss function $F_i(\mathbf{w})$ is defined by

$$F_i(\mathbf{w}) \triangleq \frac{1}{D_i} \sum_{m=1}^{D_i} f_i(\mathbf{w}; \xi_m^i),$$

where $f(\cdot)$ is the per-sample loss function. In each round of FL, K out of N users are selected from the user set $\mathcal{N}$ to compute local updates, communicate their local updates to the server, and receive the updated global model from the server. At round t, a selected user i computes the average gradient g_{t-1}^i of the loss function using a set of D_t^i data samples randomly drawn from her local dataset $\mathcal{D}_i$, based on the global model $\mathbf{w}_{t-1}$ received from the previous round $t-1$, and update her local model as

$$\mathbf{w}_t^i = \mathbf{w}_{t-1} - \eta g_t^i,$$

where

$$g_t^i \triangleq \frac{1}{D_t^i} \sum_{j=1}^{D_t^i} \nabla f(\mathbf{w}, \xi_t^{i,j}),$$

η is the stepsize, and $\xi_t^{i,j}$ is the jth data sample randomly drawn from user i's local dataset $\mathcal{D}_i$. At the end of round t, the server aggregates K users' local models and updates the global

model as

$$\mathbf{w}_t = \sum_{i=1}^{K} \frac{D_t^i}{D_t} \mathbf{w}_t^i, \tag{6.2}$$

where $D_t \triangleq \sum_{i=1}^{K} D_t^i$.

The *quality* of a user's local update is captured by the variance of the local stochastic gradient, given by

$$q_i \triangleq E\left[\left\|g_t^i - \bar{g}_t\right\|^2\right], \tag{6.3}$$

where $\bar{g}_t \triangleq E[g_t^i]$. Assume that the loss function f satisfies $E\left\|\nabla f_i\left(\mathbf{w}_t, \xi_m^i\right) - E[\nabla f_i\left(\mathbf{w}_t\right)]\right\|^2 \leq \sigma^2$, $\forall t$. It can be shown that [105]

$$E\left[\left\|g_t^i - \bar{g}_t\right\|^2\right] \leq \frac{\sigma^2}{D_t^i}.$$

Note that a user's quality is determined by the *mini-batch size* D_t^i used to update her local model. Thus, a local update computed with a larger mini-batch size has higher quality.

In this chapter, we assume that users' local data are IID. Our results in the following sections (including the training loss bounds, adaptive mini-batch size design and user selection) can be extended to the case of non-IID data, and will be studied in our future work.

Asynchronous FL. The Fl algorithm can be carried out in an asynchronous manner described as follows. In this case, the learning process consists of rounds, each lasting for a time period of the same length. At the beginning of each round, the server broadcasts the global model. At the end of each round, the server updates the global model using the local models received in the round as (6.2) (as illustrated in Fig. 6.1). Note that a user's local update can be received by the server in a round different from the round when the user receives the global model from the server for computing that local update. The update delay τ_i quantifies the difference between the round when user i receives the global model from the server and the round when user i's local update computed from that global model is received by the server. Note that τ_i is an integer, where $\tau_i = 1$ means there is no update delay, and $\tau_i > 1$ means there is an update delay. The computation time (Ci) is the time it takes for user i to *compute* (update) her local model, and the communication time (Mi) is the time it takes for user i to *communicate*

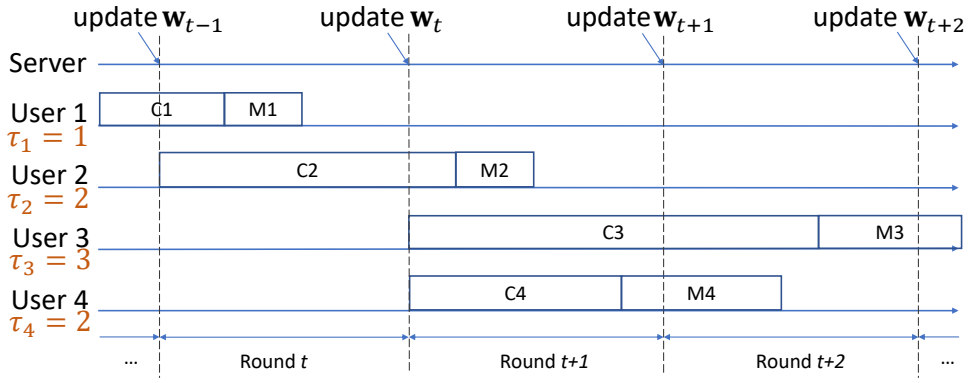

Figure 6.1: Schedule of the server's updates and users' computations (C) and communications (M) in asynchronous FL.

(upload) her updated local model to the server. For example, in Fig. 6.1, user 2 receives the global model and starts to update her local model at the beginning of round t. After computing the local model, user 2 starts to uploads her local model to the server. Then the server receives user 2's local model $\mathbf{w}_{t+1}^2$ in round $t+1$ and updates the global model as $\mathbf{w}_{t+1} = \mathbf{w}_{t+1}^2$ at the end of round $t+1$.

FL in Wireless Edge Network. A user incurs a computation cost (measured by the computation delay, energy consumption, etc) for computing a local update, which depends on the computation capability of the user's device and the mini-batch size used to compute the update. Let $c_{p,t}^i$ be user i's cost of computing her local update using one data sample in round t. Besides the computation cost, a user also incurs communication cost for communicating local updates to the server (measured by the communication delay, energy consumption, etc), which depends on the user's wireless channel condition. Let $c_{m,t}^i$ be user i's communication cost in round t. Note that the computation cost $c_{p,t}^i$ and the computation cost $c_{m,t}^i$ generally vary across users and over rounds of the FL algorithm.

6.4 Learning Accuracy Bound Analysis

In this section, under the quality-aware FL framework of the previous section, we study the training loss bounds for three settings: 1) non-convex problems; 2) asynchronous algorithms; 3) non-convex problems and asynchronous algorithms. We will first characterize the performance bounds as functions of users' mini-batch sizes over the training process. Based on the obtained

results, we then discuss the impacts of mini-batch sizes and other system parameters (including stepsize) on the training loss.

6.4.1 Case of Non-Convex Learning

We first analyze the training loss bound for non-convex problems with synchronous algorithms. For non-convex optimization, the metrics for convex optimization (e.g., $F(\mathbf{w}_T) - F(\mathbf{w}^*)$) are not suitable, since it is hard to find the global optimum for non-convex optimization problems. Thus, we analyze the ergodic convergence [115], where we randomly select an index $\tilde{t}$ from $\{1, 2, \ldots, T\}$ with probability $\{\eta_t / \sum_{t=1}^{T} \eta_t\}$, and use $\mathbf{w}_{\tilde{t}}$ as the final model of the training process. The expected squared gradient norm $\|\bar{g}_{\tilde{t}}\|^2$ of $\mathbf{w}_{\tilde{t}}$ can be upper bounded as follows.

Theorem 6.1 Suppose F is L-smooth, and $E\left\|\nabla f_i\left(\mathbf{w}_t, \xi_m^i\right) - E[\nabla f_i\left(\mathbf{w}_t\right)]\right\|^2 \leq \sigma^2, \forall i, t$, take the stepsize $\eta_t \leq \frac{1}{L}, \forall t$, the ergodic convergence is given by

$$\frac{1}{\sum_{t=1}^{T} \eta_t} \sum_{t=1}^{T} \eta_t \|\bar{g}_t\|^2 \leq \frac{2E(F(\mathbf{w}_0) - F(\mathbf{w}^*)) + \sum_{t=1}^{T} L\eta_t^2 \frac{\sigma^2}{D_t}}{\sum_{t=1}^{T} \eta_t}, \tag{6.4}$$

where $D_t \triangleq \sum_{i \in S_t} D_t^i$, and D_t^i is the mini-batch size of user i in round t.

Remark 6.1 In (6.4), $\frac{\sigma^2}{D_t}$ is the upper bound of the variance of the global model update $\mathbf{w}_t$, which is determined by the variances of participating users' local model updates. Thus the weight $L\eta_t^2 / \sum_{t=1}^{T} \eta_t$ of the variance $\frac{\sigma^2}{D_t}$ of the global update captures the impact of the quality of local updates on the training loss. We also know that a larger stepsize results in a larger change of the model. Thus, given the total number of rounds T and the sum of stepsizes $\sum_{t=1}^{T} \eta_t$, the larger the stepsize of the round, the larger the impact of local updates' quality on the training loss. Also observe that with any sequence of $\{\eta_t\}$ such that $\sum_{t=1}^{T} \eta_t$ diverge and $\sum_{t=1}^{T} \eta_t^2$ converge (e.g., $\eta_t = 1/t$), and with any non-decreasing mini-batch size, the bound converges to 0 as $T \to \infty$.

Taking a closer look at Theorem 6.1, we can properly choose the stepsize and total mini-batch size in each round and obtain the following convergence rate:

Proposition 6.1 Suppose F is L-smooth, $E\left\|\nabla f_i\left(\mathbf{w}_t, \xi_m^i\right) - E[\nabla f_i\left(\mathbf{w}_t\right)]\right\|^2 \leq \sigma^2$, and $E\left\|\nabla F_i\left(\mathbf{w}_t\right)\right\|^2 \leq B^2$, $\forall i, t$, take any mini-batch size $\{D_t\}$ and a constant stepsize $\eta = \frac{\sqrt{D_{\min}}}{L\sqrt{T}}$, where $D_{\min} \triangleq \min\{D_t\}$, the ergodic convergence is given by

$$\frac{1}{T}\sum_{t=1}^{T}\|\bar{g}_t\|^2 \leq \frac{2LE(F(\mathbf{w}_0) - F(\mathbf{w}^*)) + \sigma^2}{\sqrt{D_{\min}T}}. \tag{6.5}$$

Proposition 6.1 shows that when the stepsize is small enough, the convergence rate achieves $\mathcal{O}(1/\sqrt{D_{\min}T})$, which is consistent with the result obtained in [114], Corollary 1.

6.4.2 Case of Asynchronous Learning

We then analyze the training loss bound when asynchronous algorithms are used for convex problems. In this subsection, for ease of exposition, we focus on using a constant stepsize. Our results can be extended to using a time-varying stepsize.

Theorem 6.2 Suppose F is L-smooth and μ-strongly convex, $E\left\|\nabla f_i\left(\mathbf{w}_t, \xi_m^i\right) - E[\nabla f_i\left(\mathbf{w}_t\right)]\right\|^2 \leq \sigma^2$, and $E\left\|\nabla F_i\left(\mathbf{w}_t\right)\right\|^2 \leq B^2$, $\forall i, t$, take the stepsize $\eta \leq \frac{1}{L}$, then the training loss is bounded by

$$\begin{aligned} E[F(\mathbf{w}_T) - F(\mathbf{w}^*)] &\leq (1-\mu\eta)^T(F(\mathbf{w}_0) - F(\mathbf{w}^*)) \\ &+ \sum_{t=1}^{T}\left[(1-\mu\eta)^{T-t}\left(L^2\eta^3\sum_{i\in M_t}\frac{D_t^i}{D_t}\Big(\sum_{t'=t-\tau_i+1}^{t-1}\frac{\sigma^2}{D_{t'}}\right.\right. \\ &\left.\left.+\Gamma(\tau_i-1)B^2\Big) + \frac{\eta}{2}\frac{\sigma^2}{D_t}\right)\right], \end{aligned} \tag{6.6}$$

where M_t is the set of users who update at time t, Γ is the maximum update delay.

Remark 6.2 Theorem 6.2 shows that the training loss bound consists of two terms. The first term decreases geometrically with the number of rounds T, and is due to that SGD in expectation makes progress towards the optimal solution. The second term of the bound is caused by the randomness of data sampling for computing the update in SGD. Compared to the training loss in the case of synchronous learning [112], each term in the second term not only depends on the total mini-batch size D_t of users who finish uploading their local models in the current round

t, but also the total mini-batch size $D_{t'}$ of each past round t' such that $t' \in \{t-\tau_i+1, \ldots, t-1\}$, where $i \in M_t$. For example, in Fig. 6.1, the training loss in round $t+1$ not only depends on user 2's mini-batch size, but also on user 1's mini-batch size. This implies that in each round, the larger update delays of users, the worse the training loss.

Remark 6.3 According to (6.6), the training loss due to users' update delays in each round is $\sum_{i \in M_t} \frac{D_t^i}{D_t} (\sum_{t'=t-\tau_i+1}^{t-1} \frac{\sigma^2}{D_{t'}} + \Gamma(\tau_i - 1)B^2)$. When there is no update delay, i.e., $\tau_i = 1, \forall i \in M_t$, this term is 0, and the training loss bound degenerates to the case of synchronous learning in [112]. When update delays exist, this term decreases as the mini-batch sizes of users who upload their local models in the past rounds increase.

To obtain some useful insights, we next focus on the special case where only one user uploads her local model in a round. Let user t be the user who uploads her local model in round t. Then, using Theorem 6.2, the training loss bound is given by

$$\begin{aligned} &E[F(\mathbf{w}_T) - F(\mathbf{w}^*)] \leq (1-\mu\eta)^T (F(\mathbf{w}_0) - F(\mathbf{w}^*)) + \\ &\sum_{t=1}^{T} \left[(1-\mu\eta)^{T-t} \left(L^2\eta^3 (\sum_{t'=t-\tau_t+1}^{t-1} \frac{\sigma^2}{D_{t'}} + \Gamma(\tau_t - 1)B^2) + \frac{\eta}{2} \frac{\sigma^2}{D_t} \right) \right]. \end{aligned} \tag{6.7}$$

To analyze the impact of user t on the training loss, we find the terms determined by D_t and τ_t in (6.7) as follows.

$$\begin{aligned} I_t = &(1-\mu\eta)^{T-t} \left(L^2\eta^3 (\sum_{t'=t-\tau_t+1}^{t-1} \frac{\sigma^2}{D_{t'}} + \Gamma(\tau_t - 1)B^2) \right. \\ &\left. + \frac{\sigma^2}{D_t} \left(\frac{\eta}{2} + L^2\eta^3 \sum_{\tau'=1}^{\Gamma} \mathbf{1}_{\tau_{t+\tau'} \geq \tau'} (1-\mu\eta)^{-\tau'} \right) \right), \end{aligned} \tag{6.8}$$

where $\mathbf{1}$ is an indicator function such that $\mathbf{1} = 1$ when the user who uploads her local model in round $t+\tau'$ receives the global model from the server before round t, and $\mathbf{1} = 0$ otherwise.

Remark 6.4 From (6.8), we can see that when user t's update delay τ_t is independent of her mini-batch size D_t, I_t decreases as D_t increases and/or τ_t decreases. This implies that in a given round, a user with a larger mini-batch size or a smaller update delay reduces the training

loss. Also observe that user t's mini-batch size affects some later rounds such that users who uploads their local models in those rounds receive the global models before round t.

Remark 6.5 In (6.8), the weight of user t's impact on the training loss is $(1-\mu\eta)^{T-t}$. Note that this weight increases with the round number t as $1-\mu\eta<1$. Therefore, the update delay and mini-batch size in a later round have larger impacts on the training loss than in an earlier round. This observation has important implications: it is better for a user to have a larger mini-batch size or a smaller update delay in a later round rather than an earlier round to reduce the training loss.

6.4.3 Case of Non-Convex and Asynchronous Learning

Next, we analyze the training loss bound for non-convex problems and asynchronous algorithms.

Theorem 6.3 Suppose F is L-smooth and $E\left\|\nabla f_i\left(\mathbf{w}_t,\xi_m^i\right)-E[\nabla f_i\left(\mathbf{w}_t\right)]\right\|^2\leq\sigma^2$, $\forall i,t$, take the stepsize $\eta_t\leq\frac{1}{L}$, $\forall t$, the ergodic convergence is given by

$$\begin{aligned}\frac{1}{\sum_{t=1}^T\eta_t}\sum_{t=1}^T\eta_t\|\bar{g}_t\|^2\leq&\frac{2E(F(\mathbf{w}_0)-F(\mathbf{w}^*))}{\sum_{t=1}^T\eta_t}\\&+\frac{\sum_{t=1}^T(L\eta_t^2\frac{\sigma^2}{D_t}+2L^2\eta_t\sum_{i\in M_t}\frac{D_t^i}{D_t}\sum_{t'=t-\tau_i+1}^{t-1}\eta_{t'}^2\frac{\sigma^2}{D_{t'}})}{\sum_{t=1}^T\eta_t},\end{aligned}\tag{6.9}$$

where the stepsize satisfies $L\eta_t+L^2\Gamma\eta_t\sum_{t'=1}^{\Gamma-1}\eta_{t+t'}\leq 1$, $\forall t$.

Remark 6.6 From (6.9), we can see that the convergence rate for the combined non-convex and asynchronous setting shows similar properties as that for each of the two settings. First, the convergence rate not only depends on the total mini-batch size D_t used in the current round t, but also on the mini-batch sizes used in several past rounds. Second, given the total number of rounds T and the sum of stepsizes $\sum_{t=1}^T\eta_t$, the larger the stepsize of the round, the larger the impact of local updates' quality on the training loss. Also observe that the impact of the quality $\frac{\sigma^2}{D_t}$ of the local updates in round t is captured not only by the stepsize η_t in round t but also by the stepsizes in several later rounds.

6.5 Dynamic Cost-Effective User and Sampling Size Selection

In this section, we study how to design users' mini-batch sizes over the training process to minimize the training loss bound for non-convex and asynchronous FL, respectively. For the non-convex setting, we also investigate how to select participating users. In the meanwhile, we take into account users' computation and communication costs. Note that we consider continuous-valued mini-batch size D_t in our theoretical analysis, which can be converted back to the nearest integer values when used in practice.

6.5.1 Case of Non-Convex Learning

First we study the optimal user selection and mini-batch size design for non-convex problems with synchronous algorithms. We aim to minimize the sum of the training loss and users' total communication and computation cost. The optimization problem can be formulated as

$$\begin{aligned}\min_{\{S_t\},\{D_t\}} \ & \gamma \frac{1}{\sum_{t=1}^{T}\eta_t} \sum_{t=1}^{T} \eta_t \|\bar{g}_t\|^2 + (1-\gamma) \sum_{t=1}^{T} \sum_{i\in S_t} (c_{p,t}^i D_t^i + c_{m,t}^i), \\ s.t. \ & D_t^i \leq D_B^i, \forall i, t,\end{aligned}$$

where $\{S_t\}$ is the set of selected users in all rounds, $\{D_t\}$ is the set of assigned mini-batch sizes of users in all rounds, D_t is the mini-batch size of the user who updates her local model in round t, $\gamma \in (0, 1]$ is the weight that balances the training loss and the cost, which can be determined according to the server's concern, and D_B^i is user i's maximum possible mini-batch size.

From (6.4), we rewrite the problem as follows:

$$\begin{aligned}\min_{\{S_t\},\{D_t\}} \ & \sum_{t=1}^{T} \mathcal{J}_1(S_t, D_t) = \gamma \sum_{t=1}^{T} \frac{L\eta_t^2 \frac{\sigma^2}{D_t}}{\sum_{t=1}^{T}\eta_t} + (1-\gamma) \sum_{t=1}^{T} \sum_{i\in S_t} (c_{p,t}^i D_t^i + c_{m,t}^i), \\ s.t. \ & D_t^i \leq D_B^i, \forall i, t,\end{aligned}$$

Since $\sum_{t=1}^{T}\eta_t$ is given, we can see that the problem above can be decomposed into T independent problems, each for one of the T rounds. Thus, we focus on finding the optimal

S_t and D_t for a single round t. Moreover, we decompose the problem in round t into two subproblems: 1) we first study the optimal mini-batch size design given any user selection; 2) then we study the optimal user selection given the optimal mini-batch size design.

Optimal Mini-Batch Size Design. Given any set of selected users, since the total communication cost is fixed, a user with a lower computation cost is preferred over one with a higher computation cost. Hence, the optimal mini-batch sizes are determined in the ascending order of users' computation costs until the minimum value of $\mathcal{J}_1$ is reached. It can be shown that $\mathcal{J}_1$ is a convex function of D_t^i when other users' mini-batch sizes are given. The following result characterizes users' optimal mini-batch sizes.

Theorem 6.4 Let users in S_t be ordered as $c_{p,t}^1 \leq c_{p,t}^2 \leq \cdots \leq c_{p,t}^{|S_t|}$. Then the users' optimal mini-batch sizes are determined iteratively in this order, where the ith user's optimal mini-batch size is given by

$$D_t^{i*}(S_t) = \min\{D_B^i, \max\{\sqrt{\frac{L\gamma\eta_t^2\sigma^2}{(1-\gamma)c_{p,t}^i \sum_{t-1}^{T} \eta_t}} - \sum_{j=1}^{i-1} D_t^{j*}, 0\}\}.$$

User Selection Algorithm With the optimal mini-batch size design, the user selection problem in round t can be rewritten as

$$\min_{S_t} \ \mathcal{J}_1(S_t, D_t^*(S_t)) = \gamma \left(\frac{L\eta_t^2\sigma^2}{\sum_{i\in S_t} D_t^{i*}(S_t) \sum_{t=1}^{T} \eta_t} \right) + (1-\gamma) \sum_{i\in S_t} (c_{p,t}^i D_t^{i*}(S_t) + c_{m,t}^i). \tag{6.10}$$

First note that problem (6.10) can be cast as the maximization problem of $-\mathcal{J}_1(S_t, D_t^*(S_t))$. Then we have the following property of the problem.

Lemma 6.1 $-\mathcal{J}_1(S_t, D_t^*(S_t))$ is a negative non-monotone submodular function.

Note that it is difficult to solve a negative non-monotone submodular maximization problem with performance guarantee (e.g., with an approximation ratio). To overcome this challenge, we transform the above problem into a non-negative non-monotone submodular maximization problem.

First, we find the maximum value of $\mathcal{J}_1(S_t, D_t^*(S_t))$. From the optimal mini-batch size design, we can see that there exists some user j, such that for any user $j' \in S_t$ that has $c_{p,t}^{j'} \geq c_{p,t}^{j}$, we have $D_t^{j'*}(S_t) = 0$. The objective function $\mathcal{J}_1$ decreases as users' optimal mini-batch sizes are determined iteratively according to Proposition 6.4 until user j, and does not change when the optimal mini-batch sizes of users after j are determined. Therefore, given a selected user set S_t, $\mathcal{J}_1$ is maximized when only one user's mini-batch size is non-zero. Thus, for any $S \subseteq \mathcal{N}$, the maximum value of $\mathcal{J}_1$ is given by

$$\mathcal{J}_{1,\max} = \frac{\gamma L \eta_{\max}^2 \sigma^2}{\sum_{t=1}^{T} \eta_t} + (1-\gamma)(c_{p,\max} D_{B,\max} + N c_{m,\max}),$$

where $\eta_{\max} = \max\{\eta_t | \forall t\}$, $c_{p,\max} = \max\{c_{p,t}^i | \forall i, t\}$, $D_{B,\max} = \max\{D_B^i | \forall i\}$, and $c_{m,\max} = \max\{c_{m,t}^i | \forall i, t\}$.

Based on the maximum value of $\mathcal{J}_1(S_t, D_t^*(S_t))$, we define a new function $\mathcal{G}(S_t)$ and rewrite the user selection problem in round t as

$$\max_{S_t} \mathcal{G}(S_t) \triangleq \begin{cases} -\mathcal{J}_1(S_t, D_t^*(S_t)) + \mathcal{J}_{1,\max}, & \text{if } S_t \neq \emptyset \\ 0, & \text{if } S_t = \emptyset \end{cases}$$

Note that in each round t of the FL algorithm, at least one user is selected (i.e., $|S_t| > 0$) with a positive mini-batch size (i.e., $D_t > 0$). This is because when $S_t = \emptyset$, the global model is not updated so that round t should not be counted as a round of the FL algorithm. Thus, we can define that $\mathcal{G}(\emptyset) \triangleq 0$. We can see that the above two problems are equivalent since $\mathcal{J}_{1,\max}$ is a constant. Next, we focus on finding the solution that maximizes $\mathcal{G}(S_t)$. From Lemma 6.1, it follows directly that $\mathcal{G}(S_t)$ is a non-negative non-monotone submodular function.

Next, we can apply the DeterministicUSM Algorithm [109] to solve the user selection problem, which is given in Algorithm 9. The approximation ratio of DeterministicUSM is given by the following lemma.

Lemma 6.2 [109] Algorithm DeterministicUSM is a $\frac{1}{3}$-approximation algorithm for maximizing function $\mathcal{G}(S_t)$.

Algorithm 9: Approximate optimal user selection

$X_0 \leftarrow \emptyset, Y_0 \leftarrow \mathcal{N}$;
for $i = 1, 2, \cdots, N$ **do**
  $a_i \leftarrow \mathcal{G}\left(X_{i-1} \cup \{i\}\right) - \mathcal{G}\left(X_{i-1}\right)$;
  $b_i \leftarrow \mathcal{G}\left(Y_{i-1} \setminus \{i\}\right) - \mathcal{G}\left(Y_{i-1}\right)$;
  if $a_i \geq b_i$ **then**
    $X_i \leftarrow X_{i-1} \cup \{i\}, Y_i \leftarrow Y_{i-1}$;
    else
      $X_i \leftarrow X_{i-1}, Y_i \leftarrow Y_{i-1} \setminus \{i\}$;
$S_t \leftarrow X_N$;
return S_t.

Given the result above, we then show the approximation ratio of Algorithm 9 for our problem, under the optimal mini-batch size design given in Proposition 6.4. The proof of the following result is given in the appendix.

Theorem 6.5 Under the optimal mini-batch size design, with the user selection given by Algorithm 9, the system loss is upper bounded by

$$\sum_{t=1}^{T} \mathcal{J}_1(S_t^*, D_t^*(S^*(t))) \leq \frac{1}{3}\mathcal{OPT} + \mathcal{O}(T),$$

where $\mathcal{OPT}$ is the system loss of the optimal user and mini-batch size selection.

6.5.2 Case of Asynchronous Learning

We next study the case of asynchronous algorithms for convex problems. In this chapter, we assume that the update delays[1] of users are fixed, regardless of the mini-batch sizes used to compute the local updates. This is a reasonable assumption when the communication times of local updates are relatively large compared to their computation times. The case where update delays depend on the mini-batch sizes is a very challenging problem, and will be studied in our future work. For ease of exposition, in this subsection, we also assume that only one user uploads her local model in one round.

[1]Note that the update delay is an integer and is not the total computation and communication time of the local update.

As discussed earlier, the impact of each user t's mini-batch size on the training loss depends on the set of other users' computation and communication periods during which user t uploads her local model. We need to take into account this coupling of users' updates in terms of their impacts on the training loss, based on the update delays over the training process. Thus, we develop an adaptive algorithm that determines the mini-batch size for each user's each update, based on the delay information of only Γ number of updates in the future.

When users' update schedules are given[2], the total communication cost is fixed. Thus it suffices to minimize the sum of the training loss bound and users' total computation cost:

$$\begin{aligned} &\min_{\{D_t\}} \gamma E[F(\mathbf{w}_T) - F(\mathbf{w}^*)] + (1-\gamma)\sum_{t=1}^{T} c_{p,t} D_t, \\ &s.t.\ D_t \leq D_B^t, \forall t, \end{aligned} \tag{6.11}$$

where D_B^t is user t's maximum possible mini-batch size. Since the first term of the training loss bound given in (6.7) does not depend on users' mini-batch sizes $\{D_t\}$, it suffices to minimize the second term. Furthermore, we rewrite the second term of (6.6) as the sum of terms that are determined by user t's update delay and mini-batch size. Thus, from (6.7), we rewrite (6.11) as

$$\begin{aligned} \min_{\{D_t\}} \sum_{t=1}^{T} \mathcal{J}_2(D_t) = &\sum_{t=1}^{T} \Big(\gamma(1-\mu\eta)^{T-t} \Big(L^2\eta^3\Gamma(\tau_t - 1)B^2 \\ &+ \frac{\eta}{2}\frac{\sigma^2}{D_t} + L^2\eta^3\frac{\sigma^2}{D_t}\sum_{\tau'=1}^{\Gamma} \mathbf{1}_{\tau_{t+\tau'} \geq \tau'}(1-\mu\eta)^{-\tau'}\Big) \\ &+(1-\gamma)c_{p,t}D_t\Big), \\ s.t.\ D_t \leq D_B^t, \forall t. \end{aligned} \tag{6.12}$$

where $\tau_{t+\tau'}$, $\forall \tau' \in \{1, \ldots, \Gamma\}$ are known. Thus, for user t who uploads her local model in round t, the server can find the value of $\sum_{\tau'=1}^{\Gamma} \mathbf{1}_{\tau_{t+\tau'} \geq \tau'}(1-\mu\eta)^{-\tau'}$ based on the update delays of users who upload their local models in next Γ rounds.

[2]Note that the training loss depends heavily on users' update schedules. We focus on the design of users' mini-batch size given users' update schedules here and will study the joint design of user selection and mini-batch size in future work.

Note that problem (6.12) can be decomposed into T independent problems, each for one of the T users. Thus, we focus on finding the optimal D_t that minimizes $\mathcal{J}_2(D_t)$ for a single user t. It can be shown that $\mathcal{J}_2(D_t)$ is a convex function of D_t. Thus the optimal mini-batch size D_t^* can be found as the local minimum of $\mathcal{J}_2(D_t)$, given as below.

Theorem 6.6 Given users' update schedules and their update delays, the optimal mini-batch size for each user t who uploads her local model in round t is given by

$$D_t^* = \min\{\sqrt{\frac{\gamma v_t \sigma^2}{(1-\gamma)c_{p,t}}}, D_B^t\},$$

where $v_t = (1-\mu\eta)^{T-t}(\frac{\eta}{2} + L^2\eta^3 \sum_{\tau'=1}^{\Gamma} \mathbf{1}_{\tau_{t+\tau'} \geq \tau'}(1-\mu\eta)^{-\tau'})$.

Theorem 6.6 shows that the optimal mini-batch size D_t^* is larger in a later round. This is because the weight $(1-\mu\eta)^{T-t}$ of an update on the training loss bound increases with the round number t, so that D_t^* also increases with the round number. Also note that D_t^* increases as the number of users who receive the global models before round t and finish uploading their local models after round t increases. This is because the impact of the mini-batch size of user t on the training loss increases when there are more of those users.

6.6 Simulation Results

In this section, we conduct synthetic data simulations to validate the theoretical findings. We implement a simulated system consisting of a virtual server and a number of virtual users. For convex optimization problem, we generate 10000 data samples according to the linear model, i.e., $y = \mathbf{w}^T\mathbf{x}$, and use the mean square error function as the loss function, i.e., $f(\mathbf{w}, \xi) = \frac{1}{2}\|y - \mathbf{w}^T\mathbf{x}\|^2$, where $\xi = (\mathbf{x}, y)$ is a data sample. Each data point consists of 10 features and 1 label. For non-convex optimization problems, we generate 10000 data samples for a binary linear classification model of which the data is generated according to $\Pr(y = 1|\mathbf{x} = x) = \lambda(\langle \mathbf{w}, x \rangle)$ [122], where $\lambda(\langle \mathbf{w}, x \rangle) = (1 + e^{-\lambda})^{-1}$. Each data point consists of 10 features and 1 label. The stepsize η is set as a constant for all settings.

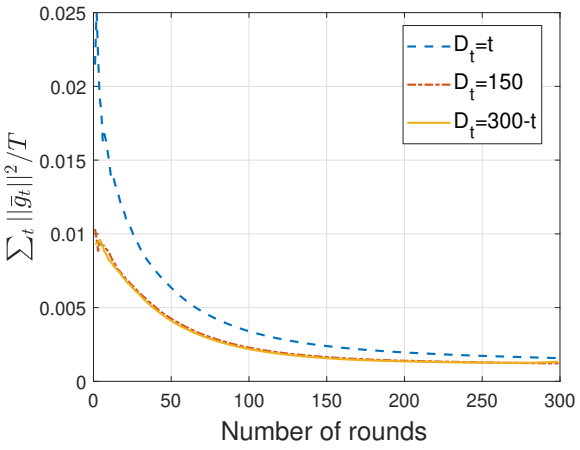

Figure 6.2: Impact of mini-batch size on the training loss of synchronous FL for non-convex optimization.

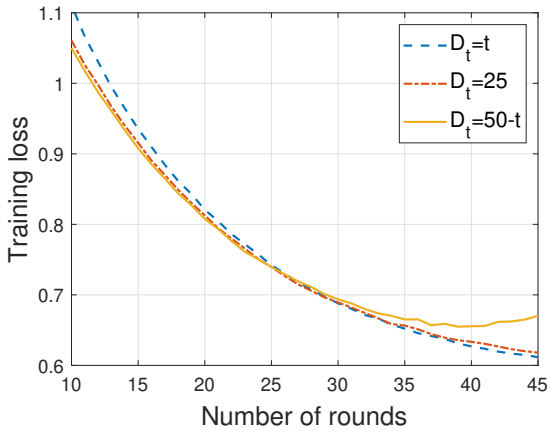

Figure 6.3: Impact of mini-batch size on the training loss of asynchronous FL for convex optimization.

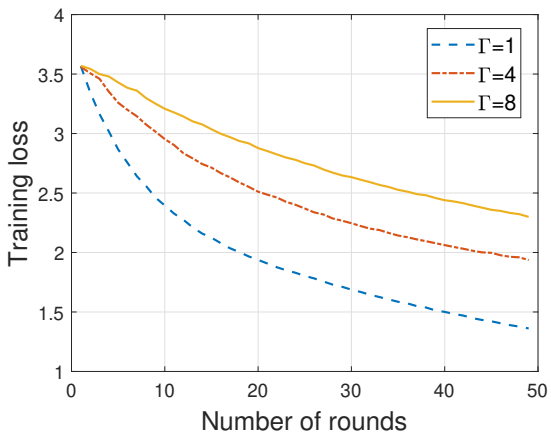

Figure 6.4: Impact of maximum update delay on the training loss of asynchronous FL for convex optimization.

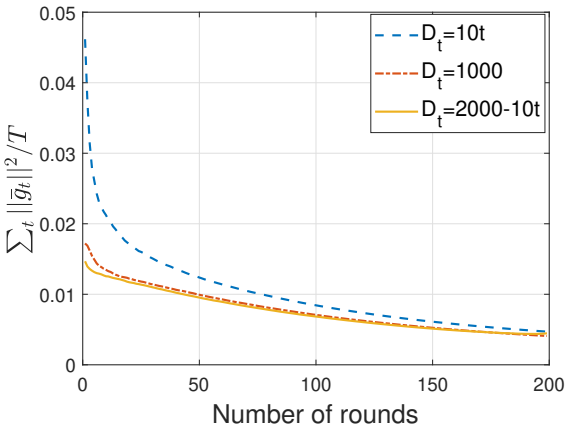

Figure 6.5: Impact of mini-batch size on the training loss of asynchronous FL for non-convex optimization.

We first evaluate the performance of the case of non-convex loss function with synchronous learning. We compare the training loss using time-invariant, descending and ascending mini-batch sizes to update the global model over rounds. The average mini-batch size over all rounds are the same for above three distributions to achieve fair comparison. Fig. 6.2 shows that although the mini-batch sizes over time of three distributions are different, they result in the same training loss at the end of training. The case of ascending mini-batch size has the worst learning accuracy in beginning rounds, and the case of descending mini-batch size has the best learning accuracy in beginning rounds. This is because, in beginning rounds, the case of descending mini-batch size uses more data to update the FL model. In our theoretical result, we have shown that using larger mini-batch to update implies better quality and leads to a lower training loss. Moreover, since the weight of local updates' quality (stepsize η) is the same in all rounds, the impact of local updates' quality on the training loss is the same. In ending rounds, as the total mini-batch size over rounds converges to the same for three cases, the training loss converges to the same value.

We next evaluate the performance of the case of convex loss function with asynchronous learning. We first compare the training loss using time-invariant, descending and ascending mini-batch sizes to update the global model. Different from the case of non-convex loss function with synchronous learning, Fig. 6.3 shows that although the average mini-batch sizes over time of three distributions are the same, different distributions of the mini-batch size result in

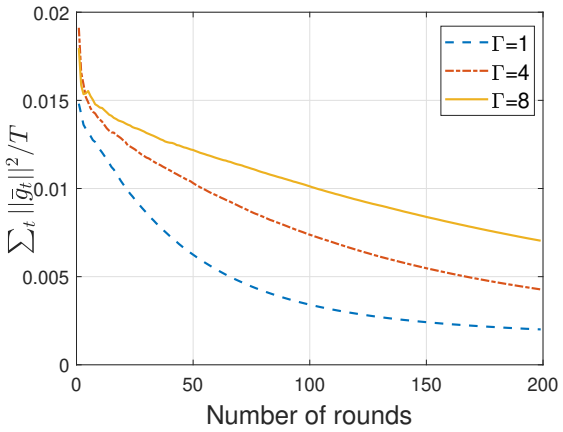

Figure 6.6: Impact of maximum update delay on the training loss of asynchronous FL for non-convex optimization.

different training loss at the end of training. The case of ascending mini-batch size has the worst learning accuracy in beginning rounds and results in the best learning accuracy in ending rounds. This conforms the result from Theorem 6.2 that the update in a later round has a larger impact on the learning accuracy. We also compare the training loss while users' maximum update delays are different ($\Gamma \in \{1, 4, 8\}$). We simulate for 50 local iterations in total, with the mini-batch size in each local iteration set as 25. From Fig. 6.4, we can see that when users update without delay ($\Gamma = 1$), the system suffers the lowest training loss. The training loss increases as the maximum update delay Γ increases. we can see that the simulation result is consistent with the result given in Theorem 6.2. FL suffers a larger error caused by the delay of users' updates.

Lastly, we evaluate the training loss of the case of non-convex loss function with asynchronous learning. Same as the other two cases, we first compare the training loss using time-invariant, descending and ascending mini-batch sizes to update the global model when the maximum update delay $\Gamma = 4$. Fig. 6.5 shows that even with the update delay, the three distributions result in the same training loss at the end of training which is the same as the case of non-convex loss function with synchronous learning. We then compare the training loss while users' maximum update delays are different ($\Gamma \in \{1, 4, 8\}$). In Fig. 6.6, we can see that the result is similar to that of the case of convex loss function with asynchronous learning.

6.7 Conclusion

In this chapter, we have studied quality-aware distributed computation for WFL with non-convex problems and asynchronous algorithms. We have characterized the performance bounds on the training loss as a function of users' local updates' quality over the training process, for both non-convex and asynchronous settings. The results show that the impact of a local update's quality 1) increases with the stepsize used in the round for non-convex learning, and 2) increases when there are more other users' local updates (depending on the update delays) which are coupled with that local update for asynchronous learning. We have also developed channel-aware adaptive algorithms that select participating users and determine their mini-batch sizes. Simulations have been used to evaluate the proposed algorithms.

6.8 Appendix

6.8.1 Proof of Theorem 6.1

First, we have

$$E\left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}(g_t^i-\bar{g}_t)\right\|^2=\sum_{i\in S_t}\frac{D_t^{i2}}{D_t^2}E\left\|g_t^i-\bar{g}_t\right\|^2\leq\sum_{i\in S_t}\frac{D_t^{i2}}{D_t^2}\frac{\sigma^2}{D_t^i}=\frac{\sigma^2}{D_t},\tag{6.13}$$

where the inequality follows from (6.3).

Suppose that S_t is the selected user set in round t, we have

$$
\begin{aligned}
&E\left(F\left(\mathbf{w}_t\right)-F\left(\mathbf{w}_{t-1}\right)\right) \\
&\leqslant -\eta_t E\left\langle \nabla F\left(\mathbf{w}_{t-1}\right), \sum_{i\in S_t}\frac{D_t^i}{D_t}\bar{g}_t^i \right\rangle + \frac{L\eta_t^2}{2}E\left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}g_t^i\right\|^2 \\
&\leqslant -\eta_t E\left\langle \nabla F\left(\mathbf{w}_{t-1}\right), \sum_{i\in S_t}\frac{D_t^i}{D_t}\bar{g}_t^i \right\rangle + \frac{L\eta_t^2}{2}\left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}(g_t^i-\bar{g}_t)\right\|^2 + \frac{L\eta_t^2}{2}E\left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}\bar{g}_t^i\right\|^2 \\
&\leqslant -\frac{\eta_t^2}{2}\left(\left\|\bar{g}_t\right\|^2 + \left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}\bar{g}_t^i\right\|^2 - \left\|\bar{g}_t - \sum_{i\in s_t}\frac{D_t^i}{D_t}\bar{g}_t^i\right\|^2\right) + \frac{L\eta_t^2}{2}\frac{\sigma^2}{D_t} + \frac{L\eta_t^2}{2}E\left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}\bar{g}_t^i\right\|^2 \\
&= -\frac{\eta_t^2}{2}\left\|\bar{g}_t\right\|^2 + \left(\frac{L\eta_t^2}{2}-\frac{\eta_t}{2}\right)\left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}\bar{g}_t^i\right\|^2 + \frac{L\eta_t^2}{2}\frac{\sigma^2}{D_t} \leqslant -\frac{\eta_t^2}{2}\left\|\bar{g}_t\right\|^2 + \frac{L\eta_t^2}{2}\frac{\sigma^2}{D_t}
\end{aligned}
$$

Thus we have

$$
\eta_t\left\|\bar{g}_t\right\|^2 \leq -2\left(E\left(F\left(\mathbf{w}_t\right)-F\left(\mathbf{w}_{t-1}\right)\right)\right) + L\eta_t^2\frac{\sigma^2}{D_t}.
$$

Summing over $t\in\{1,\ldots,T\}$ and dividing both sides by $\sum_{t=1}^T\eta_t$ yields (6.4).

6.8.2 Proof of Theorem 6.2

First, we have

$$
\begin{aligned}
&E\left(F\left(\mathbf{w}_t\right)-F\left(\mathbf{w}^*\right)\right) \\
\leq& F\left(\mathbf{w}_{t-1}\right)-F\left(\mathbf{w}^*\right) - \eta E\left\langle \nabla F\left(\mathbf{w}_{t-1}\right), \sum_{i\in M_t}\frac{D_t^i}{D_t}\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right)\right\rangle + \frac{L\eta^2}{2}E\left\|\sum_{i\in M_t}\frac{D_t^i}{D_t}\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right)\right\|^2 \\
\leq& F\left(\mathbf{w}_{t-1}\right)-F\left(\mathbf{w}^*\right) + \frac{\eta}{2}\underbrace{E\left\|\nabla F\left(\mathbf{w}_{t-1}\right)-\sum_{i\in M_t}\frac{D_t^i}{D_t}\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right)\right\|^2}_{A} - \frac{\eta}{2}E\left\|\nabla F\left(\mathbf{w}_{t-1}\right)\right\|^2,
\end{aligned}
$$

where M_t is the set of users who update their local model in round t.

Next we focus on bounding A.

$$
\begin{aligned}
&A \\
=&E\left\|\nabla F\left(\mathbf{w}_{t-1}\right)-\sum_{i\in M_t}\frac{D_t^i}{D_t}E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right))+\sum_{i\in M_t}\frac{D_t^i}{D_t}E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right))-\sum_{i\in M_t}\frac{D_t^i}{D_t}\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right)\right\|^2 \\
\leq&E\left\|\nabla F\left(\mathbf{w}_{t-1}\right)-\sum_{i\in M_t}\frac{D_t^i}{D_t}E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right))\right\|^2 \\
&+E\left\|\sum_{i\in M_t}\frac{D_t^i}{D_t}E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right))-\sum_{i\in M_t}\frac{D_t^i}{D_t}\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right)\right\|^2 \\
\leq&E\left\|\nabla F\left(\mathbf{w}_{t-1}\right)-\sum_{i\in M_t}\frac{D_t^i}{D_t}E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right))\right\|^2+\sum_{i\in M_t}\frac{D_t^{i2}}{D_t^2}E\left\|E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right))-\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right)\right\|^2 \\
\leq&\sum_{i\in M_t}\frac{D_t^i}{D_t}L^2\left\|\mathbf{w}_{t-1}-\mathbf{w}_{t-\tau_i}\right\|^2+\frac{\sigma^2}{D_t}.
\end{aligned}
\tag{6.14}
$$

Then we bound each term

$$
\begin{aligned}
\left\|\mathbf{w}_{t-1}-\mathbf{w}_{t-\tau_i}\right\|^2\leq&\left\|\sum_{t'=t-\tau_i+1}^{t-1}\eta\sum_{i'\in M_{t'}}\frac{D_{t'}^{i'}}{D_{t'}}\nabla F_i\left(\mathbf{w}_{t'-\tau_{i'}}\right)\right\|^2 \\
=&2\eta^2\left\|\sum_{t'=t-\tau_i+1}^{t-1}\sum_{i'\in M_{t'}}\frac{D_{t'}^{i'}}{D_{t'}}(\nabla F_i\left(\mathbf{w}_{t'-\tau_{i'}}\right)-E(\nabla F_i\left(\mathbf{w}_{t'-\tau_{i'}}\right)))\right\|^2 \\
&+2\eta^2\left\|\sum_{t'=t-\tau_i+1}^{t-1}\sum_{i'\in M_{t'}}\frac{D_{t'}^{i'}}{D_{t'}}E(\nabla F_i\left(\mathbf{w}_{t'-\tau_{i'}}\right))\right\|^2 \\
\leq&2\eta^2\sum_{t'=t-\tau_i+1}^{t-1}\left\|\sum_{i'\in M_{t'}}\frac{D_{t'}^{i'}}{D_{t'}}(\nabla F_i\left(\mathbf{w}_{t'-\tau_{i'}}\right)-E(\nabla F_i\left(\mathbf{w}_{t'-\tau_{i'}}\right)))\right\|^2 \\
&+2\eta^2\Gamma\sum_{t'=t-\tau_i+1}^{t-1}\left\|\sum_{i'\in M_{t'}}\frac{D_{t'}^{i'}}{D_{t'}}E(\nabla F_i\left(\mathbf{w}_{t'-\tau_{i'}}\right))\right\|^2 \\
\leq&2\eta^2\sum_{t'=t-\tau_i+1}^{t-1}\frac{\sigma^2}{D_{t'}}+2\eta^2\Gamma(\tau_i-1)B^2.
\end{aligned}
\tag{6.15}
$$

Thus we have

$$
\begin{aligned}
&E\left(F\left(\mathbf{w}_T\right)-F\left(\mathbf{w}^*\right)\right) \leq F\left(\mathbf{w}_{T-1}\right)-F\left(\mathbf{w}^*\right) \\
&+L^2\eta^3\sum_{i\in M_T}\frac{D_T^i}{D_T}\left(\sum_{t'=T-\tau_i+1}^{T-1}\frac{\sigma^2}{D_{t'}}+\Gamma(\tau_i-1)B^2\right)-\frac{\eta}{2}E\left\|\nabla F\left(\mathbf{w}_{T-1}\right)\right\|^2+\frac{\eta}{2}\frac{\sigma^2}{D_T} \\
\leq&(1-\mu\eta)(F\left(\mathbf{w}_{T-1}\right)-F\left(\mathbf{w}^*\right)) \\
&+L^2\eta^3\sum_{i\in M_T}\frac{D_T^i}{D_T}\left(\sum_{t'=T-\tau_i+1}^{T-1}\frac{\sigma^2}{D_{t'}}+\Gamma(\tau_i-1)B^2\right)+\frac{\eta}{2}\frac{\sigma^2}{D_T} \\
\leq&(1-\mu\eta)^T(F(\mathbf{w}_0)-F(\mathbf{w}^*)) \\
&+\sum_{t=1}^{T}\left[(1-\mu\eta)^{T-t}\left(L^2\eta^3\sum_{i\in M_t}\frac{D_t^i}{D_t}(\sum_{t'=t-\tau_i+1}^{t-1}\frac{\sigma^2}{D_{t'}}\ +\Gamma(\tau_i-1)B^2)+\frac{\eta}{2}\frac{\sigma^2}{D_t}\right)\right]
\end{aligned}
$$

6.8.3 Proof of Theorem 6.3

First, we have

$$
\begin{aligned}
&E\left(F\left(\mathbf{w}_t\right)-F\left(\mathbf{w}_{t-1}\right)\right) \\
\leqslant&-\eta_t E\left\langle\nabla F\left(\mathbf{w}_{t-1}\right),\sum_{i\in S_t}\frac{D_t^i}{D_t}\bar{g}_t^i\right\rangle+\frac{L\eta_t^2}{2}E\left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}g_t^i\right\|^2 \\
=&-\frac{\eta_t}{2}\left(\left\|E(\nabla F_i\left(\mathbf{w}_{t-1}\right))\right\|^2+\left\|\sum_{i\in M_t}\frac{D_t^i}{D_t}E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right))\right\|^2\right. \\
&\left.+\underbrace{\left\|E(\nabla F_i\left(\mathbf{w}_{t-1}\right))-\sum_{i\in M_t}\frac{D_t^i}{D_t}E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right))\right\|^2}_{A_1}\right)+\frac{L\eta_t^2}{2}\underbrace{E\left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}g_t^i\right\|^2}_{A_2}.
\end{aligned}
$$

From (6.14) and (6.15), A_1 can be bounded as

$$
\begin{aligned}
A_1\leq&\sum_{i\in M_t}\frac{D_t^i}{D_t}L^2\left\|\mathbf{w}_{t-1}-\mathbf{w}_{t-\tau_i}\right\|^2 \\
\leq&2L^2\sum_{i\in M_t}\frac{D_t^i}{D_t}\left(\sum_{t'=t-\tau_i+1}^{t-1}\eta_{t'}^2\frac{\sigma^2}{D_{t'}}+\Gamma\sum_{t'=t-\tau_i+1}^{t-1}\eta_{t'}^2\left\|\sum_{i'\in M_{t'}}\frac{D_{t'}^{i'}}{D_{t'}}E(\nabla F_i\left(\mathbf{w}_{t'-\tau_{i'}}\right))\right\|^2\right).
\end{aligned}
$$

Next, we bound A_2.

$$
\begin{aligned}
&A_2\\
=&E\left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right)-E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right))+E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right)))\right\|^2\\
=&E\left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right)-E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right)))\right\|^2+E\left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right))\right\|^2\\
\leq&\frac{\sigma^2}{D_t}+E\left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right))\right\|^2,
\end{aligned}
$$

where the inequality follows (6.13).

Thus, we have

$$
\begin{aligned}
&E\left(F\left(\mathbf{w}_t\right)-F\left(\mathbf{w}_{t-1}\right)\right)\\
\leq&-\frac{\eta_t}{2}\left(\left\|E(\nabla F_i\left(\mathbf{w}_{t-1}\right))\right\|^2+\left\|\sum_{i\in M_t}\frac{D_t^i}{D_t}E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right))\right\|^2+2L^2\sum_{i\in M_t}\frac{D_t^i}{D_t}\left(\sum_{t'=t-\tau_i+1}^{t-1}\eta_{t'}^2\frac{\sigma^2}{D_{t'}}\right.\right.\\
&\left.\left.+\Gamma\sum_{t'=t-\tau_i+1}^{t-1}\eta_{t'}^2\left\|\sum_{i'\in M_{t'}}\frac{D_{t'}^{i'}}{D_{t'}}E(\nabla F_i\left(\mathbf{w}_{t'-\tau_{i'}}\right))\right\|^2\right)\right)+\frac{L\eta_t^2}{2}\left(\frac{\sigma^2}{D_t}+E\left\|\sum_{i\in S_t}\frac{D_t^i}{D_t}E(\nabla F_i\left(\mathbf{w}_{t-\tau_i}\right))\right\|^2\right).
\end{aligned}
$$

Summing over $t\in\{1,\ldots,T\}$ we have

$$
\begin{aligned}
&E\left(F\left(\mathbf{w}_t\right)-F\left(\mathbf{w}_0\right)\right)\leq\\
&\sum_{t=1}^{T}-\frac{\eta_t}{2}\left\|\bar{g}_t\right\|^2+\sum_{t=1}^{T}\left(\left(\frac{L\eta_t^2}{2}-\frac{\eta_t}{2}+L^2\Gamma\eta_t^2\sum_{t'=1}^{\Gamma}\eta_{t+t'}\right)\left\|\sum_{i\in M_t}\frac{D_t^i}{D_t}E(\nabla F(\mathbf{w}_{t-\tau_i}))\right\|^2\right)\\
&+\sum_{t=1}^{T}\left(\eta_tL^2\sum_{i\in M_t}\frac{D_t^i}{D_t}\sum_{t'=t-\tau_i+1}^{t-1}\eta_{t'}^2\frac{\sigma^2}{D_{t'}}+\frac{L\eta_t^2}{2}\frac{\sigma^2}{D_t}\right)\\
\leq&\sum_{t=1}^{T}-\frac{\eta_t}{2}\left\|\bar{g}_t\right\|^2+\sum_{t=1}^{T}\left(\eta_tL^2\sum_{i\in M_t}\frac{D_t^i}{D_t}\sum_{t'=t-\tau_i+1}^{t-1}\eta_{t'}^2\frac{\sigma^2}{D_{t'}}+\frac{L\eta_t^2}{2}\frac{\sigma^2}{D_t}\right),
\end{aligned}
$$

where the last inequality follows $L\eta_t+L^2\Gamma\eta_t\sum_{t'=1}^{\Gamma-1}\eta_{t+t'}\leq 1$.

Then we have

$$\sum_{t=1}^{T} \eta_t \left\|\bar{g}_t\right\|^2 \leq 2E\left(F\left(\mathbf{w}_0\right) - F\left(\mathbf{w}^*\right)\right) + \sum_{t=1}^{T}\left(2\eta_t L^2 \sum_{i\in M_t} \frac{D_t^i}{D_t} \sum_{t'=t-\tau_i+1}^{t-1} \eta_{t'}^2 \frac{\sigma^2}{D_{t'}} + L\eta_t^2 \frac{\sigma^2}{D_t}\right),$$

Dividing both sides by $\sum_{t=1}^{T} \eta_t$ yields (6.9).

6.8.4 Proof of Lemma 6.1

First, we have that $-\mathcal{J}_1(S_t, D_t^*(S_t))$ is a non-monotone function, since its first term increases with $|S_t|$ and its second term decreases with $|S_t|$.

Next, we prove that $-\mathcal{J}_1(S_t, D_t^*(S_t))$ is a submodular function. For ease of expression, we write $-\mathcal{J}_1(S_t, D_t^*(S_t))$ as $-\mathcal{J}_1(S_t)$ in the following proof.

From (6.4), we can see that a user's optimal data sampling size has three possible values which are 0, $\eta_t\sigma\sqrt{\frac{L\gamma}{(1-\gamma)c_{p,t}^i\sum_{t-1}^{T}\eta_t}} - \sum_{j=1}^{i-1} D_t^{j*}$, and D_B^i. From Theorem 6.4, we know that for any selected user set $S_1 \subset S_2$,

$$\sum_{i\in S_1} D_t^{i*}(S_1) \leq \sum_{i\in S_2} D_t^{i*}(S_2). \tag{6.16}$$

We also know that after reordering users, the optimal data sampling size of user j' is always no greater than that of user j for any $j' > j$. Thus we have, for any selected user set $S_1 \subset S_2$ and any user $x \in S_2$, there are four possible combinations of $D_t^{x*}(S_1 \cup \{x\})$ and $D_t^{x*}(S_2 \cup \{x\})$:

1) $D_t^{x*}(S_1 \cup \{x\}) = D_t^{x*}(S_2 \cup \{x\})$;

2) $D_t^{x*}(S_1 \cup \{x\}) = D_B^x$ and $D_t^{x*}(S_2 \cup \{x\}) = \eta_t\sigma\sqrt{\frac{L\gamma}{(1-\gamma)c_{p,t}^i\sum_{t-1}^{T}\eta_t}} - \sum_{i=1}^{x-1} D_t^{i*}(S_2)$;

3) $D_t^{x*}(S_1 \cup \{x\}) = D_B^x$ and $D_t^{x*}(S_2 \cup \{x\}) = 0$;

4) $D_t^{x*}(S_1 \cup \{x\}) = \eta_t\sigma\sqrt{\frac{L\gamma}{(1-\gamma)c_{p,t}^i\sum_{t-1}^{T}\eta_t}} - \sum_{i=1}^{x-1} D_t^{i*}(S_1)$ and $D_t^{x*}(S_2 \cup \{x\}) = 0$.

We also give the definition of submodular functions.

Definition 6.1 *A set function f on S is submodular if and only if*

$$f(S_1 \cup \{x\}) - f(S_1) \geq f(S_2 \cup \{x\}) - f(S_2)$$

for each $S_1 \subset S_2 \subseteq S$ and $x \in S \setminus S_2$.

For any round t, selected user set S_t and user $x \notin S_t$, we have

$$\begin{aligned} & -\mathcal{J}(S_t \cup \{x\}) - (-\mathcal{J}(S_t)) \\ &= \frac{\gamma L \eta_t^2 \sigma^2}{\sum_{t=1}^{T} \eta_t} \left(\frac{D_t^{x*}(S_t \cup \{x\})}{\sum\limits_{i \in S_t} D_t^{i*}(S_t) \sum\limits_{i \in S_t \cup \{x\}} D_t^{i*}(S_t \cup \{x\})} \right) - (1-\gamma) \left(c_{p,t}^{x} D_t^{x*}(S_t \cup \{x\}) + c_{m,t}^{x} \right) \end{aligned} \tag{6.17}$$

Then, for each combination of $D_t^{x*}(S_1 \cup \{x\})$ and $D_t^{x*}(S_2 \cup \{x\})$, we prove that $-\mathcal{J}(S_1 \cup \{x\}) + \mathcal{J}(S_1) \geq -\mathcal{J}(S_2 \cup \{x\}) + \mathcal{J}(S_2)$.

1) $D_t^{x*}(S_1 \cup \{x\}) = D_t^{x*}(S_2 \cup \{x\})$:

From (6.16) and (6.17), we have

$$\begin{aligned} & -\mathcal{J}(S_1 \cup \{x\}) + \mathcal{J}(S_1) - (-\mathcal{J}(S_2 \cup \{x\}) + \mathcal{J}(S_2)) \\ &= \frac{\gamma L \eta_t^2 \sigma^2}{\sum_{t=1}^{T} \eta_t} \left(\frac{D_t^{x*}(S_1 \cup \{x\})}{\sum\limits_{i \in S_1} D_t^{i*}(S_1) \sum\limits_{i \in S_1 \cup \{x\}} D_t^{i*}(S_1 \cup \{x\})} \right) \\ &- \frac{\gamma L \eta_t^2 \sigma^2}{\sum_{t=1}^{T} \eta_t} \left(\frac{D_t^{x*}(S_2 \cup \{x\})}{\sum\limits_{i \in S_2} D_t^{i*}(S_2) \sum\limits_{i \in S_2 \cup \{x\}} D_t^{i*}(S_2 \cup \{x\})} \right) \\ &\geq 0. \end{aligned}$$

2) $D_t^{x*}(S_1 \cup \{x\}) = D_B^x$ and $D_t^{x*}(S_2 \cup \{x\}) = \eta_t \sigma \sqrt{\frac{L\gamma}{(1-\gamma) c_{p,t}^{i} \sum_{t-1}^{T} \eta_t}} D_t^{i*}$:

From (6.4), we have

$$D_B^x < \eta_t \sigma \sqrt{\frac{L\gamma}{(1-\gamma) c_{p,t}^{i} \sum_{t-1}^{T} \eta_t}} - \sum_{j \in S_1} D_t^{i*}(S_1)$$

and

$$D_B^x > \eta_t \sigma \sqrt{\frac{L\gamma}{(1-\gamma)c_{p,t}^i \sum_{t-1}^T \eta_t}} - \sum_{j \in S_2} D_t^{i*}(S_2).$$

Thus we have

$$\frac{\gamma L \eta_t^2 \sigma^2 / (1-\gamma) \sum_{t=1}^T \eta_t}{(D_B^x + \sum_{i \in S_1} D_t^{i*}(S_1))^2} \leq c_{p,t}^x \leq \frac{\gamma L \eta_t^2 \sigma^2 / (1-\gamma) \sum_{t=1}^T \eta_t}{(D_B^x + \sum_{i \in S_2} D_t^{i*}(S_2))^2}. \tag{6.18}$$

From (6.17) and (6.18), we have

$$\begin{aligned}
&- \mathcal{J}(S_1 \cup \{x\}) + \mathcal{J}(S_1) - (-\mathcal{J}(S_2 \cup \{x\}) + \mathcal{J}(S_2)) \\
&\geq \frac{\gamma L \eta_t^2 \sigma^2}{\sum_{t=1}^T \eta_t} \Bigg[\frac{D_B^x}{\sum_{i \in S_1} D_t^{i*}(S_1)(D_B^x + \sum_{i \in S_1} D_t^{i*}(S_1))} \\
&\quad - \frac{\eta_t \sigma \sqrt{\frac{L\gamma}{(1-\gamma)c_{p,t}^i \sum_{t-1}^T \eta_t}} - \sum_{i \in S_2} D_t^{i*}(S_2)}{\sum_{i \in S_2} D_t^{i*}(S_2)(D_B^x + \sum_{i \in S_2} D_t^{i*}(S_2))} - \\
&\quad \frac{D_B^x}{(D_B^x + \sum_{i \in S_1} D_t^{i*}(S_1))^2} + \frac{\eta_t \sigma \sqrt{\frac{L\gamma}{(1-\gamma)c_{p,t}^i \sum_{t-1}^T \eta_t}} - \sum_{i \in S_2} D_t^{i*}(S_2)}{(D_B^x + \sum_{i \in S_2} D_t^{i*}(S_2))^2} \Bigg] \\
&= \frac{{D_B^x}^2}{\sum_{i \in S_1} D_t^{i*}(S_1)(D_B^x + \sum_{i \in S_1} D_t^{i*}(S_1))^2} - \frac{(\eta_t \sigma \sqrt{\frac{L\gamma}{(1-\gamma)c_{p,t}^i \sum_{t-1}^T \eta_t}} - \sum_{i \in S_2} D_t^{i*}(S_2))^2}{\sum_{i \in S_2} D_t^{i*}(S_2)(D_B^x + \sum_{i \in S_2} D_t^{i*}(S_2))^2} \\
&\geq 0.
\end{aligned}$$

3) $D_t^{x*}(S_1 \cup \{x\}) = D_B^x$ and $D_t^{x*}(S_2 \cup \{x\}) = 0$:

Since $D_t^{x*}(S_2 \cup \{x\}) = 0$, we have $-\mathcal{J}(S_2 \cup \{x\}) + \mathcal{J}(S_2) = -(1-\gamma)c_{m,t}^x$. Thus, we have

$$
\begin{aligned}
&- \mathcal{J}(S_1 \cup \{x\}) + \mathcal{J}(S_1) - (-\mathcal{J}(S_2 \cup \{x\}) + \mathcal{J}(S_2)) \\
&= \frac{\gamma L \eta_t^2 \sigma^2}{\sum_{t=1}^T \eta_t} \left[\frac{D_B^x}{\sum\limits_{i \in S_t} D_t^{i*}(S_t) \sum\limits_{i \in S_t \cup \{x\}} D_t^{i*}(S_t \cup \{x\})} \right] - (1-\gamma) \left(c_{p,t}^x D_B^x \right) \\
&\geq \frac{\gamma L \eta_t^2 \sigma^2}{\sum_{t=1}^T \eta_t} \left[\frac{\eta_t \sigma \sqrt{\frac{L\gamma}{(1-\gamma)c_{p,t}^i \sum_{t-1}^T \eta_t}} - \sum\limits_{i \in S_1} D_t^{i*}(S_1)}{\sum\limits_{i \in S_t} D_t^{i*}(S_t) \sum\limits_{i \in S_t \cup \{x\}} D_t^{i*}(S_t \cup \{x\})} \right] \\
&- (1-\gamma) \left[c_{p,t}^x (\eta_t \sigma \sqrt{\frac{L\gamma}{(1-\gamma)c_{p,t}^i \sum_{t-1}^T \eta_t}} - \sum_{i \in S_1} D_t^{i*}(S_1)) \right] \\
&= 0.
\end{aligned}
$$

4) $D_t^{x*}(S_1 \cup \{x\}) = \eta_t \sigma \sqrt{\frac{L\gamma}{(1-\gamma)c_{p,t}^i \sum_{t-1}^T \eta_t}} - \sum_{i \in S_1} D_t^{i*}(S_1)$ and $D_t^{x*}(S_2 \cup \{x\}) = 0$:

We omit the proof of this combination since it is similar with that of combination 2).

6.8.5 Proof of Theorem 6.5

From Lemma 6.2, we have for any round t,

$$\mathcal{G}(S_t) \geq \frac{1}{3} \mathcal{G}(\mathcal{OPT}_t),$$

where $\mathcal{OPT}_t$ is the optimal user set in round t.

Then, we have for any round t,

$$\mathcal{J}_1(S_t, D_t^*(S_t)) + \mathcal{J}_{1,\max} \geq \frac{1}{3} (\mathcal{J}_1(\mathcal{OPT}_t, D_t^*(\mathcal{OPT}_t)) + \mathcal{J}_{1,\max}).$$

Thus, the system loss over T rounds is bounded by

$$\sum_{t=1}^{T} \mathcal{J}_1(S_t, D_t^*(S_t)) \geq \sum_{t=1}^{T} (\frac{1}{3}\mathcal{J}_1(\mathcal{OPT}_t, D_t^*(\mathcal{OPT}_t)) + \frac{2}{3}\mathcal{J}_{1,\max})$$
$$\geq \frac{1}{3}\mathcal{OPT} + \mathcal{O}(T).$$

6.8.6 Other Omitted Proofs

Proposition 6.1 directly follows from Theorem 6.1 by replacing the time-varying stepsize with the constant stepsize.

Theorem 6.4 can be obtained by solving $\mathcal{J}_1'(D_t) = 0$, where $\mathcal{J}_1'(D_t)$ is the first derivative of $\mathcal{J}_1(D_t)$.

Theorem 6.6 can be obtained by solving $\mathcal{J}_2'(D_t) = 0$, where $\mathcal{J}_2'(D_t)$ is the first derivative of $\mathcal{J}_2(D_t)$.

Chapter 7

Truthful Incentive Mechanism for Federated Learning with Crowdsourced Data Labeling.

7.1 Introduction

Federated learning (FL) [80] is an emerging and promising ML paradigm, which performs the training of ML models in a distributed manner. Instead of transmitting data from a potentially large number of devices to a central server in the edge or cloud for training, FL allows the data to remain at devices (such as smartphone), and trains a global ML model on the server by collecting and aggregating model updates locally computed on each device based on her local data. One significant advantage of using FL is to preserve the privacy of individual device's data. Moreover, since only local ML model updates, instead of local data, are sent to the server, the communication costs can be greatly reduced. Furthermore, FL can exploit the substantial computation capabilities of ubiquitous smart devices, which are often under-utilized. As a result, FL can achieve collaborative intelligence, which can enable many AI applications based on networked systems, such as connected and autonomous vehicles, collaborative robots, multi-user virtual/mixed reality.

Recent studies on FL typically focus on supervised learning, which requires a large amount of training data with data labels in the learning process. In many applications of ML, data labels have to be generated manually by human users. For example, for image classification, the object in an image should be recognized and annotated by a human user as the label of the image data. Therefore, as FL does not allow a client to share her local data with the server or other clients, to participate in FL, a client needs to manually label her local data (e.g., images), before she can compute local model updates from her locally labeled data.

However, data labels generated by human clients of FL are subject to errors. For example, a client may misclassify a dog as a cat. As a result, this incorrect data label will lead to error in the local model, and thus error in the global model obtained by the FL server. Moreover, the labeling error rate of a client generally varies for different clients, depending on the client's knowledge level of the data labeling task. For example, a client who is familiar with dogs will have a lower labeling error rate than another client who is not. Furthermore, the accuracy of data labels is also affected by a client's effort made in the data labeling task. The data label error rate will be low when the client makes much effort in labeling the data, and otherwise is high when the client makes little or no effort. For example, a client may make no effort in image classification by randomly guessing the object in an image without actually recognizing the object.

While a client's effort impacts the accuracy of her data labels, the effort can be her hidden action that is only known by the client herself and cannot be observed by the FL server. Due to the inaccurate nature of data labels, a strategic client may label her local data arbitrarily without making effort in data labeling, while the server will not be able to verify whether effort is actually made or not. Moreover, the effort made by a client in computing her local model update, which can be quantified by the mini-batch size used by the client in stochastic gradient descent, can also be the client's hidden action that cannot be verified by the server. As a result, a client may have incentive to compute her local update with a small mini-batch size so as to reduce her resources used in local computation. Furthermore, the local model computed by a client from her local data can also be her private information that she can manipulate in favor of herself, e.g., a client may increase or decrease her true local model and report it to the server.

In the presence of such strategic clients with hidden data labeling and local computation efforts and private local models, our goal is to incentivize the clients to make truthful efforts as desired by the FL server and reveal their true local models. Such a truthful incentive mechanism is desirable as it eliminates the possibility of manipulation, which would encourage clients to participate in FL. More importantly, the truthful elicitation of clients' efforts and local models ensures that the FL server can obtain a global model with high and guaranteed accuracy from the learning process, which is a key performance metric of FL.

The joint elicitation of data labeling effort, local computation effort, and local models for FL calls for new design that is very different from existing truthful mechanisms. First, the training loss of the global model obtained from FL has a non-trivial dependence on clients' exerted efforts and reported models. As a result, existing incentive mechanisms for effort and data elicitation do not work for the problem here. Second, due to the complex relation between the impacts of labeling effort, computation effort, and local models on the training loss, the joint elicitation of effort and models needs to overcome the coupling therein. Third, given the truthful incentive mechanism for effort and model elicitation, the FL server needs to determine how much effort should be made by each client, in order to maximize the server's payoff.

The main contributions of this chapter can be summarized as follows.

- We propose an FL framework with crowdsourced data labeling based on a truthful incentive mechanism, where the labels of a client's local training data for FL are manually generated by the human client and are subject to errors. We consider strategic clients whose actual efforts in data labeling and local model computation as well as actual local models cannot be verified by the FL server.

- We first characterize the performance bounds on the training loss as a function of clients' data labeling effort, local computation effort (quantified by the mini-batch size), and reported local models. It shows that the labeling and computation efforts as well as the reported models have non-trivial impacts on the training loss. Based on the obtained insights, we develop the Labeling and Computation Effort and Local model Elicitation (LCEME) mechanism which incentivize clients to truthfully make efforts in data labeling and local computation, and report local models. The truthful design of the LCEME mechanism overcomes the intricate coupling in the joint elicitation of labeling effort, computation effort, and local models. Based on the LCEME mechanism, we then characterize the optimal computation effort assignment for maximizing the FL server's payoff.

- We evaluate the proposed FL with crowdsourced data labeling using the popular MNIST-based hand-written digit recognition problem. The results demonstrate that the proposed

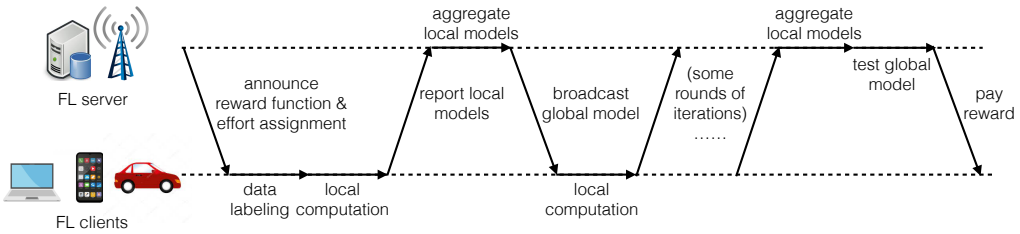

Figure 7.1: Schedule of FL with crowdsourced data labeling based on a truthful incentive mechanism.

algorithms outperform the methods that do not consider data labeling errors or do not use an incentive mechanism.

The remainder of this chapter is organized as follows. Section 7.2 reviews the related work. In Section 7.3, we describe the system model and formulate the problem of incentive mechanism design. In Section 7.4, we study the performance bound on the training loss. In Section 7.5, we devise the LCEME mechanism and the server's optimal effort allocation. Simulation results are presented in Section 7.6. Section 7.7 concludes this chapter.

7.2 Related Work

Incentive Mechanism for Federated Learning. FL has emerged as a disruptive computing paradigm for ML by democratizing the learning process to potentially many individual devices. Most existing studies on FL have focused on algorithm design for FL, such as for reducing the local model drifts across non-IID clients and participating clients selection. Meanwhile, there have been several recent works on computation and communication resource allocation for FL [95, 96, 97, 98, 99, 100, 101, 102]. On the other hand, a few recent work studied incentive mechanisms [123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133] for FL that take into account participating clients' strategic behavior. In particular, most of these work considered compensating clients' communication and computation costs with an economic approach, such as Stackelberg game [127], auction theory [128], cooperative game [129, 130], and contract theory [131, 132]. However, all these prior works have focused on either incentivizing clients' participation via cost compensation, or truthfully eliciting clients' participation costs. [133]

proposed VCG-based mechanisms that incentivize clients to truthfully report their local models. In contrast, this paper studies incentive mechanisms for truthful elicitation of clients' local models as well as their efforts in data labeling and local computation.

Truthful Incentive Mechanism for Effort and Data Elicitation. There have been a lot of research on incentive mechanisms for various applications of data collection and processing, particularly for data crowdsourcing [76, 22, 23, 134, 135, 136, 25, 45, 54, 55]. Many incentive mechanisms incentivize agents to truthfully reveal their participating cost, where the cost is considered to be private for a agent that may not be revealed truthfully without appropriate incentive. There have been recent studies on truthful mechanism design for hidden efforts in the economics literature [137], which is concerned with strategic agents that can make hidden efforts not desired by a principal who recruits the agents to work on a task. A few recent works have studied this problem in the context of crowdsourcing [138, 26, 25, 57, 139]. Mechanism design for truthful elicitation of strategic agents' data (e.g., opinions) has been extensively studied in various applications (e.g., [140]), more recently for crowdsourcing [138, 25, 58, 57, 139]. The data of an agent can be its private information that the agent can manipulate in favor of her benefit. Different from existing works, in this paper, we focus on FL and aim to design truthful mechanisms that jointly elicit clients' hidden efforts in data labeling and local computation and their private local models. The truthful mechanism design here is non-trivially different from existing works, due to 1) complex dependence of the training loss on clients' data labeling and local computation efforts and local models; 2) intricate coupling in joint elicitation of the clients' efforts and models.

7.3 System Model and Problem Formulation

In this section, we first describe a FL system with crowdsourced data labeling based on a truthful incentive mechanism (as illustrated in Fig. 7.1), and then present the design objectives for truthful incentive mechanisms.

7.3.1 FL with Crowdsourced Data Labeling

Consider the following FL problem:

$$\min_{\mathbf{w}} \; F(\mathbf{w}) \triangleq \sum_{i=1}^{N} p_i F_i(\mathbf{w}), \tag{7.1}$$

where $F_i(\mathbf{w})$ is defined by

$$F_i(\mathbf{w}) \triangleq \frac{1}{\tilde{D}_i} \sum_{m=1}^{\tilde{D}_i} f(\mathbf{w}; \xi_m^i),$$

$f(\cdot)$ is the per-sample loss function of client i, $N \triangleq |\mathcal{N}|$ is the number of clients, p_i is the weight of client i, $\sum_{i\in\mathcal{N}} p_i \triangleq 1$, $\mathcal{D}_i = \{\xi_1^i, \xi_2^i, \ldots, \xi_{\tilde{D}_i}^i\}$ is client i's local dataset for updating the model parameter, and $D \triangleq \sum_{i=1}^{N} \tilde{D}_i$. Without loss of generality, for ease of exposition, we assume that all clients have the same per-sample loss function $f(\cdot)$.

Data Labeling. To participate in FL tasks, each client needs to maintain a local dataset $\mathcal{D}_i$. In this chapter, we assume that the FL clients collaboratively train for classification ML tasks, and each client needs to observe class labels corresponding to the feature information that she holds. After observing the labels, each client i in the client set $\mathcal{N}$ obtains a dataset $\mathcal{D}_i$.

The labeling effort $e_i \in \{0, 1\}$ represents whether client i makes effort in labeling her data, where $e_i = 1$ and $e_i = 0$ indicate making and not making effort, respectively. If client i makes effort, then the labels in her dataset are correct; otherwise, the labels are randomly selected from all possible classes without considering the corresponding features. We know that an ML model trained on a correctly labeled dataset is more likely to make useful predictions than a model trained on incorrectly labeled data. Therefore, making effort $e_i = 1$ means higher accuracy of the trained model than not making effort (We prove this intuition in Section 7.3.). We assume that every client can fully control the amount of effort they make, and the server does not have such information.

Local Model Computation. In each round of FL, clients communicate their local updates to the server, and receive the updated global model from the server. In round t,[1] each client i receives the global model $\mathbf{w}_{t-1}$ from the server, sets $\mathbf{w}_{t,0}^i = \mathbf{w}_{t-1}$, and then performs H local iterations of SGD. In the hth local iteration, client i computes the average gradient $g_{t,h-1}^i$ of the loss function using a mini-batch of D_i data samples randomly drawn from her local dataset $\mathcal{D}_i$. Then client i updates her local model as

[1]We use t and h as the index of communication rounds and local iterations, respectively. The subscript (t, h) denotes the hth local iteration in round t.

$$\mathbf{w}_{t,h}^{i} = \mathbf{w}_{t,h-1}^{i} - \eta g_{t,h-1}^{i},$$

where

$$g_{t,h-1}^{i} \triangleq \frac{1}{D_i} \sum_{j=1}^{D_i} \nabla f(\mathbf{w}, \xi_{t,h}^{i,j}),$$

η is the step size, and $\xi_{t,h}^{i,j}$ is the jth data sample randomly drawn from client i's local dataset $\mathcal{D}_i$. After H local iterations, client i sends her local update $\mathbf{w}_{t,H}^{i}$ for round t to the server.

The computation effort D_i represents the mini-batch size client i uses to update her local model in each round. Due to the randomness of data sampling for computing the update in SGD, the computed gradient of a client could deviate from the expected gradient, and thus slow down the convergence of the FL global model. It has been proved that the larger the mini-batch size D_i, the lower the variance of her local update [105]. Thus, a local update computed with a larger mini-batch size benefits the FL training.

At the end of round t, the server aggregates clients' local models and updates the global model as

$$\mathbf{w}_t = \sum_{i=1}^{N} p_i \mathbf{w}_{t,H}^{i}.$$

Effort Assignment. Before data labeling and local computation, the server assigns the labeling effort e_i' and computation effort D_i' to each client i and notifies client i of e_i' and D_i'. The labeling effort $e_i' \in \{0, 1\}$ indicates whether the server desires client i to make effort in labeling, and the computation effort D_i' indicates the mini-batch size that the server desires client i to use to update her local model in each round. Clients' effort assignments generally vary for different clients due to their diverse characteristics (e.g., weight in model aggregation, computation capability).

After being assigned effort e_i', each client i generates labels for the local dataset by making actual effort e_i. Since e_i is a hidden action of client i, it is possible that client i manipulates e_i against the assignment e_i' to her own advantage such that $e_i \neq e_i'$.

Furthermore, a client incurs a computation cost (measured by the computation time, energy consumption, etc.) for computing a local model update, which depends on the computation capability of the client and the mini-batch size used to compute the update. Thus, client i may

also have incentive to manipulate D_i against the assignment D_i' to her own advantage such that $D_i \neq D_i'$.

Local Model Reporting. When reporting the local model to the server, a client i may have incentive to misreport her local model to her own advantage, i.e.,

$$\mathbf{w}_t^i = \mathbf{w}_{t-1} - \gamma_i \eta g_{t-1}^i,$$

where $\gamma_i \geq 0, \forall i \in \mathcal{N}$ is the model reporting coefficient, which is the multiple of gradient that client i uses to update her local model[2]. When $\gamma_i = 1$, client i reports the actual local model to the server, which is desired by the FL server. When $\gamma_i \neq 1$, the gradient is reduced or increased. In this case, the trained model of FL will be affected, and thus the training loss $F(\mathbf{w})$.

7.3.2 Truthful Incentive Mechanism for FL

After the training process, the FL server tests the trained global model of FL to a data sample ξ randomly drawn from a testing dataset $\mathcal{D}_0$. It is commonly assumed in existing studies that the FL server can test the trained FL model (e.g., [131]). Then the server can determine each client's reward based on the testing loss $f(\mathbf{w}_T, \xi)$ observed for the testing data sample ξ. Note that the server only needs to test the trained model to a single random data sample from $\mathcal{D}_0$. For example, the testing can be performed when the server applies the trained model to an unseen data sample for inference/prediction and observes its true label later.

Based on the testing loss $f(\mathbf{w}_T, \xi)$, the server pays a reward r_i to each client i, according to a certain reward function:

$$r_i(e_i', \boldsymbol{e}_{-i}', D_i', \boldsymbol{D}_{-i}', \gamma_i', \boldsymbol{\gamma}_{-i}', f(\mathbf{w}_T, \xi)), \tag{7.2}$$

where $\boldsymbol{e}_{-i}'$, $\boldsymbol{D}_{-i}'$, and $\boldsymbol{\gamma}_{-i}'$ are other clients' assigned data labeling and computation effort, and model reporting coefficient, respectively. The reward function is pre-defined by the server, and announced to all clients before they participate in FL. We can see that the reward depends on

[2]In this chapter, we assume that clients' strategies do not change in FL training. We will study the case of time-varying strategies in future work.

not only the assigned efforts and model reporting coefficient, but also the testing loss of the final global model.

Each client i's payoff is the difference between the reward paid by the server and her cost in data labeling and computing her local model, given by

$$\begin{aligned} &u_i(e_i, \boldsymbol{e}', D_i, \boldsymbol{D}', \gamma_i, \boldsymbol{\gamma}') \triangleq \\ &r_i(e_i', \boldsymbol{e}'_{-i}, D_i', \boldsymbol{D}'_{-i}, \gamma_i', \boldsymbol{\gamma}'_{-i}, f(\mathbf{w}_T, \xi)) - c_l^i e_i - \sum_{t=1}^{T} c_p^i D_i, \end{aligned} \tag{7.3}$$

where $\boldsymbol{e}'$, $\boldsymbol{D}'$, and $\boldsymbol{\gamma}'$ are all clients' assigned data labeling effort, computation effort, and model reporting coefficient, respectively. The data labeling cost coefficient c_l^i captures the resources consumed by client i if she makes an effort, i.e., $e_i = 1$, in data labeling, and the computation cost coefficient c_p^i is client i's cost of computing her local update using one data sample. If client i makes no effort, i.e., $e_i = 0$, there incurs no cost. Here we assume that clients have the same data labeling cost coefficient (i.e., $c_l = c_l^i, \forall i \in \mathcal{N}$), and the labeling and computation cost coefficients are known to the server. This assumption is reasonable when the costs of labeling a client's dataset and computing using a data sample are determined by uniform market prices (e.g., in Amazon Mechanical Turk, a usual reward for labeling an image is \$0.1). We can also relax the restriction of the uniform labeling cost coefficient. Since a client i can only affect the training loss through her actual e_i, D_i, and γ_i, we omit the loss function $f(\mathbf{w}_T, \xi)$ in the expression of client i's utility u_i. The detailed reward function design will be given in Section 7.5.

The server's payoff u_0 is the negative training loss minus the total reward paid to the clients, i.e.,

$$u_0(\boldsymbol{e}', \boldsymbol{D}', \boldsymbol{\gamma}', f(\mathbf{w}_T, \xi)) \triangleq -f(\mathbf{w}_T, \xi) - \sum_{i \in \mathcal{N}} r_i. \tag{7.4}$$

Since clients may manipulate their actual efforts and report untruthful local models, the global model may be different from that when clients do not behave truthfully, i.e.,

$$\mathbf{w}_T|_{\boldsymbol{e}', \boldsymbol{D}', \boldsymbol{\gamma}'} \neq \mathbf{w}_T|_{\boldsymbol{e}, \boldsymbol{D}, \boldsymbol{\gamma}}.$$

This means that the final global model obtained with efforts and reported local model manipulation cannot solve the FL problem given in (7.1). Nevertheless, the training loss of FL is also affected, i.e.,

$$F(\mathbf{w}_T)|_{\boldsymbol{e}',\boldsymbol{D}',\boldsymbol{\gamma}'} \neq F(\mathbf{w}_T)|_{\boldsymbol{e},\boldsymbol{D},\boldsymbol{\gamma}}.$$

Furthermore, some clients' manipulation would discourage other clients to participate in FL. For the reasons we discussed above, here we aim to design a mechanism that can incentivize clients to make data labeling and computation efforts as the server desired and upload their actual local models. This can be achieved by properly defining the reward function r_i. The truthful mechanism should have the following features:

Definition 7.1 A mechanism achieves truthful strategies of all clients as a *Nash equilibrium* (NE) if, given that all other clients truthfully make data labeling and computation effort as the server desired and upload their actual local models, the best strategy for client i to maximize her payoff is to behave truthfully, i.e.,

$$\begin{aligned} &E[u_i(e_i', \boldsymbol{e}'_{-i}, D_i', \boldsymbol{D}'_{-i}, \gamma_i', \boldsymbol{\gamma}'_{-i})] \geq \\ &\qquad E[u_i(e_i, \boldsymbol{e}'_{-i}, D_i, \boldsymbol{D}'_{-i}, \gamma_i, \boldsymbol{\gamma}'_{-i})], \forall e_i, D_i, \gamma_i. \end{aligned} \tag{7.5}$$

Another aspect we should notice is that the payoff of each client i should be non-negative, so that the client will have the incentive to participate in the FL. This property is formally known as individual rationality as stated below.

Definition 7.2 A mechanism is *individually rational* (IR) if for each client i, its expected payoff is non-negative if she behaves truthfully, i.e.,

$$E[u_i(e_i', \boldsymbol{e}'_{-i}, D_i', \boldsymbol{D}'_{-i}, \gamma_i', \boldsymbol{\gamma}'_{-i})] \geq 0, \forall e_i, D_i, \gamma_i. \tag{7.6}$$

7.4 Training Loss Analysis

In this section, we characterize the performance bounds on the training loss as a function of clients' data labeling effort, local computation effort, and reported local models, which reveal the impacts of these factors on the training loss.

We first make the following general assumptions on the loss functions $F_1, \ldots, F_N$, $\forall i \in \mathcal{N}$.

Assumption 7.1 *$F_1, \ldots, F_N$ are L-smooth.*

Assumption 7.2 *$F_1, \ldots, F_N$ are μ-strongly convex.*

Assumption 7.3 *The variance of the stochastic gradient of a data sample in a device is bounded: $E \left\| \nabla f\left(\mathbf{w}_t, \xi_m^i\right) - \nabla F_i\left(\mathbf{w}_t\right) \right\|^2 \leq \sigma_i^2$, $\forall i \in \mathcal{N}$, $\forall t$.*

Assumption 7.4 *The variance of the stochastic gradient of a data sample when the client makes no effort on labeling is bounded: $E \left\| \nabla f\left(\mathbf{w}_t, \xi_m^i\right) - \nabla f\left(\mathbf{w}_t, \xi_m^{i\,\prime}\right) \right\|^2 \leq \beta$, $\forall i \in \mathcal{N}$, $\forall t$.*

Assumption 7.5 *The expected squared norm of stochastic gradients is bounded: $E \left\| \nabla F_i\left(\mathbf{w}_t\right) \right\|^2 \leq G^2$, $\forall i \in \mathcal{N}$, $\forall t$.*

In Assumption 4, we assume that the variance of the stochastic gradient of a data sample when the client makes no labeling effort is upper bounded, and the bound β is known by the server. The server can calculate the bound using the loss function and the range of the value of data samples. Next, we use a simple example to demonstrate how to obtain the bound β. We use a simple linear regression model to illustrate the convergence problem. Assume that the loss function is given by

$$f\left(\mathbf{w}, \xi_m^i\right) = \frac{1}{2} \| \boldsymbol{x}_m^i \boldsymbol{w} - y_m^i \|^2, \quad \forall i \in \mathcal{N}.$$

A data sample with correct label is denoted as $\xi_m^i = (\boldsymbol{x}_m^i, y_m^i)$. A data sample with incorrect label is denoted as $\xi_m^{i\,\prime} = (\boldsymbol{x}_m^i, y_m^{i\,\prime})$.

The variance of the stochastic gradient of a data sample is

$$\begin{aligned} & E \left\| \nabla f\left(\mathbf{w}, \xi_m^i\right) - \nabla f\left(\mathbf{w}, \xi_m^{i\,\prime}\right) \right\|^2 \\ =& \| (\boldsymbol{x}_m^i \boldsymbol{w} - y_m^i) \boldsymbol{x}_m^i - (\boldsymbol{x}_m^i \boldsymbol{w} - y_m^{i\,\prime}) \boldsymbol{x}_m^i \|^2 \\ =& \| (y_m^{i\,\prime} - y_m^i) \boldsymbol{x}_m^i \|^2 \leq 2YX, \end{aligned}$$

where ${y_m^i}^2 \leq Y$ and $\|\boldsymbol{x}_m^i\|^2 \leq X$. Then we have $\beta = 2YX$.

Theorem 7.1 Suppose Assumptions 1 to 5 hold, and the step size $\eta \leq \frac{1}{2L}$. Then the FL training loss is bounded above by:

$$\begin{aligned} &E[F(\mathbf{w}_T) - F(\mathbf{w}^*)] \leq L(1-\mu\eta)^{TH} E\left\|\mathbf{w}_0 - \mathbf{w}^*\right\|^2 \\ &+ 2L\eta^2 \sum_{t=1}^{T}\sum_{h=1}^{H}(1-\mu\eta)^{TH-(t-1)H-h} \\ &\sum_{i\in\mathcal{N}}\Bigg({p_i}^2\frac{\sigma_i^2}{D_i} + 6Lp_id_i + p_i(1-e_i)\beta \\ &+2p_i\left((\gamma_i-1)^2 + (H-1)^2\right)\left(G^2 + \frac{\sigma_i^2}{D_i} + (1-e_i)\beta\right)\Bigg), \end{aligned} \tag{7.7}$$

where $d_i \triangleq E[F_i(\mathbf{w}^*)] - E[F_i(\mathbf{w}_i^*)]$ quantifies the heterogeneity degree of the data held by client i [15].

The first term of the training loss bound decreases geometrically with the total number of local iterations TH, and is due to that SGD in expectation makes progress towards the optimal solution. The bound is also affected by other factors, i.e., the randomness of data sampling in SGD for computing local updates $p_i^2\frac{\sigma_i^2}{D_i}$, the data heterogeneity of clients' data p_id_i, the data labeling effort level of each client $p_i(1-e_i)\beta$, the local model misreporting γ_i, and the number of local iterations per round H. We can see that any $\gamma_i \neq 1$ increases the training loss bound. Thus, it is desired that all clients report their actual local model (i.e., $\gamma_i = 1$, $\forall i \in \mathcal{N}$) to minimize the training loss. Moreover, as the coefficients in the training loss bound depend on the aggregation weight p_i, a client with a higher weight p_i has a larger impact on the training loss than that with a lower weight p_i.

The randomness of data sampling in SGD for computing local updates affects the training loss, which depends on each clients' mini-batch size D_i in each iteration (i.e., computation effort). We can observe that a larger mini-batch size D_i reduces the training loss. The terms involving e_i depend on by the data labeling effort of each client. If client i makes effort in data labeling, these terms equal to 0; otherwise, if client i makes no effort in data labeling, these

terms equal to $p_i\beta$. Thus, it is desirable that all clients make data labeling effort (i.e., $e_i = 1$, $\forall i \in \mathcal{N}$) to minimize the training loss.

7.5 Truthful Incentive Mechanisms for Data Labeling Effort, Local Computation Effort, and Local Model Elicitation

In this section, we first propose the LCEME mechanism that satisfy the truthful and IR properties to incentivize clients to make efforts as the server desired and report actual local models. Then, we find the optimal computation effort assignment under the LCEME mechanism that maximizes the server's payoff.

7.5.1 LCEME Mechanism Design

We first present the design of LCEME mechanism.

Definition 7.3 Given the data labeling effort assignment $e_i' = 1$, the model reporting coefficient assignment $\gamma_i' = 1$, and any computation effort assignment D_i', the LCEME mechanism's reward function for client i, $\forall i \in \mathcal{N}$, is given by

$$\begin{aligned} &r_i(e_i', \boldsymbol{e}_{-i}', D_i', \boldsymbol{D}_{-i}', \gamma_i', \boldsymbol{\gamma}_{-i}', f(\mathbf{w}_T, \xi)) \\ &= \Omega(\boldsymbol{D}') - \Phi(D_i') f(\mathbf{w}_T, \xi) + c_l, \end{aligned} \tag{7.8}$$

where

$$\Omega(\boldsymbol{D}') = \Phi(D_i') \Big(L(1-\mu\eta)^{TH} E \left\| \mathbf{w}_0 - \mathbf{w}^* \right\|^2 + A \sum_{i \in \mathcal{N}} (6Lp_i d_i + {p_i}^2 \frac{\sigma_i^2}{D_i'} + 2p_i (H-1)^2 (G^2 + \frac{\sigma_i^2}{D_i'})) \Big) + T c_p^i D_i',$$

$\boldsymbol{e}' = \mathbf{1}^{1\times N}$, $\boldsymbol{\gamma}' = \mathbf{1}^{1\times N}$, $\Phi(D_i') = \frac{{D_i'}^2 c_p^i T}{A\sigma_i^2 p_i (p_i + 2(H-1)^2)}$, $A = 2L\eta \frac{1-(1-\mu\eta)^{TH}}{\mu}$, and the assigned computation effort D_i' satisfies $D_i' \geq \sigma_i \sqrt{\frac{c_l p_i (p_i + 2(H-1)^2)}{\beta c_p^i T (1 + 2(H-1)^2)}}$.

Note that the reward function depends on the testing loss which is observed by the server. In this paper, for ease of exposition, we assume that the expected testing loss is equal to the training loss. This assumption is reasonable: in practice, the entire training dataset of FL (i.e., $\cup_{i=1}^{N} \mathcal{D}_i$) is often a good representation of the testing dataset $\mathcal{D}_0$, so that the expected testing

loss is well approximated by the training loss. Based on this assumption, the expected payoff of client i is given by:

$$\begin{aligned} & E_{\xi}[u_i(e_i, \boldsymbol{e}'_{-i}, D_i, \boldsymbol{D}'_{-i}, \gamma_i, \boldsymbol{\gamma}'_{-i})] \\ & = E_{\xi}[r_i(e'_i, \boldsymbol{e}'_{-i}, D'_i, \boldsymbol{D}'_{-i}, \gamma'_i, \boldsymbol{\gamma}'_{-i}, f(\mathbf{w}_T, \xi))] - c_l e_i - T c_p^i D_i \\ & = \Omega(\boldsymbol{D}') - \Phi(D'_i) F(\mathbf{w}_T) + c_l - c_l e_i - T c_p^i D_i \end{aligned} \tag{7.9}$$

where ξ is a random data sample drawn from the testing dataset $\mathcal{D}_0$.

Next, based on Theorem 7.1, we approximate the expected training loss $F(\mathbf{w}_T)$ in terms of the optimal training loss $F(\mathbf{w}^*)$ plus the upper bound on the training loss gap given in the right-hand-side of (7.7). Then we assume that each client uses $\hat{u}_i$ as her expected payoff function, where $\hat{u}_i$ is defined as (7.9) with $F(\mathbf{w}_T)$ replaced by the right-hand-side of (7.7) (the optimal training loss term $F(\mathbf{w}^*)$ is omitted as it does not affect the truthful mechanism design). This is a reasonable assumption since 1) a client cannot find the expected training loss $F(\mathbf{w}_T)$, but can find the upper bound in (7.7); 2) using the upper bound on the training loss gap can capture the worst case of the client's expected payoff. Therefore, in the rest of this paper, each client determines her strategic behavior for maximizing the payoff function $\hat{u}_i$.

In the following, we use two theorems to prove that the LCEME mechanism satisfies the truthful and IR properties defined in Definition 7.1 and 7.2, with respect to the clients' payoff functions $\hat{u}_i$.

Theorem 7.2 The LCEME mechanism is truthful.

We show how the LCEME mechanism achieves the truthful property using three lemmas.

Lemma 7.1 Under the LCEME mechanism, given that client i makes any data labeling effort e_i and computation effort D_i, her optimal reported local model is her true local model, i.e., $\gamma_i = 1$.

It can be shown that the expected payoff of client i is a convex function of γ_i. We can obtain the result of Lemma 7.1 by calculating the partial derivative of the expected payoff of client i with respect to γ_i and letting the derivative equal to 0.

Using Lemma 7.1, we can express client i's approximated expected payoff $\hat{u}_i$ as

$$\begin{aligned}\hat{u}_i(e_i, D_i, D_i') &= \Phi(D_i')A({p_i}^2\frac{\sigma_i^2}{D_i'} + 2p_i(H-1)^2\frac{\sigma_i^2}{D_i'}) + Tc_p^iD_i' \\ &- \Phi(D_i')A\Bigg(p_i^2\frac{\sigma_i^2}{D_i} + p_i(1-e_i)\beta \\ &+2p_i(H-1)^2(\frac{\sigma_i^2}{D_i} + (1-e_i)\beta)\Bigg) + c_l - c_le_i - Tc_p^iD_i.\end{aligned}$$

Lemma 7.2 Under the LCEME mechanism, given that clients report their optimal local models $\gamma_i = 1$, $\forall i \in \mathcal{N}$, and client i makes any computation effort, client i's optimal actual effort is the desired effort, i.e., $e_i = 1$.

Then, we show that, when client i makes any labeling effort, her expected payoff is always lower than that when she makes effort:

$$\begin{aligned}&\hat{u}_i(1, D_i, D_i') - \hat{u}_i(e_i, D_i, D_i') \\ =&\frac{{D_i'}^2c_p^iT(1+2(H-1)^2)}{\sigma_i^2p_i^2(p_i+2(H-1)^2)}p_i(1-e_i)\beta - c_l + c_le_i \\ =&(\frac{{D_i'}^2c_p^iT(1+2(H-1)^2)\beta}{\sigma_i^2p_i(p_i+2(H-1)^2)} - c)(1-e_i) \\ \geq&(c-c)(1-e_i) \geq 0,\end{aligned}$$

where the inequality follows from the constraint on D_i'.

Using Lemma 7.1 and Lemma 7.2, we can express client i's approximated expected payoff $\hat{u}_i$ as

$$\begin{aligned}\hat{u}_i(D_i, D_i') = &- \Phi(D_i')A(p_i^2\frac{\sigma_i^2}{D_i} + 2p_i(H-1)^2\frac{\sigma_i^2}{D_i}) - Tc_p^iD_i \\ &+ \Phi(D_i')A({p_i}^2\frac{\sigma_i^2}{D_i'} + 2p_i(H-1)^2\frac{\sigma_i^2}{D_i'}) + Tc_p^iD_i'.\end{aligned}$$

Lemma 7.3 Given that clients report their optimal local models $\gamma_i = 1$ and make effort in data labeling $e_i = 1$, $\forall i \in \mathcal{N}$, client i's optimal actual computation effort is the desired computation effort, i.e., $D_i = D_i'$.

Now that the expected payoff is a convex function of client i's actual computation effort D_i, we can obtain client i's optimal actual computation effort D_i by calculating the partial

derivative of the expected payoff of client i with respect to D_i and letting the derivative equal to 0, which is the desired computation effort D_i'.

Given the definition of truthful mechanisms (Definition 7.1), the LCEME mechanism is truthful. □

Theorem 7.3 The LCEME mechanism is IR.

Here we discuss the rationale of the LCEME mechanism. The server's goal is to incentivize clients to make actual data labeling and computation effort as desired by the server and report their true local models. Thus, client i's reward function r_i should be a function of her actual efforts (e_i and D_i) and model report coefficient (γ_i). Otherwise, clients can deceive the server to gain more rewards. Thus, we design the reward function as a function of the training loss, which has been proved to be determined by clients' actual efforts and model reporting strategies in Theorem 7.1. In the refined reward function, client i's optimal strategy to maximize her expected payoff is to make data labeling and computation efforts as desired by the server and report her actual local model.

7.5.2 Optimal Computation Effort Assignment

A desirable objective for the server is to find the optimal assignment that maximizes her expected payoff.

Definition 7.4 The server's optimal assignment $\boldsymbol{D}^*$ for LCEME mechanism is the assignment function $\boldsymbol{D}'$ that maximizes the server's payoff, i.e.,

$$\begin{aligned} &\boldsymbol{D}^* \triangleq \arg\max_{\boldsymbol{D}'} E[u_0(\boldsymbol{D}', f(\mathbf{w}_T, \xi))] \\ &s.t.\ D_i^* \geq \sqrt{\frac{c_l \sigma_i^2 p_i (p_i + 2(H-1)^2)}{\beta c_p^i T (1 + 2(H-1)^2)}}, \forall i \in \mathcal{N}. \end{aligned} \tag{7.10}$$

The constraint in (7.10) is to make sure that the LCEME mechanism is truthful.

The problem given in (7.10) is equivalent to the problem:

$$\begin{aligned} \boldsymbol{D}^* \triangleq \arg\min_{\boldsymbol{D}'} \left\{ F(\mathbf{w}_T) - F(\mathbf{w}^*) + \sum_{i\in\mathcal{N}} r_i \right\}, \\ s.t.\ D_i^* \geq \sqrt{\frac{c_l \sigma_i^2 p_i (p_i + 2(H-1)^2)}{\beta c_p^i T (1 + 2(H-1)^2)}}, \forall i \in \mathcal{N}, \end{aligned} \tag{7.11}$$

where $F(\mathbf{w}^*)$ can be seen as a constant.

From the above problem formulation, we observe that there exists a tradeoff between the FL training loss and the server's payment to clients. We know that the training loss reduces when clients use larger mini-batch sizes to compute their local updates from Theorem 7.1. However, using larger mini-batch sizes increases the server's payment. Therefore, we aim to find the optimal computation effort (in the form of mini-batch size) assignment for each client to maximize the server's payoff.

Theorem 7.4 The server's optimal computation effort allocation is given by

$$D_i^* = \max \left\{ \sqrt{\frac{A(p_i^2\sigma_i^2 + 2p_i(H-1)^2)}{c_p^i T}}, \sqrt{\frac{c_l \sigma_i^2 p_i (p_i + 2(H-1)^2)}{\beta c_p^i T (1 + 2(H-1)^2)}} \right\}, \forall i \in \mathcal{N}.$$

From Theorem 7.4, we can see that the server's optimal computation effort for a client i increases with her weight p_i and gradient variance σ_i^2. This is because when client i has a larger p_i and/or σ_i^2, the effect of the randomness of her SGD computation per data sample on the global model will be larger. From Theorem 7.1, we know that a larger mini-batch size D_i reduces the randomness of data sampling in SGD. Thus, assigning a larger computation effort for client i can reduce the training loss. We also see that D_i^* decreases as client i's computation cost c_p^i increases. This is because a larger computation cost increases the reward paid by the server. When a client's computation cost is large, the server prefers to allocate a smaller mini-batch size to the client to reduce the payment. We can also show that a client's optimal mini-batch size increases as the number of local iterations H increases. This is because a local update's

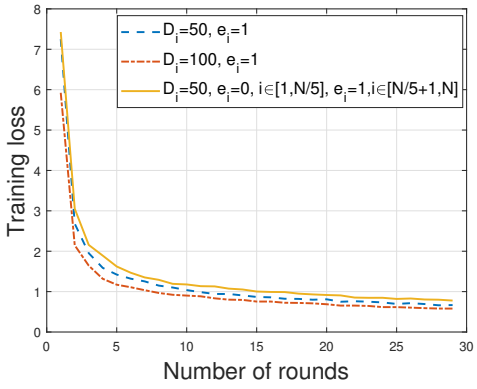

Figure 7.2: Impact of effort level on the training loss.

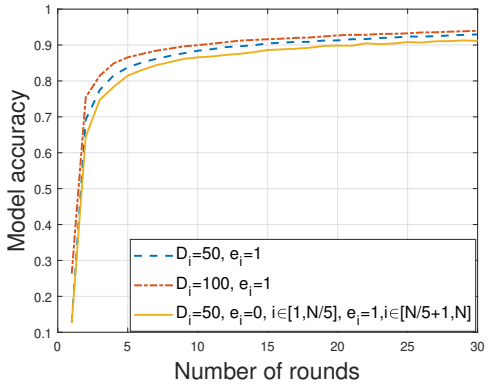

Figure 7.3: Impact of effort level on the model accuracy.

quality can be improved by using a larger mini-batch size, and thus reduce the error caused by performing multiple local iterations.

7.6 Simulation Results

In this section, we conduct real data based simulations to validate the theoretical findings and evaluate the LCEME mechanism. We first describe the simulation setups, and then we present the evaluation results and analyses.

We implement a simulated system consisting of a server and 10 clients. We use the widely used MNIST dataset [141] for simulations in Matlab. Each training element is a handwritten digit picture that represents numbers from 0 to 9. Each client conducts one layer of CNN for one local iteration in each round ($H = 1$). We denote the heterogeneity degree of a client's dataset as the percentage of data with labels the same as the last digit of the client's index.

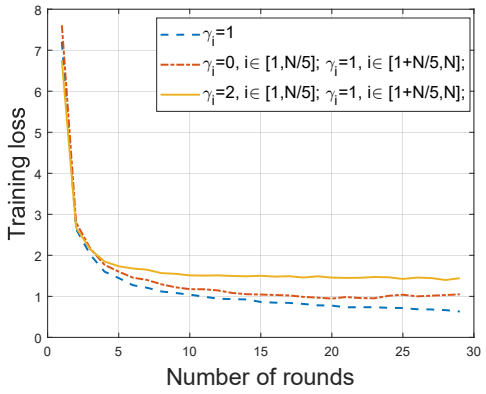

Figure 7.4: Impact of model reporting coefficient on the training loss.

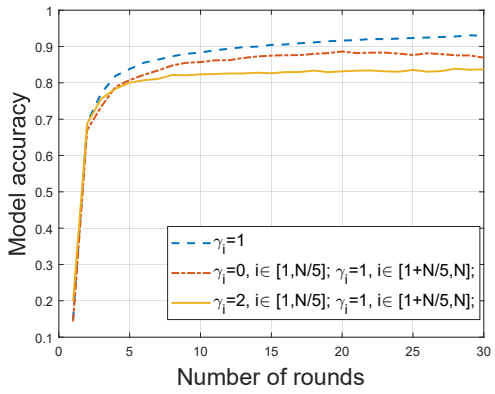

Figure 7.5: Impact of model reporting coefficient on the model accuracy.

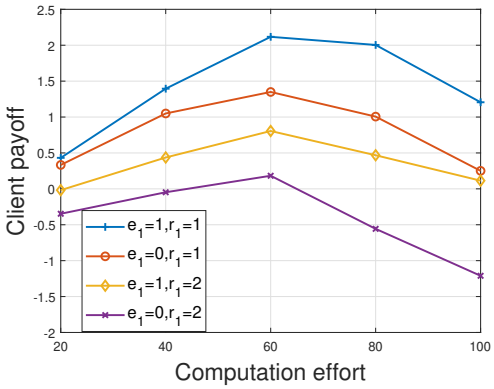

Figure 7.6: Impact of clients' behavior on the payoff.

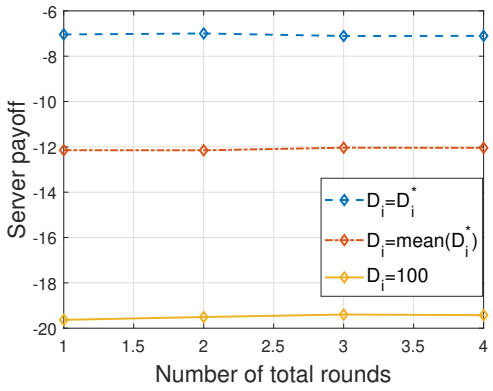

Figure 7.7: Impact of computation effort allocation on server's payoff.

For the remaining data of the client, we uniformly draw the training data samples from the entire training set. Unless otherwise specified, client i's heterogeneity degree is 0.4, and the mini-batch size is $D_i = 50$.

7.6.1 Impact of Clients' Strategies on Training Loss

We first compare the training loss while clients' data labeling and computation efforts changes. From Figs. 7.2 and 7.3, we can see that the training loss decreases and the model accuracy increases as D_i increases. We also observe that when there exist clients make no effort in data labeling, the training loss increases and the model accuracy decreases. The observations conform to our theoretical result in Theorem 7.1. We also compare the training loss while clients report local models with different model reporting coefficients and truthfully make efforts. We observe from Figs. 7.4 and 7.5 that the training loss is minimized when all clients report their actual local model. When there exist clients report local model untruthfully, the training loss increases and the model accuracy decreases. This conforms to the result in Theorem 7.1 that the more clients truthfully report local models, the lower the training loss. We also observe that, although the training loss bounds are the same when $\gamma_i = 0$ and $\gamma_i = 1$, the training loss is lower when $\gamma = 0$. Figs. 7.2, 7.3, 7.4, and 7.5 demonstrates that, when clients truthfully make efforts and report local models, the training loss is minimized and the model accuracy is maximized.

7.6.2 Impact of Truthfulness on Clients' Payoff

We compare a client's payoff while making the desired data labeling effort $e_1 = 1$ or not $e_1 = 0$, and reporting the actual local model $\gamma_1 = 1$ or not $\gamma_1 \neq 1$, as the computation effort D_1 changes. The assigned computation effort $D_1' = 60$. We let other clients behave truthfully. We observe from Fig. 7.6 that a client's payoff, when she makes data labeling and computation effort as the server desired and reports actual local model, is always higher than that when her behavior is untruthful. Furthermore, we also observe that the client's payoff is positive when she behave truthfully. The simulation results demonstrate that the LCEME mechanism is truthful and achieves the IR property.

7.6.3 Server's Payoff

We compare the server's payoff while clients make different computation efforts. From Fig. 7.7, we can see that the server's payoff is maximized when clients make the server's optimal computation effort. When clients do not make the optimal computation effort, the server's payoff is lower even if the total computation effort of clients is the same as the optimal computation effort allocation. This is because, in the former case, the computation effort allocation does not care about clients' heterogeneous computation cost and thus causes higher computation costs. We also simulate the case where clients' computation effort $D_i = 100$ is always higher than the optimal computation effort. We observe that among three cases, this case results in the lowest server's payoff. This is because clients' computation costs are ignored when assigning D_i, resulting in an increase in the server cost.

7.7 Conclusion

In this paper, we studied FL with crowdsourced data labels, where the local data of each participating client are labeled manually by the client. We characterized the performance bounds on the training loss as a function of clients' data labeling effort, local computation effort, and reported local models. We then devised truthful incentive mechanisms which motivate strategic

clients to make truthful efforts as desired by the server in data labeling and local model computation, and also report true local models to the server based on the derived performance bound. Simulations based on real data are demonstrated the efficacy of the proposed algorithms.

7.8 Appendix

7.8.1 Proof of Theorem 7.1

We define a virtual sequence $\bar{\mathbf{w}}_{t,h}$, given by $\bar{\mathbf{w}}_{t,h} = \sum_{i\in\mathcal{N}} p_i \mathbf{w}^i_{t,h}, \forall t, h$. Note that $\bar{\mathbf{w}}_{t,h}$ is not accessible when clients have not completed H local iterations (i.e., $h < H$), and $\mathbf{w}_t = \bar{\mathbf{w}}_{t,H}$.

$$\begin{aligned}\|\bar{\mathbf{w}}_{t,H} - \mathbf{w}^*\|^2 = \left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^* - \eta \sum_{i\in\mathcal{N}} \gamma_i p_i {g^i_{t,H-1}}'\right\|^2 \leq \\ 2\underbrace{\|\bar{\mathbf{w}}_{t,H-1} - \eta\bar{g}_{t,H-1} - \mathbf{w}^*\|^2}_{A_1} + 2\underbrace{\left\|\eta\bar{g}_{t,H-1} - \eta\sum_{i\in\mathcal{N}} \gamma_i p_i {g^i_{t,H-1}}'\right\|^2}_{A_2}\end{aligned} \tag{7.12}$$

where ${g^i_{t,h}}'$ is the gradient when client i makes any data labeling effort, $\bar{g}_{t,h} \triangleq \sum_{i\in\mathcal{N}} p_i \bar{g}^i_{t,h} \triangleq \sum_{i\in\mathcal{N}} p_i E[g^i_{t,h}]$, and $g^i_{t,h}$ is the gradient when client i makes data labeling effort.

$$A_1 = \|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*\|^2 + \underbrace{\eta^2\|\bar{g}_{t,H-1}\|^2}_{B_1} - \underbrace{2\eta\left\langle\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*, \bar{g}_{t,H-1}\right\rangle}_{B_2}. \tag{7.13}$$

For B_2, we have

$$B_2 = -2\eta\sum_{i\in\mathcal{N}} p_i\langle\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^i_{t,H-1}, \bar{g}^i_{t,H-1}\rangle - 2\eta\sum_{i\in\mathcal{N}} p_i\langle\mathbf{w}^i_{t,H-1} - \mathbf{w}^*, \bar{g}^i_{t,H-1}\rangle.$$

We use the convexity of $\|\cdot\|^2$ and the L-smoothness of F_i to bound B_1, the Cauchy-Schwarz inequality and AM-GM inequality to bound the first term of B_2, and the μ-strong convexity of

F_i to bound the second term of B_2. We have

$$
\begin{aligned}
A_1 \leq & \left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*\right\|^2 + 2L\eta^2 \sum_{i\in\mathcal{N}} p_i (F_i(\mathbf{w}^i_{t,H-1}) - F_i(\mathbf{w}^*_i)) \\
& + \sum_{i\in\mathcal{N}} p_i \left(\left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^i_{t,H-1}\right\|^2 + \eta^2 \left\|\bar{g}^i_{t,H-1}\right\|^2 \right) \\
& - 2\eta \sum_{i\in\mathcal{N}} p_i \left(F_i(\mathbf{w}^i_{t,H-1}) - F_i(\mathbf{w}^*) + \frac{\mu}{2} \left\|\mathbf{w}^i_{t,H-1} - \mathbf{w}^*\right\|^2 \right) \\
\leq & \left(1 - \mu\eta\right) \left\|\mathbf{w}_{t,H-1} - \mathbf{w}^*\right\|^2 + \sum_{i\in\mathcal{N}} p_i \left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^i_{t,H-1}\right\|^2 \\
& + 4L\eta^2 \sum_{i\in\mathcal{N}} p_i (F_i(\mathbf{w}^i_{t,H-1}) - F_i(\mathbf{w}^*_i)) \\
& - 2\eta \sum_{i\in\mathcal{N}} p_i \left(F_i(\mathbf{w}^i_{t,H-1}) - F_i(\mathbf{w}^*) \right),
\end{aligned}
$$

in which we denote the last two lines as C_1.

$$
\begin{aligned}
C_1 = & 4L\eta^2 \sum_{i\in\mathcal{N}} p_i (F_i(\mathbf{w}^*) - F_i(\mathbf{w}^*_i)) \\
& - 2\eta(1 - 2L\eta) \sum_{i\in\mathcal{N}} p_i (F_i(\mathbf{w}^i_{t,H-1}) - F_i(\mathbf{w}^*)) \\
\leq & 4L\eta^2 \sum_{i\in\mathcal{N}} p_i d_i - 2\eta(1 - 2L\eta) \Bigg(- \sum_{i\in\mathcal{N}} p_i \\
& \left(\eta L \left(F_i(\bar{\mathbf{w}}_{t,H-1} - F_i(\mathbf{w}^*_i)) + \frac{1}{2\eta} \left\|\mathbf{w}^i_{t,H-1} - \bar{\mathbf{w}}_{t,H-1}\right\|^2 + F_i(\bar{\mathbf{w}}_{t,H-1}) - F_i(\mathbf{w}^*)\right)\right) \Bigg). \\
\leq & 2\eta(1 - 2L\eta)(\eta L - 1) \sum_{i\in\mathcal{N}} p_i \left(F_i(\bar{\mathbf{w}}_{t,H-1} - F_i(\mathbf{w}^*_i))\right) \\
& + (4L\eta^2 + 2L\eta^2(1 - 2L\eta)) \sum_{i\in\mathcal{N}} p_i d_i + (1 - 2L\eta) \sum_{i\in\mathcal{N}} p_i \left\|\mathbf{w}^i_{t,H-1} - \bar{\mathbf{w}}_{t,H-1}\right\|^2 \\
\leq & 6L\eta^2 \sum_{i\in\mathcal{N}} p_i d_i + \sum_{i\in\mathcal{N}} p_i \left\|\mathbf{w}^i_{t,H-1} - \bar{\mathbf{w}}_{t,H-1}\right\|^2.
\end{aligned}
$$

Thus we can further bound A_1 as

$$
\begin{aligned}
E[A_1] \leq & \left(1 - \mu\eta\right) \left\|\bar{\mathbf{w}}_{t,H-1} - \mathbf{w}^*\right\|^2 \\
& + 6L\eta^2 \sum_{i\in\mathcal{N}} p_i d_i + 2 \sum_{i\in\mathcal{N}} p_i \left\|\mathbf{w}^i_{t,H-1} - \bar{\mathbf{w}}_{t,H-1}\right\|^2.
\end{aligned}
\tag{7.14}
$$

Next, we bound

$$\begin{aligned}&\sum_{i\in\mathcal{N}} p_i E\left\|\bar{\mathbf{w}}_{t,h}-\mathbf{w}_{t,h}^i\right\|^2 \leq \sum_{i\in\mathcal{N}} p_i E\left\|\mathbf{w}_{t,h}^i-\mathbf{w}_{t,1}\right\|^2\\ \leq&\eta^2\sum_{i\in\mathcal{N}} p_i E\|\sum_{h=1}^{H-1} {g_{t,h}^i}'\|^2 \leq \eta^2(H-1)\sum_{i\in\mathcal{N}} p_i \sum_{h=1}^{H-1} E\|{g_{t,h}^i}'\|^2.\end{aligned} \tag{7.15}$$

Using Assumption 7.4, we have

$$\begin{aligned}&E\|g_{t,h}^i-{g_{t,h}^i}'\|^2 = E\|\frac{1}{D_i}\sum_j(\nabla f_i(\mathbf{w}_{t,h},\xi_t^{i,j})-\nabla f_i(\mathbf{w}_{t,h},\xi_t^{i,j'}))\|^2\\ \leq&\frac{1}{D_i}\sum_j E_{\xi_t^{i,j'}|\xi_t^{i,j}}\left[\|(\nabla f_i(\mathbf{w}_{t,h},\xi_t^{i,j})-\nabla f_i(\mathbf{w}_{t,h},\xi_t^{i,j'}))\|^2\right]\\ \leq&(1-e_i)\beta.\end{aligned} \tag{7.16}$$

From [105], we have

$$E\left\|\bar{g}_{t,h}^i-g_{t,h}^i\right\|^2 \leq \frac{\sigma_i^2}{D_i}. \tag{7.17}$$

From (7.16), (7.17), and Assumption 5, we have

$$\begin{aligned}&E\left\|{g_{t,h}^i}'\right\|^2 = E\left\|{g_{t,h}^i}'-g_{t,h}^i+g_{t,h}^i-\bar{g}_{t,h}^i+\bar{g}_{t,h}^i\right\|^2\\ \leq&2E\left\|{g_{t,h}^i}'-g_{t,h}^i\right\|^2+2E\left\|g_{t,h}^i-\bar{g}_{t,h}^i\right\|^2+2E\left\|\bar{g}_{t,h}^i\right\|^2\\ \leq&2(1-e_i)\beta+\frac{2\sigma_i^2}{D_i}+2G^2.\end{aligned} \tag{7.18}$$

Thus we can bound (7.15) as

$$\begin{aligned}&\sum_{i\in\mathcal{N}} p_i E\left\|\bar{\mathbf{w}}_{t,h}-\mathbf{w}_{t,h}^i\right\|^2\\ \leq&2\eta^2(H-1)^2\sum_{i\in\mathcal{N}} p_i((1-e_i)\beta+\frac{\sigma_i^2}{D_i}+G^2).\end{aligned} \tag{7.19}$$

Next, we bound A_2. From (7.17) and (7.18), we have

$$\begin{aligned}
E[A_2] =& \left\| \eta \bar{g}_{t,H-1} - \eta \sum_{i \in \mathcal{N}} \gamma_i p_i {g^i_{t,H-1}}' \right\|^2 \\
=& \eta^2 E \left\| \bar{g}_{t,H-1} - g_{t,H-1} + g_{t,H-1} + \sum_{i \in \mathcal{N}} \gamma_i p_i {g^i_{t,H-1}}' \right\|^2 \\
\leq& 2\eta^2 E \left\| \bar{g}_{t,H-1} - g_{t,H-1} \right\|^2 + 2\eta^2 E \left\| g'_{t,H-1} - g_{t,H-1} \right\|^2 \\
&+ 2\eta^2 \sum_{i \in \mathcal{N}} p_i (\gamma_i - 1)^2 E \left\| {g^i_{t,H-1}}' \right\|^2 \\
\leq& 2\eta^2 \sum_{i \in \mathcal{N}} ({p_i}^2 \frac{\sigma_i^2}{D_i} + p_i(1 - e_i)\beta \\
&+ 2p_i(\gamma_i - 1)^2 (G^2 + \frac{\sigma_i^2}{D_i} + (1 - e_i)\beta)).
\end{aligned} \tag{7.20}$$

Combining (7.12), (7.14), (7.19), and (7.20), we have

$$\begin{aligned}
& E \left\| \mathbf{w}_{T,H} - \mathbf{w}^* \right\|^2 \\
\leq& 2(1 - \mu\eta) \left\| \mathbf{w}_{T,H-1} - \mathbf{w}^* \right\|^2 + 12L\eta^2 \sum_{i \in \mathcal{N}} p_i d_i \\
&+ 4\eta^2 \sum_{i \in \mathcal{N}} ({p_i}^2 \frac{\sigma_i^2}{D_i} + p_i(1 - e_i)\beta) \\
&+ 4\eta^2 \sum_{i \in \mathcal{N}} p_i ((\gamma_i - 1)^2 + 2(H - 1)^2)(G^2 + \frac{\sigma_i^2}{D_i} + (1 - e_i)\beta).
\end{aligned}$$

Using induction and the smoothness of F, we have (7.7).

7.8.2 Proof of Theorem 7.3

Given that all users behave truthfully, the expected payoff of user i, $\forall i$ is given by

$$\begin{aligned}
E[u_i] =& \Omega(\boldsymbol{D}') - \Phi(D'_i) F(\mathbf{w}_T) + c_l - c_l e'_i - T c^i_p D'_i. \\
\geq& \Phi(D'_i)(F(\mathbf{w}_T) - F(\mathbf{w}^*)) + T c^i_p D'_i \\
&- \Phi(D'_i)(F(\mathbf{w}_T) - F(\mathbf{w}^*)) + c_l - c_l e'_i - T c^i_p D'_i = 0.
\end{aligned}$$

7.8.3 Proof of Theorem 7.4

The total expected reward paid by the server is bounded by $\sum_{i\in\mathcal{N}} r_i \geq \sum_{i\in\mathcal{N}}(c_l + Tc_p^i D_i)$. Using (7.7), we have

$$F(\mathbf{w}_T) - F(\mathbf{w}^*) + \sum_{i\in\mathcal{N}} r_i \leq L(1-\mu\eta)^{TH} E \left\|\mathbf{w}_0 - \mathbf{w}^*\right\|^2$$
$$+ \sum_{i\in\mathcal{N}}(A{p_i}^2\frac{\sigma_i^2}{D_t^i} + 6Lp_i d_i + 2p_i(H-1)^2\frac{\sigma_i^2}{D_t^i}) + c_l + Tc_p^i D_i).$$

It can be shown that the above upper bound is a convex function of D_i. The optimal mini-batch size ${D_i}^*$ can be obtained by calculating the partial derivative of the bound with respect to D_i and letting the derivative equals to 0.

Chapter 8

Summary and Future Works.

8.1 Summary

In this dissertation, we have studied quality-aware data crowdsourcing and federated learning in wireless networks. For quality-aware data crowdsourcing, we study truthful incentive mechanisms that elicit workers' private information with privacy-preserving property, and data poisoning attacks on dynamic crowdsourcing. For WFL, we first study quality-aware distributed computation for FL, which controls the quality of users' local updates via the sampling sizes. We study the performance bounds on the training loss as a function of local updates' quality over the training process for different FL setting, and develop cost-effective dynamic distributed learning algorithms that adaptively select participating users and their mini-batch sizes, based on users' costs in wireless networks. We first study IID and non-IID cases for convex setting. Then, we investigate non-convex setting and asynchronous setting with IID data. Given the commonality of data crowdsourcing and labeling quality of the training data of FL, we study the impact of the strategic behavior of FL clients and develop incentive mechanisms that incentivize strategic clients to make truthful efforts as desired by the server in local data labeling and local model computation, and also report true local models to the server.

8.2 Future Works

In this Section, we discuss some possible future work directions following the works in previous chapters.

8.2.1 Non-Convex and Asynchronous FL with Non-IID Data

One direction is to consider non-convex and asynchronous FL where users have non-IID local data. In many practical situations, users' local datasets are non-IID due to various reasons (eg, location, device, user behavior, etc). From the training loss analysis of convex synchronous FL with non-IID data, we observer that users' data heterogeneity slows down the convergence of the global model of FL and can not be eliminated by increasing the mini-batch sizes or decreasing the learning rates. It is worthwhile to study how the training loss of non-convex and asynchronous FL is affected by the data heterogeneity. Furthermore, based on the new training loss bound, the optimal mini-batch size and user selection design for channel-aware adaptive algorithms can be very different from in the IID data set.

8.2.2 Heterogeneous Number of Local Iterations

Another direction is to study the case when users perform multiple numbers of local iterations. Different from the setting where users perform a single iteration of local SGD and send their local updates to the server for global aggregation in each round t, users can perform multiple local iterations of SGD before global aggregation at the server. In this case, the server updates the global model only in the rounds when users send in their local updates. When users perform multiple numbers of local iterations, the training loss is affected by the additional randomness of multiple local updates.

Furthermore, when users perform heterogeneous numbers of local iterations and have non-IID local data, the learned global model may not be consistent with the objective of FL. Due to the heterogeneity in the local learning process at each client, the actual learned FL model may deviate from some users' local optimal model, which is not the purpose of FL. In this situation, it is important to know the difference between the actual learned FL model and the initial FL objective and study how to make use of the bias to achieve better performance.

References

[1] OpenSignal: 3G and 4G LTE Cell Coverage Map. [Online]. Available: https://opensignal.com/

[2] Google Maps. [Online]. Available: https://maps.google.com/

[3] Atmos: Crowd sensing weather platform. [Online]. Available: http://beja.m-iti.org/web/?q=node/1

[4] "Amazon Mechanical Turk is a marketplace for work." [Online]. Available: https://www.mturk.com/

[5] "reCAPTCHA: Easy on humans, tough on bots." [Online]. Available: http://www.google.com/recaptcha

[6] "Crowdin: A localization project management platfom and translation tool." [Online]. Available: https://crowdin.com

[7] B. Zhu, J. Wang, L. He, and J. Song, "Joint transceiver optimization for wireless communication phy using neural network," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1364–1373, 2019.

[8] W. Cui, K. Shen, and W. Yu, "Spatial deep learning for wireless scheduling," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1248–1261, 2019.

[9] Y. Yu, T. Wang, and S. C. Liew, "Deep-reinforcement learning multiple access for heterogeneous wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1277–1290, 2019.

[10] X. Tao, Y. Duan, M. Xu, Z. Meng, and J. Lu, "Learning qoe of mobile video transmission with deep neural network: A data-driven approach," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1337–1348, 2019.

[11] N. Jiang, Y. Deng, A. Nallanathan, and J. A. Chambers, "Reinforcement learning for real-time optimization in nb-iot networks," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1424–1440, 2019.

[12] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186.

[13] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *Siam Review*, vol. 60, no. 2, pp. 223–311, 2018.

[14] S. U. Stich, "Local sgd converges fast and communicates little," in *International Conference on Learning Representations (ICLR)*, 2019.

[15] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," in *International Conference on Learning Representations (ICLR)*, 2020.

[16] G. Ding, J. Wang, Q. Wu, L. Zhang, Y. Zou, Y.-D. Yao, and Y. Chen, "Robust spectrum sensing with crowd sensors," *IEEE Transactions on Communications*, vol. 62, no. 9, pp. 3129–3143, 2014.

[17] Waze: Outsmarting traffic, together. [Online]. Available: https://www.waze.com/

[18] M. Mun, S. Reddy, K. Shilton, N. Yau, J. Burke, D. Estrin, M. Hansen, E. Howard, R. West, and P. Boda, "PEIR, the personal environmental impact report, as a platform for participatory sensing systems research," in *ACM International Conference on Mobile systems, applications, and services (MobiSys)*, 2009.

[19] R. K. Rana, C. T. Chou, S. S. Kanhere, N. Bulusu, and W. Hu, "Ear-phone: An end-to-end participatory urban noise mapping system," in *ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2010.

[20] SmartRoadSense: A crowd sensing application for the continued monitoring of road quality. [Online]. Available: http://informatica.uniurb.it/smartroadsense-crowdsensing-civico/

[21] X. Wang, X. Wang, and S. Mao, "CiFi: Deep convolutional neural networks for indoor localization with 5GHz Wi-Fi," in *IEEE International Conference on Communications (ICC)*, 2017.

[22] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing," in *ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2012.

[23] I. Koutsopoulos, "Optimal incentive-driven design of participatory sensing systems," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2013.

[24] X. Gong and N. Shroff, "Truthful mobile crowdsensing for strategic users with private qualities," in *International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2017.

[25] Y. Luo, N. B. Shah, J. Huang, and J. Walrand, "Parametric prediction from parametric agents," in *The 10th Workshop on the Economics of Networks, Systems and Computation (NetEcon)*, 2015.

[26] Y. Cai, C. Daskalakis, and C. H. Papadimitriou, "Optimum statistical estimation with strategic data sources," in *Conference on Learning Theory (COLT)*, 2015.

[27] Y. Liu and Y. Chen, "A bandit framework for strategic regression," in *Advances in Neural Information Processing Systems*, 2016, pp. 1821–1829.

[28] ——, "Machine-learning aided peer prediction," in *ACM Conference on Economics and Computation (EC)*, 2017.

[29] D. R. Karger, S. Oh, and D. Shah, "Budget-optimal crowdsourcing using low-rank matrix approximations," in *IEEE Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2011.

[30] D. Lee, J. Kim, H. Lee, and K. Jung, "Reliable multiple-choice iterative algorithm for crowdsourcing systems," in *ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, 2015.

[31] G. Bianchi, C. Carusi, and L. Bracciale, "An online approach for joint task assignment and worker evaluation in crowd-sourcing," in *2017 International Symposium on Networks, Computers and Communications (ISNCC)*. IEEE, 2017, pp. 1–8.

[32] Y. Bachrach, T. Graepel, T. Minka, and J. Guiver, "How to grade a test without knowing the answers—a bayesian graphical model for adaptive crowdsourcing and aptitude testing," *arXiv preprint arXiv:1206.6386*, 2012.

[33] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics New York, 2001.

[34] S. M. Kay, "Fundamentals of statistical signal processing, volume i: Estimation theory," 1993.

[35] M. Xiao, J. Wu, S. Zhang, and J. Yu, "Secret-sharing-based secure user recruitment protocol for mobile crowdsensing," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2017.

[36] C. Miao, L. Su, W. Jiang, Y. Li, and M. Tian, "A lightweight privacy-preserving truth discovery framework for mobile crowd sensing systems," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2017.

[37] X. Tang, C. Wang, X. Yuan, and Q. Wang, "Non-interactive privacy-preserving truth discovery in crowd sensing applications," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2018.

[38] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[39] H. Jin, L. Su, B. Ding, K. Nahrstedt, and N. Borisov, "Enabling privacy-preserving incentives for mobile crowd sensing systems," in *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2016.

[40] X. Jin and Y. Zhang, "Privacy-preserving crowdsourced spectrum sensing," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2016.

[41] J. Lin, D. Yang, M. Li, J. Xu, and G. Xue, "Frameworks for privacy-preserving mobile crowdsensing incentive mechanisms," *IEEE Transactions on Mobile Computing*, vol. 17, no. 8, pp. 1851–1864, 2017.

[42] Z. Wang, J. Li, J. Hu, J. Ren, Z. Li, and Y. Li, "Towards privacy-preserving incentive for mobile crowdsensing under an untrusted platform," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2019.

[43] H. Jin, L. Su, H. Xiao, and K. Nahrstedt, "INCEPTION: Incentivizing privacy-preserving data aggregation for mobile crowd sensing systems," in *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2016.

[44] L. Yang, M. Zhang, S. He, M. Li, and J. Zhang, "Crowd-empowered privacy-preserving data aggregation for mobile crowdsensing," in *International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*. ACM, 2018, pp. 151–160.

[45] W. Wang, L. Ying, and J. Zhang, "The value of privacy: Strategic data subjects, incentive mechanisms and fundamental limits," in *ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, 2016.

[46] J. Wang, J. Tang, D. Yang, E. Wang, and G. Xue, "Quality-aware and fine-grained incentive mechanisms for mobile crowdsensing," in *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2016.

[47] Z. Wang, J. Hu, R. Lv, J. Wei, Q. Wang, D. Yang, and H. Qi, "Personalized privacy-preserving task allocation for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 6, pp. 1330–1341, 2018.

[48] W. Jin, M. Xiao, M. Li, and L. Guo, "If you do not care about it, sell it: Trading location privacy in mobile crowd sensing," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2019.

[49] Y. Liu and M. Liu, "An online learning approach to improving the quality of crowdsourcing," in *ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, 2015.

[50] H. Jin, L. Su, D. Chen, K. Nahrstedt, and J. Xu, "Quality of information aware incentive mechanisms for mobile crowd sensing systems," in *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2015.

[51] H. Jin, L. Su, and K. Nahrstedt, "CENTURION: Incentivizing multi-requester mobile crowd sensing," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2017.

[52] X. Zhang and X. Gong, "Online data quality learning for quality-aware crowdsensing," in *IEEE International Conference on Sensing, Communication and Networking (SECON)*, 2019.

[53] Y. Zhao and X. Gong, "Truthful quality-aware data crowdsensing for machine learning," in *IEEE International Conference on Sensing, Communication and Networking (SECON)*, 2019.

[54] L. Pu, X. Chen, J. Xu, and X. Fu, "Crowdlet: Optimal worker recruitment for self-organized mobile crowdsourcing," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2016.

[55] H. Zhang, B. Liu, H. Susanto, G. Xue, and T. Sun, "Incentive mechanism for proximity-based mobile crowd service systems," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2016.

[56] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic game theory*. Cambridge University Press, 2007, vol. 1.

[57] Y. Liu and Y. Chen, "Learning to incentivize: Eliciting effort via output agreement," *International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.

[58] H. Jin, L. Su, and K. Nahrstedt, "Theseus: Incentivizing truth discovery in mobile crowd sensing systems," in *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2017.

[59] C.-J. Ho, S. Jabbari, and J. W. Vaughan, "Adaptive Task Assignment for Crowdsourced Classification," p. 9.

[60] A. Gupta, K. Ligett, F. McSherry, A. Roth, and K. Talwar, "Differentially private combinatorial optimization," in *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 2010, pp. 1106–1125.

[61] F. McSherry and K. Talwar, "Mechanism design via differential privacy." in *FOCS*, vol. 7, 2007, pp. 94–103.

[62] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," *Journal of Privacy and Confidentiality*, vol. 7, no. 3, pp. 17–51, 2016.

[63] L. Bracciale, M. Bonola, P. Loreti, G. Bianchi, R. Amici, and A. Rabuffi, "CRAWDAD dataset roma/taxi (v. 2014-07-17)," Downloaded from https://crawdad.org/roma/taxi/20140717, Jul. 2014.

[64] X. Gong and N. Shroff, "Incentivizing truthful data quality for quality-aware mobile data crowdsourcing," in *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2018.

[65] C. Miao, Q. Li, H. Xiao, W. Jiang, M. Huai, and L. Su, "Towards data poisoning attacks in crowd sensing systems," in *Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*. ACM, 2018, pp. 111–120.

[66] E. D. Klenske and P. Hennig, "Dual control for approximate Bayesian reinforcement learning," *Journal of Machine Learning Research*, vol. 17, no. 8, pp. 1–30, 2016.

[67] J. Gittins, K. Glazebrook, and R. Weber, *Multi-armed bandit allocation indices*. John Wiley Sons, 2011.

[68] F. Liu and N. Shroff, "Data poisoning attacks on stochastic bandits," *arXiv preprint arXiv:1905.06494*, 2019.

[69] Y. Ma, K.-S. Jun, L. Li, and X. Zhu, "Data poisoning attacks in contextual bandits," in *International Conference on Decision and Game Theory for Security*. Springer, 2018, pp. 186–204.

[70] K.-S. Jun, L. Li, Y. Ma, and J. Zhu, "Adversarial attacks on stochastic bandits," in *Advances in Neural Information Processing Systems*, 2018, pp. 3640–3649.

[71] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," *arXiv preprint arXiv:1206.6389*, 2012.

[72] Y. Wang and K. Chaudhuri, "Data poisoning attacks against online learning," *arXiv preprint arXiv:1808.08994*, 2018.

[73] B. Li, Y. Wang, A. Singh, and Y. Vorobeychik, "Data poisoning attacks on factorization-based collaborative filtering," in *Advances in neural information processing systems*, 2016, pp. 1885–1893.

[74] S. Alfeld, X. Zhu, and P. Barford, "Data poisoning attacks against autoregressive models," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[75] C. Miao, Q. Li, L. Su, M. Huai, W. Jiang, and J. Gao, "Attack under disguise: An intelligent data poisoning attack mechanism in crowdsourcing," in *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 2018, pp. 13–22.

[76] L. Duan, T. Kubo, K. Sugiyama, J. Huang, T. Hasegawa, and J. Walrand, "Incentive mechanisms for smartphone collaboration in data acquisition and distributed computing," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2012.

[77] Z. Qin, Q. Li, and G. Hsieh, "Defending against cooperative attacks in cooperative spectrum sensing," *IEEE Transactions on Wireless communications*, vol. 12, no. 6, pp. 2680–2687, 2013.

[78] M. Fang, M. Sun, Q. Li, N. Z. Gong, J. Tian, and J. Liu, "Data poisoning attacks and defenses to crowdsourcing systems," in *WWW'21: Proceedings of The Web Conference 2021*. The Web Conference, 2020.

[79] S. Zhi, F. Yang, Z. Zhu, Q. Li, Z. Wang, and J. Han, "Dynamic truth discovery on numerical data," in *2018 IEEE International Conference on Data Mining (ICDM)*, 2018, pp. 817–826.

[80] B. McMahan and D. Ramage, "Federated learning: Collaborative machine learning without centralized training data," *Google Research Blog*, vol. 3, 2017.

[81] O. Shamir and N. Srebro, "Distributed stochastic optimization and learning," in *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2014, pp. 850–857.

[82] Y. Zhang and X. Lin, "Disco: Distributed optimization for self-concordant empirical loss," in *International Conference on Machine Learning*, 2015, pp. 362–370.

[83] S. Teerapittayanon, B. McDanel, and H.-T. Kung, "Distributed deep neural networks over the cloud, the edge and end devices," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 328–339.

[84] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "When edge meets learning: Adaptive control for resource-constrained distributed machine

learning," in *International Conference on Computer Communications (INFOCOM)*. IEEE, 2018, pp. 63–71.

[85] Y. Zhang, M. J. Wainwright, and J. C. Duchi, "Communication-efficient algorithms for statistical optimization," in *Advances in Neural Information Processing Systems*, 2012, pp. 1502–1510.

[86] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," in *Advances in Neural Information Processing Systems*, 2017, pp. 1709–1720.

[87] T. Sun, R. Hannah, and W. Yin, "Asynchronous coordinate descent under more realistic assumptions," in *Advances in Neural Information Processing Systems*, 2017, pp. 6182–6190.

[88] A. T. Suresh, F. X. Yu, S. Kumar, and H. B. McMahan, "Distributed mean estimation with limited communication," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3329–3337.

[89] W. Wen, C. Xu, F. Yan, C. Wu, Y. Wang, Y. Chen, and H. Li, "Terngrad: Ternary gradients to reduce communication in distributed deep learning," in *Advances in neural information processing systems*, 2017, pp. 1509–1519.

[90] T. Chen, G. Giannakis, T. Sun, and W. Yin, "Lag: Lazily aggregated gradient for communication-efficient distributed learning," in *Advances in Neural Information Processing Systems*, 2018, pp. 5050–5060.

[91] Y. H. Oh, Q. Quan, D. Kim, S. Kim, J. Heo, S. Jung, J. Jang, and J. W. Lee, "A portable, automatic data qantizer for deep neural networks," in *Proceedings of the 27th International Conference on Parallel Architectures and Compilation Techniques*. ACM, 2018, p. 17.

[92] M. I. Jordan, J. D. Lee, and Y. Yang, "Communication-efficient distributed statistical inference," *Journal of the American Statistical Association*, pp. 1–14, 2018.

[93] L. Li, D. Shi, R. Hou, H. Li, M. Pan, and Z. Han, "To talk or to work: Flexible communication compression for energy efficient federated learning over heterogeneous mobile edge devices," in *International Conference on Computer Communications (INFOCOM)*. IEEE, 2021.

[94] B. Luo, X. Li, S. Wang, J. Huang, and L. Tassiulas, "Cost-effective federated learning design," in *International Conference on Computer Communications (INFOCOM)*. IEEE, 2021.

[95] N. H. Tran, W. Bao, A. Zomaya, N. M. NH, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *International Conference on Computer Communications*. IEEE, 2019, pp. 1387–1395.

[96] H. H. Yang, Z. Liu, T. Q. Quek, and H. V. Poor, "Scheduling policies for federated learning in wireless networks," *IEEE Transactions on Communications*, vol. 68, no. 1, pp. 317–333, 2019.

[97] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, 2020.

[98] J. Xu and H. Wang, "Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective," *Transactions on Wireless Communications*, 2020.

[99] W. Shi, S. Zhou, Z. Niu, M. Jiang, and L. Geng, "Joint device scheduling and resource allocation for latency constrained wireless federated learning," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 453–467, 2020.

[100] J. Ren, Y. He, D. Wen, G. Yu, K. Huang, and D. Guo, "Scheduling for cellular federated edge learning with importance and channel awareness," *IEEE Transactions on Wireless Communications*, vol. 19, no. 11, pp. 7690–7703, 2020.

[101] S. Wang, M. Lee, S. Hosseinalipour, R. Morabito, M. Chiang, and C. G. Brinton, "Device sampling for heterogeneous federated learning: Theory, algorithms, and implementation," in *International Conference on Computer Communications (INFOCOM)*. IEEE, 2021.

[102] J. Zhang, N. Li, and M. Dedeoglu, "Federated learning over wireless networks: A band-limited coordinated descent approach," in *International Conference on Computer Communications (INFOCOM)*. IEEE, 2021.

[103] Y. Tu, Y. Ruan, S. Wang, S. Wagle, C. G. Brinton, and C. Joe-Wang, "Network-aware optimization of distributed learning for fog computing," in *International Conference on Computer Communications*, 2020.

[104] M. P. Friedlander and M. Schmidt, "Hybrid deterministic-stochastic methods for data fitting," *SIAM Journal on Scientific Computing*, vol. 34, no. 3, pp. A1380–A1405, 2012.

[105] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, "Optimal distributed online prediction using mini-batches," *The Journal of Machine Learning Research*, vol. 13, pp. 165–202, 2012.

[106] O. Shamir, "Without-replacement sampling for stochastic gradient methods," *Advances in neural information processing systems*, vol. 29, pp. 46–54, 2016.

[107] D. Nagaraj, P. Jain, and P. Netrapalli, "Sgd without replacement: Sharper rates for general smooth convex functions," in *International Conference on Machine Learning*. PMLR, 2019, pp. 4703–4711.

[108] Y. Zhao and X. Gong, "Quality-aware distributed computation and user selection for cost-effective federated learning," in *International Conference on Computer Communications (INFOCOM) Workshop on Distributed Machine Learning and Fog Networks (FOGML)*. IEEE, 2021.

[109] N. Buchbinder, M. Feldman, J. Naor, and R. Schwartz, "A tight linear time (1/2)-approximation for unconstrained submodular maximization," in *2013 IEEE*

54th Annual Symposium on Foundations of Computer Science. Los Alamitos, CA, USA: IEEE Computer Society, oct 2012, pp. 649–658. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/FOCS.2012.73

[110] Y. LeCun, C. Cortes, and C. Burges, "Mnist handwritten digit database," *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, vol. 2, 2010.

[111] Z. Yang, M. Chen, W. Saad, C. S. Hong, M. Shikh-Bahaei, H. V. Poor, and S. Cui, "Delay minimization for federated learning over wireless communication networks," *arXiv preprint arXiv:2007.03462*, 2020.

[112] Y. Zhao and X. Gong, "Quality-aware distributed computation and user selection for cost-effective federated learning," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2021.

[113] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," *arXiv preprint arXiv:1907.02189*, 2019.

[114] H. Yu, S. Yang, and S. Zhu, "Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 5693–5700.

[115] S. Ghadimi and G. Lan, "Stochastic first-and zeroth-order methods for nonconvex stochastic programming," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2341–2368, 2013.

[116] H. Yu and R. Jin, "On the computation and communication complexity of parallel sgd with dynamic batch sizes for stochastic non-convex optimization," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7174–7183.

[117] X. Lian, Y. Huang, Y. Li, and J. Liu, "Asynchronous parallel stochastic gradient for nonconvex optimization," in *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*, 2015, pp. 2737–2745.

[118] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," *arXiv preprint arXiv:1903.03934*, 2019.

[119] F. Niu, B. Recht, C. Ré, and S. J. Wright, "Hogwild!: A lock-free approach to parallelizing stochastic gradient descent," *arXiv preprint arXiv:1106.5730*, 2011.

[120] L. Cannelli, F. Facchinei, V. Kungurtsev, and G. Scutari, "Asynchronous parallel algorithms for nonconvex optimization," *Mathematical Programming*, pp. 1–34, 2019.

[121] X. Lian, W. Zhang, C. Zhang, and J. Liu, "Asynchronous decentralized parallel stochastic gradient descent," in *International Conference on Machine Learning*. PMLR, 2018, pp. 3043–3052.

[122] S. Mei, Y. Bai, A. Montanari *et al.*, "The landscape of empirical risk for nonconvex losses," *Annals of Statistics*, vol. 46, no. 6A, pp. 2747–2774, 2018.

[123] R. H. L. Sim, Y. Zhang, M. C. Chan, and B. K. H. Low, "Collaborative machine learning with incentive-aware model rewards," in *International Conference on Machine Learning*. PMLR, 2020, pp. 8927–8936.

[124] H. Yu, Z. Liu, Y. Liu, T. Chen, M. Cong, X. Weng, D. Niyato, and Q. Yang, "A sustainable incentive scheme for federated learning," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 58–69, 2020.

[125] P. Sun, H. Che, Z. Wang, Y. Wang, T. Wang, L. Wu, and H. Shao, "Pain-fl: Personalized privacy-preserving incentive for federated learning," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3805–3820, 2021.

[126] M. Zhang, E. Wei, and R. Berry, "Faithful edge federated learning: Scalability and privacy," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3790–3804, 2021.

[127] S. R. Pandey, N. H. Tran, M. Bennis, Y. K. Tun, A. Manzoor, and C. S. Hong, "A crowdsourcing framework for on-device federated learning," *IEEE Transactions on Wireless Communications*, vol. 19, no. 5, pp. 3241–3256, 2020.

[128] Y. Jiao, P. Wang, D. Niyato, B. Lin, and D. I. Kim, "Toward an automated auction framework for wireless federated learning services market," *IEEE Transactions on Mobile Computing*, vol. 20, no. 10, pp. 3034–3048, 2020.

[129] K. Donahue and J. Kleinberg, "Model-sharing games: Analyzing federated learning under voluntary participation," in *AAAI Conference on Artificial Intelligence*, 2021.

[130] ——, "Optimality and stability in federated learning: A game-theoretic approach," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[131] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 700–10 714, 2019.

[132] N. Ding, Z. Fang, and J. Huang, "Optimal contract design for efficient federated learning with multi-dimensional private information," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 186–200, 2020.

[133] M. Zhang, E. Wei, and R. Berry, "Faithful edge federated learning: Scalability and privacy," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3790–3804, 2021.

[134] Z. Feng, Y. Zhu, Q. Zhang, L. M. Ni, and A. V. Vasilakos, "Trac: Truthful auction for location-aware collaborative sensing in mobile crowdsourcing," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2014.

[135] A. Tarable, A. Nordio, E. Leonardi, and M. A. Marsan, "The importance of being earnest in crowdsourcing systems," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2015.

[136] N. B. Shah and D. Zhou, "Double or nothing: Multiplicative incentive mechanisms for crowdsourcing," in *Conference on Neural Information Processing Systems (NIPS)*, 2015.

[137] P. Bolton and M. Dewatripont, *Contract theory*. MIT press, 2005.

[138] A. Dasgupta and A. Ghosh, "Crowdsourced judgement elicitation with endogenous proficiency," in *International World Wide Web Conference (WWW)*, 2013.

[139] Y. Liu and Y. Chen, "Sequential peer prediction: Learning to elicit effort using posted prices." in *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[140] D. Prelec, "A Bayesian truth serum for subjective data," *Science*, vol. 306, no. 5695, pp. 462–466, 2004.

[141] THE MNIST DATABASE (http://yann.lecun.com/exdb/mnist/). [Online]. Available: http://yann.lecun.com/exdb/mnist/