Byzantine-Resilient Federated Learning at Edge

Youming Tao, *Student Member, IEEE*, Sijia Cui, Wenlu Xu, Haofei Yin, Dongxiao Yu, *Senior Member, IEEE*, Weifa Liang, *Senior Member, IEEE*, and Xiuzhen Cheng, *Fellow, IEEE*

***Abstract*—Both Byzantine resilience and communication efficiency have attracted tremendous attention recently for their significance in edge federated learning. However, most existing algorithms may fail when dealing with real-world irregular data that behaves in a heavy-tailed manner. To address this issue, we study the stochastic convex and non-convex optimization problem for federated learning at edge and show how to handle heavy-tailed data while retaining the Byzantine resilience, communication efficiency and the optimal statistical error rates simultaneously. Specifically, we first present a Byzantine-resilient distributed gradient descent algorithm that can handle the heavy-tailed data and meanwhile converge under the standard assumptions. To reduce the communication overhead, we further propose another algorithm that incorporates gradient compression techniques to save communication costs during the learning process. Theoretical analysis shows that our algorithms achieve order-optimal statistical error rate in presence of Byzantine devices. Finally, we conduct extensive experiments on both synthetic and real-world datasets to verify the efficacy of our algorithms.**

***Index Terms*—Byzantine resilience, communication efficiency, edge intelligent systems, federated learning.**

I. Introduction

RECENT years have witnessed the proliferation of smart edge devices, which leads to an unprecedented amount of data generated at the network edge. Thanks to the significant increasing in computation power of edge devices and the ubiquitous deployment of communication infrastructures, data can be processed locally and aggregated across devices efficiently. With these merits, it is natural to implement large-scale machine learning algorithms at edge, which brings about the concept of edge intelligence and has empowered many emerging applications that benefit human lives, such as smart city and autonomous driving.

Due to the widespread concerns over data ownership and privacy, federated learning (FL), proposed by Google [1], has emerged as a popular paradigm for distributed ML model training, see, e.g., [2], [3], [4]. In edge FL, the data is retained in edge devices and processed in parallel, thus much more real-time results can be provided for real-world applications. However, there remain some practical issues that hinder the successful implementation of edge FL.

One commonly encountered issue in such large-scale distributed systems arises from the potential unreliability of edge devices. Distributing the computation over multiple devices induces a higher risk of failures. In particular, some devices in the system may not follow the predefined protocol and exhibit abnormal behaviors, either actively or passively due to crashes, malfunctioning hardware, unreliable communication channels or attacks from adversaries. The inherently unpredictable behaviors of faulty devices are usually modeled as *Byzantine fault* [5], [6], [7]. It has been shown in [8] that even when the number of Byzantine devices is small (even only one) and the value sent by them is moderate and even difficult to detect, the performance can still be significantly degraded. Thus, Byzantine resilience has always been a main consideration in the design of FL frameworks, see, e.g., [9], [10], [11].

Communication overhead is also an important consideration in edge FL. Typically, edge devices need to upload local results (gradients or model parameters) to the server for global aggregation repeatedly. Due to inherently limited bandwidth of wireless channels, exchanging data between edge devices and the server will incur heavy communication load and might cause network congestion, which is especially the case when the device number is huge. Heavy communication overhead is the major bottleneck that hinders the parallelism and scalability of FL at edge [12].

In addition, data is a key ingredient in machine learning. Recent studies, e.g., [13], [14], [15], have shown that heavy-tailed noises exist widely in practical multi-sensor systems. Since most data stored in edge devices are collected via various sensors, it is natural for the data used for training learning models to be irregular and behave in a heavy-tailed manner. Furthermore, in many real-world applications, data have been observed to be heavy-tailed in themselves, especially those from biomedicine [16], [17] and finance [18], [19]. In a nutshell, heavy-tailed data can be widespread at edge. Heavy-tailed data could degrade the performance of learning algorithms (see, e.g., [20], [21]), and the presence of Byzantine devices could make things worse for federated training at edge. Unfortunately, existing works on Byzantine resilience in FL all make strong assumptions on the distribution of loss gradients, for example, sub-exponential gradients [22], [23], gradients with bounded skewness [8], or gradients with norm-wise bounded

Manuscript received 23 January 2022; revised 25 August 2022; accepted 26 February 2023. Date of publication 15 March 2023; date of current version 9 August 2023. This work was supported in part by the National Key Research and Development Program of China Grant 2020YFB1005900, and in part by the National Natural Science Foundation of China (NSFC) under Grant 62122042. Recommended for acceptance by J. Cao. *(Corresponding author: Dongxiao Yu.)*

Youming Tao, Haofei Yin, Dongxiao Yu, and Xiuzhen Cheng are with the School of Computer Science and Technology, Shandong University, Qingdao, Shandong 250100, China (e-mail: ym.tao99@mail.sdu.edu.cn; hf_yin@mail.sdu.edu.cn; dxyu@sdu.edu.cn; xzcheng@sdu.edu.cn).

Sijia Cui is with the Institute of Automation, University of Chinese Academy of Sciences, Beijing 101408, China (e-mail: cuisijia2022@ia.ac.cn).

Wenlu Xu is with the Department of Statistics, University of California, Los Angeles, CA 90095 USA (e-mail: wenluxu@ucla.edu).

Weifa Liang is with the Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong (e-mail: weifa.liang@cityu.edu.hk).

Digital Object Identifier 10.1109/TC.2023.3257510

variance [24] (see Section I-C for more details). Hence, how to mitigate the impact of heavy-tailed data on edge FL is an urgent requirement and still lack investigation.

With the perceptions above, in this paper, we consider Byzantine resilience, communication efficiency and heavy-tailed data robustness simultaneously for the first time. In particular, we have the following natural question:

Is there any way to handle the heavy-tailed data for edge federated learning while retaining the Byzantine resilience, communication efficiency, and the optimal statistical error rates?

In this paper, we provide affirmative answer to the above question. We design an edge FL framework that is robust to heavy-tailed data as well as satisfies the requirement of Byzantine resilience and communication efficiency. Our main contributions and technical challenges are as follows.

A. Main Contributions

In the first part, we conduct a comprehensive study on the Byzantine-tolerant distributed gradient descent with heavy-tailed data under the standard assumptions. In particular, for heavy-tailed data, we assume that the distribution of loss gradients has only *coordinate-wise* bounded second-order raw moment. We establish the high-probability guarantees of statistical error rate for strongly convex, general convex and non-convex population risk functions respectively. Specifically, for all the cases, we show that our algorithm achieves the following statistical error rate[1]

$$\widetilde{\mathcal{O}}\left(d^2\left[\frac{\alpha^2}{n}+\frac{1}{mn}\right]\right),$$

where $\alpha \in (0, \frac{1}{2})$ is the fraction of Byzantine devices, n is the size of local dataset on each edge device and m is the number of edge devices. The error rate above matches the error rate given in [8] and it has been shown in [8] that, for strongly-convex population risk functions and a fixed d, no Byzantine-resilient algorithm can achieve an error lower than $\widetilde{\Omega}(\frac{\alpha^2}{n}+\frac{1}{mn})$, which implies that our algorithm still achieve order-wise optimality in terms of (α, n, m), even in the presence of heavy-tailed data.

In the second part, we study how to further retain the optimal statistical error rates under the requirement of both Byzantine resilience and communication efficiency. To achieve the communication efficiency, we adopt the technique of gradient compression and consider a generic class of compressors called δ-approximate compressor. Based on this, we propose a communication-efficient and Byzantine resilient distributed gradient descent algorithm with heavy-tailed data. In this case, our statistical error rates becomes

$$\widetilde{\mathcal{O}}\left(d^2\left[\frac{\alpha^2}{n}+\frac{1-\delta}{n}+\frac{1}{mn}\right]\right),$$

where δ is the compression factor, and when $\delta = 1$ (which implies no compression), the error rate becomes $\widetilde{\mathcal{O}}(\frac{\alpha^2}{n}+\frac{1}{mn})$, which means that the compression term has no order-wise contribution to the error rate.

1.Throughout this paper, the notations $\widetilde{\mathcal{O}}(\cdot)$ and $\widetilde{\Omega}(\cdot)$ hide logarithmic factors.

B. Technical Challenges

When only considering the Byzantine resilience, a natural and direct idea to address the problem under the heavy-tailed data setting is to replace the robust aggregator used by the server with some state-of-art robust mean estimators that can deal with the data with *coordinate-wise* bounded second moment, like [25], [26], [27]. Unfortunately, as far as we know, these robust mean estimators are not appropriate for the edge FL settings due to the following reasons. First, they typically limit the corrupted data to a small fraction, which is usually not the case at edge. Second, to analyse the statistical error rate of the learning algorithm, it requires the estimators to have certain continuity property so that the *uniform* estimation error can be bounded. The reason why the uniform error bound is needed is due to the fact that the analysis of these robust mean estimators relies on the assumption of i.i.d. data. In FL, this means the gradients computed using the training dataset should be i.i.d., which is true for a given model parameter. However, the model parameter is updated iteratively based on the training dataset and thus the parameters across the iterations are highly dependent on each other. As a result, in any iteration $t > 1$, the gradients computed using the training dataset are no longer i.i.d.. Hence, we have to establish uniform concentration to bound the estimation error for all possible parameters simultaneously. It is still unclear whether these existing estimators can achieve the uniform convergence. In this paper, to solve the problem, we let the server and the devices jointly estimate the expected loss gradient in each iteration. At the device end, each device first performs a robust local mean estimator based on soft truncation and noise smoothness, which is motivated by [20]. Then the central machine aggregates the gradient estimates by coordinate-wise trimmed mean to rule out the outliers caused by Byzantine nodes. Based on this, we propose an efficient and more robust distributed gradient descent algorithm. The major challenge in analysis is to bound the uniform error when the local gradient estimator is combined with the coordinate-wise trimmed mean. We overcome this by first analysing the point-wise error bound for each coordinate and then using the coordinate-wise continuity of the local gradient estimator to obtain the uniform error bound for each coordinate via the covering arguments.

When further considering the gradient compression, we let the server perform a norm-based trimmed mean to aggregate the compressed local estimators. The key challenge here becomes to analyse the uniform error bound for the combination of the compressed local gradient estimator and the norm-based trimmed mean. After introducing the gradient compression, the trimming process becomes norm-based and our previous "coordinate-wise" analysis no longer applies. Thus we have to adopt a different analysis. To tackle this problem, we build up on the techniques of [23] to directly bound the uniform error of the local estimator and then consider the impact of the compression.

C. Related Work

To cope with Byzantine attacks in distributed learning setting, most solutions rely on the outlier-robust estimators for aggregation, such as coordinate-wise median [8], coordinate-wise

mean [8], geometric median [22], [28] and majority voting [29], [30], instead of vanilla averaging. Also, there are works developing robust aggregators by combining the ideas of these robust estimators. For example, [31] proposed Krum based on the ideas of majority voting and geometric median, and [32] proposed Bulyan that ensures majority agreement on each coordinate of the aggregated gradients by combining Krum and coordinated-wise trimmed mean. A critical issue in these approaches is that their convergence guarantees rely on strong assumptions on gradient distributions. In particular, [24] provided some examples to show that these approaches do not obtain the true optimum especially when dealing with heavy-tailed distributions. To fix it, [24] proposed to utilize worker momentum and just assumed norm-wise bounded variance for loss gradients. Inspired by this, recent work [9] further investigated distributed momentum for Byzantine learning under the norm-wise bounded variance assumption. However, this assumption still implies that data are well-behaved to some extent. According to recent empirical findings, e.g., [33], the gradient noise could be α-stable random variable with extremely heavy tails. In this paper, we take the first step for dealing with more extremely heavy-tailed data (or gradients) under the assumption that the loss gradients have only *coordinate-wise* bounded second raw moment.

To the best of our knowledge, there are quite limited works focusing on Byzantine resilient learning and gradient compression simultaneously, except for a few notable exceptions of [11], [23], [34]. [34] assumed that all devices have access to the same data and their method can only tolerate blind multiplicative adversaries (i.e., adversaries that must determine how to corrupt the gradient before observing the true gradient and can only multiply each coordinate of the true gradient by arbitrary scalar). In contrast, we consider stronger adversaries and a more general setting where different devices have different local datasets. [23] proposed algorithms that combined the robust aggregators and gradient compression together, and introduced error feedback to further reduce the communication costs. However, their convergence guarantees rely on the sub-exponential gradients assumption, which makes their methods not applicable to our setting. Lately, [11] considered the Byzantine resilience and communication efficiency issues together in the heterogeneous data setting. Unfortunately, their results also rely on the norm-wise bounded variance assumption for loss gradients, while we relax this assumption in this work.

D. Road Map

The remaining part of the paper is organized as follows. The formal problem definition and system model are given in Section II. In Section III, we propose a distributed gradient descent algorithm that is robust to both Byzantine fault and heavy-tailed training data. We show that our proposed algorithm can achieve the optimal statistical error rates. In Section IV, we consider how to further reduce the communication overheads, and propose a modified algorithm by introducing the gradient compression schemes. The optimal statistical error rates are shown to be retained. In Section V, we report the experimental results. Finally, we conclude the paper in Section VI.

II. Problem Setup and Preliminaries

A. Edge Federated Learning Problem

We consider the stochastic convex and non-convex optimization problem. Formally, let $\mathcal{W} \subseteq \mathbb{R}^d$ be the parameter space containing all the possible model parameters and $\mathcal{D}$ be an unknown distribution over the data universe $\mathcal{Z}$. Given a loss function $\ell : \mathcal{W} \times \mathcal{Z} \to \mathbb{R}$, where $\ell(w, z)$ measures the risk induced by data z under the model parameter choice w, and a dataset $D = \{z_1, z_2, \ldots, z_N\}$, where z_i's are i.i.d. samples from the distribution $\mathcal{D}$ over $\mathcal{Z}$, the goal is to learn an optimal parameter choice $w^* \in \mathcal{W}$ that minimizes the population risk $R_{\mathcal{D}}(w)$, i.e.,

$$w^* \in \arg\min_{w \in \mathcal{W}} R_{\mathcal{D}}(w) \triangleq \mathbb{E}_{z \sim \mathcal{D}}[\ell(w, z)]^2. \tag{1}$$

[2] Note that since the data distributed $\mathcal{D}$ is unknown, the population risk function $R_{\mathcal{D}}(\cdot)$ is typically unknown in practice. Hence, we cannot compute w^* straightforwardly by solving the minimization problem in (1) with gradient descent (GD).

We focus on solving the above stochastic optimization problem over an edge intelligent system via the federated learning framework. The edge intelligent system consists of an edge server and m edge devices. We assume that the total N training data are evenly distributed across the m devices such that each worker machine holds $n = \frac{N}{m}$ data [3].

We consider a synchronous distributed system where the server can communicate with devices in each round. Among the m devices, at most α ($\alpha < \frac{1}{2}$) fraction of devices are Byzantine and the rest $1 - \alpha$ fraction are normal/good. In each round, the good devices will follow the predefined protocols faithfully. While for Byzantine ones, we have the following assumptions. First, we assume that the set of Byzantine devices can be *dynamic* throughout the learning process. We denote the Byzantine devices in round t as $\mathcal{B}_t$ and the remaining good devices as $\mathcal{G}_t$. Second, we assume the Byzantine devices to be *omniscient*, i.e., they have complete knowledge of the system and the learning algorithm, and have access to the computations made by the rest good devices. Third, the Byzantine devices need not obey any predefined protocols and can send arbitrary message to the server (maybe send nothing at all) in each round. Moreover, Byzantine devices can even collude with each other. The only limit on Byzantine devices is that these devices cannot contaminate the local dataset.

B. Preliminaries

We first review some key concepts in optimization.

Definition 1 (Lipschitzness). A function $f : \mathcal{W} \to \mathbb{R}$ is L-Lipschitz if for $\forall w_1, w_2 \in \mathcal{W}$

$$|f(w_1) - f(w_2)| \leq L\|w_1 - w_2\|_2.$$

2.We assume that $R_{\mathcal{D}}(w) \triangleq \mathbb{E}_{z \sim \mathcal{D}}[\ell(w, z)]$ is well-defined for every $w \in \mathcal{W}$.

3.Although this is a simplified assumption on data balance over devices, our results can be easily extended to the heterogeneous data sizes setting provided the data sizes are of the same order. The same assumption has been adopted by many related works (e.g. [8], [22], [23]).

Definition 2 (Strong Convexity). A function f is a-strongly convex on $\mathcal{W}$ if for $\forall w_1, w_2 \in \mathcal{W}$

$$f(w_1) \geq f(w_2) + \langle \nabla f(w_2), w_1 - w_2 \rangle + \frac{a}{2}\|w_1 - w_2\|_2^2.$$

Definition 3 (Smoothness). A function f is b-smooth on $\mathcal{W}$ if for $\forall w_1, w_2 \in \mathcal{W}$

$$f(w_1) \leq f(w_2) + \langle \nabla f(w_2), w_1 - w_2 \rangle + \frac{b}{2}\|w_1 - w_2\|_2^2.$$

Definition 4 (Projection). Given a convex set $\mathcal{W} \subseteq \mathbb{R}^d$, the projection of any $\theta \in \mathbb{R}^d$ to $\mathcal{W}$ is denoted by

$$\prod_{\mathcal{W}} \theta = \arg\min_{w \in \mathcal{W}} \|\theta - w\|.$$

We make use of the following assumptions in the paper.

Assumption 1. The parameter space $\mathcal{W}$ is closed, convex, and bounded with diameter Δ, i.e., for $\forall w_1, w_2 \in \mathcal{W}$, $\|w_1 - w_2\|_2 \leq \Delta$.

Assumption 2. The population risk function $R_{\mathcal{D}}(w)$ is L_R-smooth for $\forall w \in \mathcal{W}$, where L_R is a known constant.

Assumption 3. For any given data $z \in \mathcal{Z}$, the loss gradient $\nabla \ell(w, z)$ satisfies that for each coordinate $k \in [d]$, $\nabla_k \ell(w, z)$ is L_k-Lipschitz. Let $\hat{L} \triangleq \sqrt{\sum_{k=1}^d L_k^2}$.

The above three assumptions are quite standard and has been commonly adopted in the previous works, e.g., [8], [23]. Furthermore, we make the following assumption for heavy-tailed data. Specifically, for any parameter w, we assume that loss gradients have only coordinate-wise bounded second raw moments.

Assumption 4. For any given $w \in \mathcal{W}$ and each coordinate $k \in [d]$, $\mathbb{E}_{z \sim \mathcal{D}}[\nabla_k^2 \ell(w, x)] \leq v$, where v is a known constant.

We note that this assumption is reasonable and has been used in some other learning problems with heavy-tailed data as well, e.g., [20], [35], [36], [37]. We provide a concrete example of classical linear regression to validate this.

Example. Consider a linear regression model $y = \langle w^*, x \rangle + \xi$, where $x \in \mathbb{R}^d$ is the feature vector and $y \in \mathbb{R}$ is the label, and ξ is the noise. For the noise, we assume that ξ is independent of x, and satisfies: 1) $\mathbb{E}[\xi] = 0$; 2) $\mathbb{E}[\xi^2] \leq c_1$ for some constant c_1. For the feature vector, we assume that the coordinates of $x = (x_1 \ldots, x_d)$ are independent of each other, and x satisfies: 1) $\mathbb{E}[x] = 0$; 2) $\mathbb{E}[\|x\|_2^2] \leq c_2$ for some constant c_2; 3) for $\forall k \in [d]$, $\mathbb{E}[x_k^4] \leq c_3$ for some constant c_3. We consider the quadratic loss function $\ell(w, (x, y)) = \frac{1}{2}(y - \langle w, x \rangle)^2$, then $\nabla_k \ell(w, (x, y)) = (y - \langle w, x \rangle) x_k$. By some simple computation, we have for all $w \in \mathcal{W}$ that

$$\begin{aligned}
&\mathbb{E}[\nabla_k^2 \ell(w, (x, y))] \\
&= \mathbb{E}[(y - \langle w, x \rangle)^2 x_k^2] = \mathbb{E}[(\langle w^* - w, x \rangle + \xi)^2 x_k^2] \\
&= \mathbb{E}[\langle w^* - w, x \rangle^2 x_k^2] + \mathbb{E}[\xi^2] \cdot \mathbb{E}[x_k^2] \\
&\leq \|w^* - w\|^2 \mathbb{E}[x_k^2 \cdot \|x\|_2^2] + \mathbb{E}[\xi^2] \cdot \mathbb{E}[x_k^2] \\
&\leq \Delta^2 \mathbb{E}[x_k^2 \cdot \|x\|_2^2] + \mathbb{E}[\xi^2] \cdot \mathbb{E}[x_k^2] < \Delta^2 (c_3 + c_2^2) + c_1 \cdot c_2
\end{aligned}$$

where the third equality is because ξ is independent of x and $\mathbb{E}[\xi] = 0$, and the second inequality is due to Assumption 1. It can be seen from above that the loss gradient only has coordinate-wise bounded second moment, which validates Assumption 4. Furthermore, denote the bound we got by C. Then the norm-wise variance of the loss gradient in our example is bounded as $\mathbb{E}[\|\nabla \ell(w, (x, y))\|_2^2] \leq d \cdot C$, which is proportional to the dimension d. In contrast, bounded norm-wise variance assumption used in previous works only assumed it to be a universal constant, which indicates that our Assumption 4 is weaker and thus more general.

III. Byzantine-Resilient Heavy-Tailed Gradient Descent

In this section, we study how to handle the heavy-tailed data while retaining the Byzantine resilience and the statistical error rate. We propose a Byzantine-resilient heavy-tailed gradient descent algorithm called BHGD.

A. Algorithm Design

Since $R_{\mathcal{D}}(\cdot)$ is unknown, it is infeasible to apply gradient descent algorithm directly due to the impossibility to compute the exact population risk gradient $\nabla R_{\mathcal{D}}(\cdot)$. A natural alternative way is to estimate $\nabla R_{\mathcal{D}}(\cdot)$ using the data samples $D = \{z_1, z_2, \ldots, z_N\}$. In our algorithm, the estimation is done jointly by devices and the central server. Each device first calculates a local estimate of $\nabla R_{\mathcal{D}}(\cdot)$ (for simplicity, we denote the local estimator of device $i \in [m]$ by $g_i(\cdot)$) from its local dataset and sends the local estimation to the server. The server aggregates the received $\{g_1, g_2, \ldots, g_m\}$ by coordinate-wise trimmed mean and then updates the parameter vector (see Algorithm 1 for details).

The local gradient estimator in our algorithm is inspired by the robust mean estimator for heavy-tailed distribution given in [20]. To be self-contained, we first review the estimator. For simplicity, we consider a one-dimensional random variable $x \sim \mathcal{X}$ and assume that $x_1, x_2, \ldots, x_n$ are i.i.d. samples of x. The robust estimator consists of three steps:

1) *Scaling and Truncation* For each sample x_i, we re-scale it by dividing s and apply a soft truncation function ϕ on the re-scaled one. Then we calculate the empirical mean of the altered samples and put the mean back to the original scale. That is

$$\frac{s}{n} \sum_{i=1}^n \phi\left(\frac{x_i}{s}\right) \approx \mathbb{E}[x].$$

2) *Noise Multiplication* Let $\epsilon_1, \ldots, \epsilon_2$ be independent random noise generated from a common distribution ν with $\mathbb{E}[\epsilon_i] = 0$ for each. We multiply each sample x_i by $(1 + \epsilon_i)$, and then perform the scaling and truncation step on $x_i \cdot (1 + \epsilon_i)$. That is

$$\widetilde{x}(\epsilon) = \frac{s}{n} \sum_{i=1}^n \phi\left(\frac{x_i + \epsilon_i x_i}{s}\right).$$

3) *Noise Smoothing* We smooth the multiplicative noise via taking the expectation with respect to the noise distribution

Algorithm 1: Byzantine-Resilient Heavy-Tailed Gradient Descent (BHGD).

Input: Initial parameter vector $w_0 \in \mathcal{W}$, step size η, time horizon T.

Initialize: $\zeta \leftarrow \frac{1}{(\Delta n\widehat{L})^d(m+1)d(mn)^d}$, $s \leftarrow \sqrt{\frac{nv}{2\log(1/\zeta)}}$, $\tau \leftarrow \sqrt{2\log(1/\zeta)}$.

1: **for** $t \leftarrow 0, 1, \ldots, T-1$ **do**

2: Server: Send w_t to all the worker machines

3: Each good device $i \in \mathcal{G}_t$ do in parallel:

1) Computes local estimate of gradient $g_i(w_t)$. Specifically, for $\forall k \in [d]$,

$$g_{i,k}(w_t) \leftarrow \frac{1}{n}\sum_{j=1}^{n}\left[g\left(1-\frac{g^2}{2s^2\tau}\right)-\frac{g^3}{6s^2}\right] + \frac{s}{n}\sum_{j=1}^{n} C\left(\frac{g}{s}, \frac{|g|}{s\sqrt{\tau}}\right),$$

where g denotes $\nabla_k \ell(w_t, x_j)$.

2) Sends $g_i(w_t)$ to the central machine

4: Server:

1) For each $k \in [d]$:

- Sorts $g_{i,k}(w_t)$'s in a non-decreasing order.
- Removes the largest and smallest β fraction of elements in $\{g_{i,k}(w_t)\}_{i=1}^m$ and denotes the indices of the remaining elements as $\mathcal{U}_{k,t}$
- Aggregates by

$$g_k(w_t) = \frac{1}{|\mathcal{U}_{k,t}|}\sum_{i\in\mathcal{U}_{k,t}} g_{i,k}(w_t)$$

and denotes $g(w_t) = (g_1(w_t), \ldots, g_d(w_t))$.

2) Updates the parameter by

$$w_{t+1} = \prod_{\mathcal{W}}(w_t - \eta \cdot g(w_t))$$

5: **end for**

ν. That is

$$\widehat{x} = \mathbb{E}[\widetilde{x}(\epsilon)] = \frac{s}{n}\sum_{i=1}^{n}\int \phi\left(\frac{x_i+\epsilon_i x_i}{s}\right) d\nu(\epsilon_i). \quad (2)$$

We note that the randomness in final estimator in (2) is only dependent on the original samples. The explicit form of the integral in (2) is related to the choice of the soft truncation function $\phi(\cdot)$ and the noise distribution ν. The results in [38] shows that, if we set ϕ to be

$$\phi(x) = \begin{cases} \frac{2\sqrt{2}}{3}, & x > \sqrt{2} \\ x - \frac{x^3}{6}, & -\sqrt{2} \le x \le \sqrt{2} \\ -\frac{2\sqrt{2}}{3}, & x < -\sqrt{2} \end{cases} \quad (3)$$

and set $\nu = \mathcal{N}(0, \frac{1}{\tau})$, then the integral in (2) has an explicit form such that it can be computed efficiently. Generally, for any a and b, we have

$$\mathbb{E}_\nu[\phi(a+b\sqrt{\tau}\epsilon)] = a\left(1-\frac{b^2}{2}\right) - \frac{a^3}{6} + C(a,|b|). \quad (4)$$

The term $C(a,|b|)$ in (4) is a correction term with a simple form. To give its explicit form, We first define some preparatory notations

$$V_- \triangleq \frac{\sqrt{2}-a}{|b|}, \quad V_+ \triangleq \frac{\sqrt{2}+a}{|b|},$$
$$F_- \triangleq \Phi(-V_-), \quad F_+ \triangleq \Phi(-V_+),$$
$$E_- \triangleq \exp\left(-\frac{V_-^2}{2}\right), \quad E_+ \triangleq \exp\left(-\frac{V_+^2}{2}\right),$$

where Φ denotes the CDF of the standard Gaussian distribution. Then with these atomic elements, the explicit form of $C(a,|b|)$ can be described as follows:

$$C(a,|b|) = T_1 + T_2 + T_3 + T_4 + T_5,$$

where

$$T_1 \triangleq \frac{2\sqrt{2}}{3}(F_- - F_+)$$
$$T_2 \triangleq -\left(a-\frac{a^3}{6}\right)(F_- + F_+)$$
$$T_3 \triangleq \frac{|b|}{\sqrt{2\pi}}\left(1-\frac{a^2}{2}\right)(E_+ - E_-)$$
$$T_4 \triangleq \frac{ab^2}{2}\left(F_+ + F_- + \frac{1}{\sqrt{2\pi}}(V_+E_+ + V_-E_-)\right)$$
$$T_5 \triangleq \frac{|b|^3}{6\sqrt{2\pi}}((2+V_-^2)E_- - (2+V_+^2)E_+).$$

The main idea of Algorithm 1 is that, instead of using empirical mean as the local estimator which may be subject to the heavy-tailed outliers, we let each device apply the one-dimensional robust mean estimator described above to each coordinate of its local loss gradients so that a more accurate local estimator $g_i(\cdot)$ for $\nabla R_{\mathcal{D}}(\cdot)$ can be obtained. Specifically, in our setting, the parameter a, b in (4) should be $\frac{\nabla_k \ell(w_t,x_j)}{s}$, $\frac{\nabla_k \ell(w_t,x_j)}{s\sqrt{\tau}}$ respectively, and the final estimator is described in step 3(1) in Algorithm 1. The server then uses the coordinate-wise trimmed mean to aggregate these local estimators and obtain a global estimator $g(\cdot)$ for $\nabla R_{\mathcal{D}}(\cdot)$ (step 4(1)). Note that, since the trimming threshold β is at least α, the trimming operation ensures that the effect of Byzantine devices can be removed and hence the global estimator $g(\cdot)$ is close to $\nabla R_{\mathcal{D}}(\cdot)$.

B. Theoretical Results

In this part, we analyse the performance of Algorithm 1. Specifically, we study the statistical error rates for strongly convex, general convex and non-convex population risk function respectively. For strongly-convex and general-convex case, we focus on the excess population risk, i.e., $R_{\mathcal{D}}(w_T) - R_{\mathcal{D}}(w^*)$. For non-convex case, we focus on the rate of convergence to a critical point of the population risk, i.e., $\min_{t=0,1,\ldots,T}\|\nabla R_{\mathcal{D}}(w_t)\|_2$.

For all the cases, we provide the high probability upper bounds on these error rates.

The analysis of the error rates relies on the gradient estimation error in each iteration $t \in [T]$, i.e., $\|g(w_t) - \nabla R_{\mathcal{D}}(w_t)\|$. Hence the key step is to bound the uniform error of $g(w)$ for all $w \in \mathcal{W}$. Specifically, we have the following high probability bound on $\|g(w) - \nabla R_{\mathcal{D}}(w)\|_2$ for all $w \in \mathcal{W}$.

Lemma 1. For all $w \in \mathcal{W}$, it holds with probability at least $1 - \frac{1}{(mn)^d}$ that

$$\|g(w) - \nabla R_{\mathcal{D}}(w)\|_2 \in \mathcal{O}\left(\alpha d\sqrt{\frac{\log(mn)}{n}} + d\sqrt{\frac{\log(mn)}{mn}}\right).$$

We now provide statistical error rates for our algorithm.

Strongly Convex Population Risks. We first consider the case where the population risk function $R_{\mathcal{D}}(\cdot)$ is strongly convex. Note that the loss function for each data $\ell(\cdot, z)$ need not be strongly convex.

Theorem 1. Suppose Assumption 1, 2, 3 and 4 hold, and $R_{\mathcal{D}}(\cdot)$ is λ_R-strongly convex. Choose step size $\eta = \frac{1}{L_R}$ and run Algorithm 1 for T rounds, then with probability at least $1 - \frac{1}{(mn)^d}$, we have the following bound on excess population risk,

$$R_{\mathcal{D}}(w_T) - R_{\mathcal{D}}(w^*) \leq L_R\left(1 - \frac{\lambda_R}{L_R + \lambda_R}\right)^{2T}\|w_0 - w^*\|_2^2 + \frac{4L_R}{\lambda_R^2}\mathcal{E}^2,$$

where $\mathcal{E} \in \mathcal{O}\left(\alpha d\sqrt{\frac{\log(mn)}{n}} + d\sqrt{\frac{\log(mn)}{mn}}\right)$.

General Convex Population Risks. For the general convex population risk case, we need a mild technical assumption on the size of the parameter space $\mathcal{W}$.

Assumption 5. The parameter space $\mathcal{W}$ contains the following ℓ_2 ball centered at w^*:

$$\{w \in \mathbb{R}^d : \|w - w^*\|_2 \leq 2\|w_0 - w^*\|_2\}.$$

Then we have the following result on excess population risk function.

Theorem 2. Suppose Assumption 1, 2, 3, 4 and 5 hold, and $R_{\mathcal{D}}(\cdot)$ is convex. Choose step size $\eta = \frac{1}{L_R}$ and run Algorithm 1 for $T = \frac{L_R}{\mathcal{E}}\|w_0 - w^*\|_2$ rounds, then with probability at least $1 - \frac{1}{(mn)^d}$, we have the following bound on excess population risk,

$$R_{\mathcal{D}}(w_T) - R_{\mathcal{D}}(w^*) \leq 16\Delta\mathcal{E} + \frac{1}{2L_R}\mathcal{E}^2,$$

where $\mathcal{E} \in \mathcal{O}\left(\alpha d\sqrt{\frac{\log(mn)}{n}} + d\sqrt{\frac{\log(mn)}{mn}}\right)$.

Non-Convex Population Risks. For the non-convex population risk case, we need a slightly distinct technical assumption on the size of $\mathcal{W}$.

Assumption 6. Suppose that for all $w \in \mathcal{W}$, $\|\nabla R_{\mathcal{D}}(w)\|_2 \leq G$. We assume that $\mathcal{W}$ contains the ℓ_2 ball centered at the initial parameter w_0

$$\{w \in \mathbb{R}^d : \|w - w_0\|_2 \leq 2\frac{G + \mathcal{E}}{\mathcal{E}^2}[R_{\mathcal{D}}(w_0) - R_{\mathcal{D}}(w^*)]\}.$$

We have the following guarantee on the rate of convergence to a critical point of the population risk $R_{\mathcal{D}}(\cdot)$.

Theorem 3. Suppose Assumption 1, 2, 3, 4 and 6 hold. Choose step size $\eta = \frac{1}{L_R}$ and run Algorithm 1 for $T = \frac{2L_R}{\mathcal{E}^2}[R_{\mathcal{D}}(w_0) - R_{\mathcal{D}}(w^*)]$ rounds, then with probability at least $1 - \frac{1}{(mn)^d}$, we have

$$\min_{t=0,\ldots,T}\|\nabla R_{\mathcal{D}}(w_t)\|_2 \leq \sqrt{2}\mathcal{E},$$

where $\mathcal{E} \in \mathcal{O}\left(\alpha d\sqrt{\frac{\log(mn)}{n}} + d\sqrt{\frac{\log(mn)}{mn}}\right)$.

It can be seen from the results above that, for all the cases, our algorithm achieves an error rate of $\widetilde{\mathcal{O}}\left(d^2\left[\frac{\alpha^2}{n} + \frac{1}{mn}\right]\right)$. The error rate we obtain matches the error rates given in [8]. It has been shown in [8] that, for strongly-convex population risk functions and a fixed d, no algorithm can achieve an error lower than $\widetilde{\Omega}\left(\frac{\alpha^2}{n} + \frac{1}{mn}\right)$, which implies that our algorithm is order-wise optimal in terms of (α, n, m) in this case, even when considering the heavy-tailed data.

C. Analysis of Algorithm 1

Notation: Recall that we denote the Byzantine devices and the good devices in round t as $\mathcal{B}_t$ and $\mathcal{G}_t$ respectively. Moreover, we use $\mathcal{U}_{k,t}$ and $\mathcal{T}_{k,t}$ to denote the untrimmed devices and the trimmed devices with respect to coordinate k in round t. For notation simplicity, we will drop the subscript t when the context is clear.

1) Proof of Lemma 1: To prove Lemma 1, we require the following two lemmas related to local estimators $g_i(\cdot)$'s. The following two Lemmas show that $g_{i,k}(w)$ is concentrated around $\nabla_k R_{\mathcal{D}}(w)$ for all good devices $i \in \mathcal{G}$, any fixed coordinate $k \in [d]$ and any fixed parameter $w \in \mathcal{W}$.

Lemma 2. For any fixed $w \in \mathcal{W}$ and coordinate $k \in [d]$, the following holds with probability at least $1 - m \cdot \zeta$

$$\max_{i \in \mathcal{G}}|g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w)| \leq \sqrt{\frac{2v\log(1/\zeta)}{n}} + \sqrt{\frac{v}{n}}. \quad (5)$$

Lemma 3. For any fixed $w \in \mathcal{W}$ and coordinate $k \in [d]$, the following holds with probability at least $1 - \zeta$

$$\left|\frac{1}{|\mathcal{G}|}\sum_{i \in \mathcal{G}} g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w)\right| \leq \sqrt{\frac{2v\log(1/\zeta)}{(1-\alpha)mn}} + \sqrt{\frac{v}{(1-\alpha)mn}}. \quad (6)$$

Proof of Lemma 2. The one-dimension estimator defined in (2) has the following pointwise accuracy, which is given in [20]:

Lemma 4. [20, Lemma 2] Consider the dataset $\{x_i\}_{i=1}^n$ where x_i are i.i.d. samples drawn from distribution $\mathcal{X}$. Assume that $\mathcal{X}$ has finite second-order moment and $\mathbb{E}_{\mathcal{X}}[|x|^2] \leq v$. Then with probability at least $1 - \zeta$, the estimator $\widehat{x}$ defined in (2) using truncation function defined in (3), noise distribution $\nu = \mathcal{N}\left(0, \frac{1}{\tau}\right)$ and scale $s = \sqrt{\frac{nv}{2\log(1/\zeta)}}$ satisfies

$$|\widehat{x} - \mathbb{E}_{\mathcal{X}}[x]| \leq \sqrt{\frac{2v\log(1/\zeta)}{n}} + \sqrt{\frac{v}{n}}.$$

According to Lemma 4, we have for any $i \in \mathcal{G}$, $k \in [d]$ and $w \in \mathcal{W}$ that, with probability at least $1-\zeta$

$$|g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w)| \leq \sqrt{\frac{2v\log(1/\zeta)}{n}} + \sqrt{\frac{v}{n}}.$$

Then by taking union bound over all good devices $i \in \mathcal{G}$, we have with probability at least $1 - m\cdot\zeta$ that

$$\max_{i\in\mathcal{G}} |g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w)| \leq \sqrt{\frac{2v\log(1/\zeta)}{n}} + \sqrt{\frac{v}{n}},$$

which concludes the proof. □

Proof of Lemma 3. By Lemma 4, we have with probability at least $1-\zeta$ that,

$$\left|\frac{1}{|\mathcal{G}|}\sum_{i\in\mathcal{G}} g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w)\right| \leq \sqrt{\frac{2v\log(1/\zeta)}{|\mathcal{G}|n}} + \sqrt{\frac{v}{|\mathcal{G}|n}}.$$

Since $|\mathcal{G}| \geq (1-\alpha)m$, we obtain with probability at least $1-\zeta$ that

$$\left|\frac{1}{|\mathcal{G}|}\sum_{i\in\mathcal{G}} g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w)\right| \leq \sqrt{\frac{2v\log(1/\zeta)}{(1-\alpha)mn}} + \sqrt{\frac{v}{(1-\alpha)mn}},$$

which concludes the proof. □

With Lemmas 2 and 3, we are now ready to prove Lemma 1.

Due to Lemmas 2 and 3, we already have (5) and (6) hold for any fixed $w \in \mathcal{W}$ and $k \in [d]$. Next, to extend the pointwise accuracy to the uniform accuracy that holds for all $w \in \mathcal{W}$, we need to utilize the standard covering net argument. Let $\mathcal{W}_\epsilon = \{w^1, w^2, \ldots, w^{N_\epsilon}\}$ be a finite subset of $\mathcal{W}$ such that for any $w \in \mathcal{W}$, there exists some $w^p \in \mathcal{W}_\epsilon$ satisfying $\|w - w^p\|_2 \leq \epsilon$. According to the basic property of covering numbers for compact subsets of euclidean space [39], we know that $N_\epsilon \leq \left(\frac{3\Delta}{2\epsilon}\right)^d$. Take the union bound, we have both (5) and (6) hold for any $k \in [d]$ and all $w = w^p \in \mathcal{W}_\epsilon$ with probability at least $1 - N_\epsilon(m+1)\zeta$.

Then consider an arbitrary $w \in \mathcal{W}$. Suppose that $\|w - w^p\|_2 \leq \epsilon$. Since in Assumption 3, we assume that for each $k \in [d]$, $\nabla_k \ell(w,z)$ is L_k-Lipschitz for all data z, we know that

$$|\nabla_k R_{\mathcal{D}}(w) - \nabla_k R_{\mathcal{D}}(w^p)| \leq L_k\epsilon. \tag{7}$$

According to [20, Lemma 4], the one-dimension estimator defined in (2) satisfies that $|\widehat{x}(X) - \widehat{x}(X')| \leq \frac{c_\nu}{n}\|X - X'\|_1$ where X, X' denote two datasets and c_ν is a constant that equals $1 - 2\Phi(-\sqrt{\tau}) + \sqrt{\frac{2}{\tau\pi}}\exp\left(-\frac{\tau}{2}\right)$. For each coordinate $k \in [d]$ we have

$$\begin{aligned}|g_{i,k}(w) - g_{i,k}(w^p)| &\leq \frac{c_\nu}{n}\sum_{j=1}^{n} |\nabla_k \ell(w, z_j) - \nabla_k \ell(w^p, z_j)| \\ &\leq \frac{c_\nu}{n}\cdot n \cdot L_k\|w - w^p\|_2 \leq c_\nu L_k\epsilon, \end{aligned}\tag{8}$$

where the second inequality is due to Assumption 3. Based on (7) and (8), we obtain for any $k \in [d]$ and all $w \in \mathcal{W}$ that, with probability at least $1 - N_\epsilon(m+1)\zeta$

$$\begin{aligned}&\max_{\mathcal{G}} |g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w)| \\ &\leq \sqrt{\frac{2v\log(1/\zeta)}{n}} + \sqrt{\frac{v}{n}} + (1+c_\nu)L_k\epsilon, \end{aligned}\tag{9}$$

and

$$\begin{aligned}&\left|\frac{1}{|\mathcal{G}|}\sum_{i\in\mathcal{G}} g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w)\right| \\ &\leq \sqrt{\frac{2v\log(1/\zeta)}{(1-\alpha)mn}} + \sqrt{\frac{v}{(1-\alpha)mn}} + (1+c_\nu)L_k\epsilon. \end{aligned}\tag{10}$$

We next move on to $g(\cdot)$. We have the following for all $w \in \mathcal{W}$ and any coordinate $k \in [d]$

$$\begin{aligned}&|g_k(w) - \nabla_k R_{\mathcal{D}}(w)| \\ &= \left|\frac{1}{|\mathcal{U}_k|}\sum_{i\in\mathcal{U}_k}(g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w))\right| \\ &\leq \frac{1}{|\mathcal{U}_k|}\left|\sum_{i\in\mathcal{G}}(g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w))\right| \\ &\quad + \frac{1}{|\mathcal{U}_k|}\left|\sum_{i\in\mathcal{G}\cap\mathcal{T}}(g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w))\right| \\ &\quad + \frac{1}{|\mathcal{U}_k|}\left|\sum_{i\in\mathcal{B}\cap\mathcal{U}_k}(g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w))\right|. \end{aligned}\tag{11}$$

We bound each term in (11) respectively. By (10) we have

$$\begin{aligned}&\frac{1}{|\mathcal{U}_k|}\left|\sum_{i\in\mathcal{G}}(g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w))\right| \\ &= \frac{|\mathcal{G}|}{|\mathcal{U}_k|}\left|\frac{1}{|\mathcal{G}|}\sum_{i\in\mathcal{G}}(g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w))\right| \\ &\leq \frac{1-\alpha}{1-2\beta}\left(\sqrt{\frac{2v\log(1/\zeta)}{(1-\alpha)mn}} + \sqrt{\frac{v}{(1-\alpha)mn}} + (1+c_\nu)L_k\epsilon\right). \end{aligned}\tag{12}$$

By (9), we have

$$\begin{aligned}&\frac{1}{|\mathcal{U}_k|}\left|\sum_{i\in\mathcal{G}\cap\mathcal{T}}(g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w))\right| \\ &\leq \frac{2\beta}{1-2\beta}\max_{i\in\mathcal{G}}|g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w)| \\ &\leq \frac{2\beta}{1-2\beta}\left(\sqrt{\frac{2v\log(1/\zeta)}{n}} + \sqrt{\frac{v}{n}} + (1+c_\nu)L_k\epsilon\right). \end{aligned}\tag{13}$$

Since $\beta \geq \alpha$, w.l.o.g., we assume that $\mathcal{G}\cap\mathcal{T}_k \neq \emptyset$. Then by (9) again, we have

$$\begin{aligned}&\frac{1}{|\mathcal{U}_k|}\left|\sum_{i\in\mathcal{B}\cap\mathcal{U}_k}(g_i(w) - \nabla R_{\mathcal{D}}(w))\right| \\ &\leq \frac{\alpha}{1-2\beta}\max_{i\in\mathcal{G}}|g_{i,k}(w) - \nabla_k R_{\mathcal{D}}(w)| \end{aligned}$$

$$\leq \frac{\alpha}{1-2\beta}\left(\sqrt{\frac{2v\log(1/\zeta)}{n}}+\sqrt{\frac{v}{n}}+(1+c_\nu)L_k\epsilon\right). \quad (14)$$

It is worth noting that, all (12), (13) and (14) hold as long as both (9) and (10) hold, which is with probability at least $1-N_\epsilon(m+1)\zeta$. Hence, With the same probability, we have the following for all $w\in\mathcal{W}$ and any $k\in[d]$:

$$\begin{aligned}&|g_k(w)-\nabla_k R_{\mathcal{D}}(w)|\\&\leq \frac{\alpha+2\beta}{1-2\beta}\left(\sqrt{\frac{2v\log(1/\zeta)}{n}}+\sqrt{\frac{v}{n}}+(1+c_\nu)L_k\epsilon\right)\\&\quad+\frac{1-\alpha}{1-2\beta}\left(\sqrt{\tfrac{2v\log(1/\zeta)}{(1-\alpha)mn}}+\sqrt{\tfrac{v}{(1-\alpha)mn}}+(1+c_\nu)L_k\epsilon\right).\end{aligned}$$

Taking the union bound for all $k\in[d]$ yields

$$\begin{aligned}&\|g(w)-\nabla R_{\mathcal{D}}(w)\|_2\\&\leq\sqrt{2}\cdot\left(\frac{\alpha+2\beta}{1-2\beta}\sqrt{\frac{2vd\log(1/\zeta)}{n}}+\frac{\alpha+2\beta}{1-2\beta}\sqrt{\frac{vd}{n}}\right.\\&\quad+\frac{1-\alpha}{1-2\beta}\sqrt{\frac{2vd\log(1/\zeta)}{(1-\alpha)mn}}+\frac{1-\alpha}{1-2\beta}\sqrt{\frac{vd}{(1-\alpha)mn}}\\&\quad\left.+\frac{1+2\beta}{1-2\beta}(1+c_\nu)\widehat{L}\epsilon\right), \quad (15)\end{aligned}$$

which holds for all $w\in\mathcal{W}$ with probability at least $1-N_\epsilon(m+1)d\zeta$. Setting $\epsilon=\frac{3}{2n\widehat{L}}$ and noting that $N_\epsilon=(\Delta n\widehat{L})^d$, $\zeta=\frac{1}{(\Delta n\widehat{L})^d(m+1)d(mn)^d}$, we obtain for all $w\in\mathcal{W}$ that, with probability at least $1-\frac{1}{(mn)^d}$

$$\|g(w)-\nabla R_{\mathcal{D}}(w)\|_2\in\mathcal{O}\left(\alpha d\sqrt{\frac{\log(mn)}{n}}+d\sqrt{\frac{\log(mn)}{mn}}\right). \quad (16)$$

The proof of Theorems 1, 2 and 3 are mainly technical, hence we omit them here and put them in the full version of the paper [40].

IV. Byzantine-Resilient Heavy-Tailed Gradient Descent With Compression

In this section, we study how to further reduce the communication overhead in addition to retaining the Byzantine resilience and the statistical error rate. Based on BHGD, we introduce gradient compression and propose Byzantine-resilient heavy-tailed gradient descent with compression, which is called BHGD-C.

A. Algorithm Design

To reduce the communication cost, we adopt the technique of gradient compression. For the compression scheme, we consider a generic class of compression operators called δ-approximate compressors, just as the recent work [23] did. The formal definition for the compressors is given below.

Definition 5 (δ-Approximate Compressor). An operator $\mathcal{Q}(\cdot):\mathbb{R}^d\to\mathbb{R}^d$ is said to be an δ-approximate compressor on a set $\mathcal{S}\subseteq\mathbb{R}^d$ if $\forall x\in\mathcal{S}$

$$\|\mathcal{Q}(x)-x\|_2^2\leq(1-\delta)\|x\|_2^2,$$

where $\delta\in(0,1]$ is the compression factor.

The compression factor δ measures the degree of compression and $\delta=1$ implies $\mathcal{Q}(x)=x$, which means no compression. There are many compressors satisfying the definition, such as Top-k Sparsification [41], k-PCA [42], Randomized Quantization [43], 1-bit Quantization [29], ℓ_1-norm Quantization [44], etc.

In Algorithm 2, we let each non-Byzantine device compress its estimate for loss gradient by a δ-approximate compressor $\mathcal{Q}(\cdot)$ before sending it to the server (step 3(2)). No restriction is placed on Byzantine devices. Note that, in Algorithm 2, the aggregation rule used by the server is different from that of Algorithm 1. Now the server performs a norm-based trimmed mean (i.e., to trim the gradients according their norm values, see step 4(1)–4(2)). By doing this, the server eliminates only β ($\beta\geq\alpha$) fraction of local gradient estimators instead of 2β as in Algorithm 1. And we believe this will bring a more accurate estimation for the server.

Algorithm 2: Byzantine-Resilient Heavy-Tailed Gradient Descent With Compression (BHGD-C).

Input: Initial parameter vector $w_0\in\mathcal{W}$, compressor $\mathcal{Q}(.)$, step size η, time horizon T.

Initialize: $\zeta\leftarrow\frac{1}{2(\Delta\sqrt{mn})^d d(mn)^d}$, $s\leftarrow\sqrt{\frac{nv}{2\log(1/\zeta)}}$, $\tau\leftarrow\sqrt{2\log(1/\zeta)}$.

1: **for** $t\leftarrow0,1,\ldots,T-1$ **do**

2: Server: Send w_t to all the worker machines

3: Each good device $i\in\mathcal{G}_t$ do in parallel:

1) Computes local estimate of gradient $g_i(w_t)$. Specifically, for $\forall k\in[d]$,

$$g_{i,k}(w_t)\leftarrow\frac{1}{n}\sum_{j=1}^{n}\left[g\left(1-\frac{g^2}{2s^2\tau}\right)-\frac{g^3}{6s^2}\right]+\frac{s}{n}\sum_{j=1}^{n}C\left(\frac{g}{s},\frac{|g|}{s\sqrt{\tau}}\right),$$

where g denotes $\nabla_k\ell(w_t,x_j)$.

2) Sends $\mathcal{Q}(g_i(w_t))$ to the central machine.

4: Server:

1) Sorts $\mathcal{Q}(g_i(w_t))$'s in a non-decreasing order according to $\|\mathcal{Q}(g_i(w_t))\|_2$

2) Denotes the indices of the first $1-\beta$ fraction of elements as $\mathcal{U}_t$

3) Aggregates the gradients through the trimmed mean: $g(w_t)=\frac{1}{|\mathcal{U}_t|}\sum_{i\in\mathcal{U}_t}\mathcal{Q}(g_i(w_t))$

4) Updates the parameter by

$$w_{t+1}=\prod_{\mathcal{W}}(w_t-\eta\cdot g(w_t))$$

5: **end for**

B. Theoretical Results

In this part, we analyse the influence of the gradient compression on the learning performance. Throughout the analysis, we need an additional mild assumption on population risk function $R_{\mathcal{D}}(\cdot)$.

Assumption 7. For all $w \in \mathcal{W}$, $\|\nabla R_{\mathcal{D}}(w)\|_2 \leq G$, where G is a constant.

Note that, while the loss gradients are unbound, it is realistic to assume that the expected gradient (i.e., the population risk gradient) is inside a ball with some radius G.

Before presenting the statistical error rates, We first analyse the uniform accuracy of $g(w)$ for all $w \in \mathcal{W}$.

Lemma 5. With Assumption 7, for all $w \in \mathcal{W}$, it holds with probability at least $1 - \frac{1}{(mn)^d}$ that

$$\|g(w) - \nabla R_{\mathcal{D}}(w)\|_2 \in \mathcal{O}\left((\alpha + \sqrt{1-\delta})d\sqrt{\frac{\log(mn)}{n}} + d\sqrt{\frac{\log(mn)}{mn}}\right). \quad (17)$$

Next, we provide the main results on the error rates.

Strongly Convex Population Risks. Note that the loss function for each data $\ell(\cdot, z)$ need not be strongly convex. The upper bound on the excess population risk is as follows.

Theorem 4. Suppose Assumption 1, 2, 3, 4 and 7 hold, and $R_{\mathcal{D}}(\cdot)$ is λ_R-strongly convex. Choose step size $\eta = \frac{1}{L_R}$ and run Algorithm 2 for T rounds, then with probability at least $1 - \frac{1}{(mn)^d}$, we have the following bound on excess population risk

$$R_{\mathcal{D}}(w_T) - R_{\mathcal{D}}(w^*) \leq L_R\left(1 - \frac{\lambda_R}{L_R + \lambda_R}\right)^{2T}\|w_0 - w^*\|_2^2 + \frac{4L_R}{\lambda_R^2}\widetilde{\mathcal{E}}^2,$$

where $\widetilde{\mathcal{E}} \in \mathcal{O}\left((\alpha + \sqrt{1-\delta})d\sqrt{\frac{\log(mn)}{n}} + d\sqrt{\frac{\log(mn)}{mn}}\right)$.

General Convex Population Risks. We have the following upper bound on excess population risk function.

Theorem 5. Suppose Assumption 1, 2, 3, 4, 5 and 7 hold, and $R_{\mathcal{D}}(\cdot)$ is convex. Choose step size $\eta = \frac{1}{L_R}$ and run Algorithm 1 for $T = \frac{L_R}{\widetilde{\mathcal{E}}}\|w_0 - w^*\|_2$ rounds, then with probability at least $1 - \frac{1}{(mn)^d}$, we have the following bound on excess population risk,

$$R_{\mathcal{D}}(w_T) - R_{\mathcal{D}}(w^*) \leq 16\Delta\widetilde{\mathcal{E}} + \frac{1}{2L_R}\widetilde{\mathcal{E}}^2,$$

where $\widetilde{\mathcal{E}} \in \mathcal{O}\left((\alpha + \sqrt{1-\delta})d\sqrt{\frac{\log(mn)}{n}} + d\sqrt{\frac{\log(mn)}{mn}}\right)$.

Non-Convex Population Risks. For the non-convex population risk case, we need a slightly distinct technical assumption on the size of $\mathcal{W}$.

Assumption 8. The parameter space $\mathcal{W}$ contains the ℓ_2-ball centered at w_0

$$\left\{w \in \mathcal{R}^d : \|w - w_0\|_2 \leq 2\frac{G + \widetilde{\mathcal{E}}}{\widetilde{\mathcal{E}}^2}(R_{\mathcal{D}}(w_0) - R_{\mathcal{D}}(w^*))\right\}.$$

Theorem 6. Suppose Assumption 1, 2, 3, 4, 7 and 8 hold, and $R_{\mathcal{D}}(\cdot)$ is non-convex. Choose step size $\eta = \frac{1}{L_R}$ and run Algorithm 2 for $T = \frac{2L_R}{\widetilde{\mathcal{E}}^2}(R_{\mathcal{D}}(w_0) - R_{\mathcal{D}}(w^*))$ rounds, then with probability at least $1 - \frac{1}{(mn)^d}$, we have

$$\min_{t=0,1,\ldots,T}\|\nabla R_{\mathcal{D}}(w_t)\|_2^2 \leq \sqrt{2}\widetilde{\mathcal{E}},$$

where $\widetilde{\mathcal{E}} \in \mathcal{O}\left((\alpha + \sqrt{1-\delta})d\sqrt{\frac{\log(mn)}{n}} + d\sqrt{\frac{\log(mn)}{mn}}\right)$.

C. Analysis of Algorithm 2

Notation: We denote the Byzantine devices and the good devices in round t as $\mathcal{B}_t$ and $\mathcal{G}_t$ respectively. Also, we use $\mathcal{U}_t$ and $\mathcal{T}_t$ to denote the untrimmed devices and the trimmed devices in round t.

The analysis of algorithm 2 takes Lemma 5 as the core. To prove Lemma 5, we require the following two lemmas, which show that the local estimators $g_i(\cdot)$'s are concentrated around $\nabla R_{\mathcal{D}}(w)$ for all $w \in \mathcal{W}$.

Lemma 6. For all $w \in \mathcal{W}$, the following holds with probability at least $1 - m \cdot (\Delta\sqrt{n})^d d\zeta$

$$\max_{i\in\mathcal{G}}\|g_i(w) - \nabla R_{\mathcal{D}}(w)\|_2 \leq \frac{3(c_v\widehat{L} + L_R)}{2\sqrt{n}} + \sqrt{\frac{2vd\log(\zeta^{-1})}{n}} + \sqrt{\frac{v}{n}} \triangleq \mathcal{E}_1 \quad (18)$$

Lemma 7. For all $w \in \mathcal{W}$, the following holds with probability at least $1 - (\Delta\sqrt{mn})^d d\zeta$

$$\left\|\frac{1}{|\mathcal{G}|}\sum_{i\in\mathcal{G}} g_i(w) - g(w)\right\|_2 \leq \frac{3(c_v\widehat{L} + L_R)}{2\sqrt{mn}} + \sqrt{\frac{2vd\log(\zeta^{-1})}{mn}} + \sqrt{\frac{v}{mn}} \triangleq \mathcal{E}_2 \quad (19)$$

Proof of Lemma 6. According to [20, Lemma 5], we know that for any fixed $i \in \mathcal{G}$ and all $w \in \mathcal{W}$, the following holds with probability at least $1 - N_\epsilon d\zeta$

$$\|g_i(w) - \nabla R_{\mathcal{D}}(w)\|_2 \leq (c_v\widehat{L} + L_R)\epsilon + \sqrt{\frac{2vd\log(\zeta^{-1})}{n}} + \sqrt{\frac{v}{n}},$$

where $N_\epsilon \leq \left(\frac{3\Delta}{2\epsilon}\right)^d$. By setting $\epsilon = \frac{3}{2\sqrt{n}}$, we obtain that with probability at least $1 - (\Delta\sqrt{n})^d d\zeta$

$$\|g_i(w) - \nabla R\|_2 \leq \frac{3(c_v\widehat{L} + L_R)}{2\sqrt{n}} + \sqrt{\frac{2vd\log(\zeta^{-1})}{n}} + \sqrt{\frac{v}{n}}.$$

Take the union bound, we know that with probability at least $1 - m \cdot (\Delta\sqrt{n})^d d\zeta$, the following holds for all $w \in \mathcal{W}$

$$\max_{i\in\mathcal{G}}\|g_i(w) - \nabla R_{\mathcal{D}}(w)\|_2 \leq \frac{3(c_v\widehat{L} + L_R)}{2\sqrt{n}} + \sqrt{\frac{2vd\log(\zeta^{-1})}{n}} + \sqrt{\frac{v}{n}}.$$

□

Proof of Lemma 7. Following the same argument in the proof of Lemma 6 (except for taking $\epsilon = \frac{3}{2\sqrt{mn}}$), we have that, with probability at least $1 - (\Delta\sqrt{mn})^d d\zeta$, the following holds for

all $w \in \mathcal{W}$

$$\left\| \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} g_i(w) - \nabla R_{\mathcal{D}}(w) \right\|_2 \leq \frac{3(c_v \widehat{L} + L_R)}{2\sqrt{mn}} + \sqrt{\frac{2vd \log(\zeta^{-1})}{mn}} + \sqrt{\frac{v}{mn}}.$$

□

With Lemmas 6 and 7, we are now ready to prove Lemma 5. For notation simplicity, we denote $\|g(w) - \nabla R_{\mathcal{D}}(w)\|_2$ by $\widetilde{\mathcal{E}}(w)$. First, we take union bound, then with probability at least $1 - m \cdot (\Delta\sqrt{n})^d d\zeta - (\Delta\sqrt{mn})^d d\zeta \geq 1 - 2(\Delta\sqrt{mn})^d d\zeta$, (18) and (19) hold simultaneously. Conditioned on this, we next proceed to bound $\widetilde{\mathcal{E}}(w)$. Specifically, we have the following holds for all $w \in \mathcal{W}$

$$\begin{aligned}
\widetilde{\mathcal{E}}(w) &= \left\| \frac{1}{|\mathcal{U}_t|} \sum_{i \in \mathcal{U}_t} \mathcal{Q}[g_i(w)] - \nabla R_{\mathcal{D}}(w) \right\|_2 \\
&= \frac{1}{|\mathcal{U}_t|} \Bigg\| \sum_{i \in \mathcal{G}} (\mathcal{Q}[g_i(w)] - \nabla R_{\mathcal{D}}(w)) \\
&\quad - \sum_{i \in \mathcal{G} \cap \mathcal{T}_t} (\mathcal{Q}[g_i(w_t)] - \nabla R_{\mathcal{D}}(w)) \\
&\quad + \sum_{i \in \mathcal{B} \cap \mathcal{U}_t} (\mathcal{Q}[g_i(w)] - \nabla R_{\mathcal{D}}(w)) \Bigg\|_2 \\
&\leq \frac{1}{|\mathcal{U}_t|} \left\| \sum_{i \in \mathcal{G}} (\mathcal{Q}[g_i(w)] - \nabla R_{\mathcal{D}}(w)) \right\|_2 \\
&\quad + \frac{1}{|\mathcal{U}_t|} \left\| \sum_{i \in \mathcal{G} \cap \mathcal{T}_t} (\mathcal{Q}[g_i(w)] - \nabla R_{\mathcal{D}}(w)) \right\|_2 \\
&\quad + \frac{1}{|\mathcal{U}_t|} \left\| \sum_{i \in \mathcal{B} \cap \mathcal{U}_t} (\mathcal{Q}[g_i(w)] - \nabla R_{\mathcal{D}}(w)) \right\|_2 . \qquad (20)
\end{aligned}$$

We control each term in (20) separately. For the first term in (20), we have

$$\begin{aligned}
&\frac{1}{|\mathcal{U}_t|} \left\| \sum_{i \in \mathcal{G}} (\mathcal{Q}[g_i(w)] - \nabla R_{\mathcal{D}}(w)) \right\|_2 \\
&\leq \frac{1}{|\mathcal{U}_t|} \left\| \sum_{i \in \mathcal{G}} (\mathcal{Q}[g_i(w)] - g_i(w)) \right\|_2 \\
&\quad + \frac{1}{|\mathcal{U}_t|} \left\| \sum_{i \in \mathcal{G}} (g_i(w) - \nabla R_{\mathcal{D}}(w)) \right\|_2 \\
&\leq \frac{1}{|\mathcal{U}_t|} \sum_{i \in \mathcal{G}} (\|Q[g_i(w)] - g_i(w)\|_2) + \frac{1-\alpha}{1-\beta} \mathcal{E}_2 \\
&\leq \frac{1}{|\mathcal{U}_t|} \sum_{i \in \mathcal{G}} (\sqrt{1-\delta} \|g_i(w)\|_2) + \frac{1-\alpha}{1-\beta} \mathcal{E}_2 \\
&\leq \frac{\sqrt{1-\delta}}{|\mathcal{U}_t|} \sum_{i \in \mathcal{G}} (\|\nabla R_{\mathcal{D}}(w)\|_2 \\
&\quad + \|g_i(w) - \nabla R_{\mathcal{D}}(w)\|_2) + \frac{1-\alpha}{1-\beta} \mathcal{E}_2 \\
&\leq \frac{\sqrt{1-\delta}(1-\alpha)}{1-\beta} G + \frac{\sqrt{1-\delta}(1-\alpha)}{1-\beta} \mathcal{E}_1 + \frac{1-\alpha}{1-\beta} \mathcal{E}_2 \qquad (21)
\end{aligned}$$

Similarly, we bound the second term in (20) as follows:

$$\begin{aligned}
&\frac{1}{|\mathcal{U}_t|} \left\| \sum_{i \in \mathcal{G} \cap \mathcal{T}_t} (\mathcal{Q}[g_i(w)] - \nabla R_{\mathcal{D}}(w)) \right\|_2 \\
&\leq \frac{|\mathcal{T}_t|}{|\mathcal{U}_t|} \max_{i \in \mathcal{G}} \|\mathcal{Q}[g_i(w)] - \nabla R_{\mathcal{D}}(w)\|_2 \\
&\leq \frac{\beta}{1-\beta} \max_{i \in \mathcal{G}} (\|\mathcal{Q}[g_i(w)] - g_i(w)\|_2 + \|g_i(w) - \nabla R_{\mathcal{D}}(w)\|_2) \\
&\leq \frac{\beta}{1-\beta} \max_{i \in \mathcal{G}} (\sqrt{1-\delta} \|g_i(w)\|_2 + \|g_i(w) - \nabla R_{\mathcal{D}}(w)\|_2) \\
&\leq \frac{\beta}{1-\beta} (\sqrt{1-\delta} \|\nabla R_{\mathcal{D}}(w)\|_2 \\
&\quad + (1 + \sqrt{1-\delta}) \max_{i \in \mathcal{G}} \|g_i(w) - \nabla R_{\mathcal{D}}(w)\|_2) \\
&\leq \frac{\beta\sqrt{1-\delta}}{1-\beta} G + \frac{\beta(1+\sqrt{1-\delta})}{1-\beta} \mathcal{E}_1. \qquad (22)
\end{aligned}$$

Finally, we work on the third term in (20). Owing to the trimming threshold $\beta > \alpha$, we have at least one good device machine in the set $\mathcal{T}_t$ for all $t \in [T]$

$$\begin{aligned}
&\frac{1}{|\mathcal{U}_t|} \left\| \sum_{i \in \mathcal{B} \cap \mathcal{U}_t} (\mathcal{Q}[g_i(w)] - \nabla R_{\mathcal{D}}(w)) \right\|_2 \\
&\leq \frac{|\mathcal{B}|}{|\mathcal{U}_t|} \max_{i \in \mathcal{B} \cap \mathcal{U}_t} \|\mathcal{Q}[g_i(w)] - \nabla R_{\mathcal{D}}(w)\|_2 \\
&\leq \frac{\alpha}{1-\beta} \max_{i \in \mathcal{B} \cap \mathcal{U}_t} (\|\mathcal{Q}[g_i(w)]\|_2 + \|\nabla R_{\mathcal{D}}(w)\|_2) \\
&\leq \frac{\alpha}{1-\beta} \max_{i \in \mathcal{B} \cap \mathcal{U}_t} \left(\sqrt{1-\delta} \|g_i(w)\|_2 + \|g_i(w)\|_2 + \|\nabla R_{\mathcal{D}}(w)\|_2 \right) \\
&\leq \frac{\alpha}{1-\beta} ((1 + \sqrt{1-\delta}) \mathcal{E}_1 + (2 + \sqrt{1-\delta}) \|\nabla R_{\mathcal{D}}(w)\|_2) \\
&\leq \frac{\alpha(2+\sqrt{1-\delta})}{1-\beta} G + \frac{\alpha(1+\sqrt{1-\delta})}{1-\beta} \mathcal{E}_1. \qquad (23)
\end{aligned}$$

Combining (21), (22), (21) and letting $\widetilde{\mathcal{E}}$ be the uniform upper bound on $\widetilde{\mathcal{E}}(w)$ over $w \in \mathcal{W}$, we obtain that with probability at least $1 - 2(\Delta\sqrt{mn})^d d\zeta$, we have the following holds for all $w \in \mathcal{W}$

$$\begin{aligned}
\widetilde{\mathcal{E}} \leq\ & \frac{(1+\beta)\sqrt{1-\delta} + 2\alpha}{1-\beta} G \\
& + \frac{(1+\beta)\sqrt{1-\delta} + \alpha + \beta}{1-\beta} \mathcal{E}_1 + \frac{1-\alpha}{1-\beta} \mathcal{E}_2.
\end{aligned}$$

Note that $\zeta = \frac{1}{2(\Delta\sqrt{mn})^d d(mn)^d}$, hence Lemma 5 follows.

With Lemma 5, we can prove Theorem 4, 5 and 6 by replacing $\mathcal{E}$ with $\widetilde{\mathcal{E}}$ and then following almost the same arguments as in the proof of Theorem 1, 2 and 3. Thus the details are omitted here.

V. Experiments

We now conduct experiments on both synthetic and real-world data to validate the efficacy of our algorithms.

A. Experiment Setup

We will study tasks of linear regression and logistic regression on both synthetic and real-world datasets. The synthetic data is generated as follows. For linear model, we generate each data point (x, y) by $y = \langle x, w^* \rangle + \xi$, where $\xi \in \mathbb{R}$ is zero-mean noise sampled from $\mathrm{LogNormal}(0, 0.55848)$ by default and each coordinate of $x \in \mathbb{R}^d$ is sampled from $\mathrm{LogNormal}(0, 0.78)$ by default. For logistic model, we generate each data point (x, y) by $y = \mathrm{sign}[\mathrm{sigmoid}(z) - \frac{1}{2}]$, where $\mathrm{sigmoid}(z) = \frac{1}{1+\exp(-z)}$ and $z = \langle x, w^* \rangle + \xi$ where $\xi \in \mathbb{R}$ is zero-mean noise sampled from $\mathrm{LogNormal}(0, 0.55848)$ by default and each coordinate of $x \in \mathbb{R}^d$ is sample from $\mathrm{LogNormal}(0, 3)$ by default. For real-world datasets, we will use the Adult dataset [45] for a binary classification task and the Boston Housing Price dataset[4] for a linear regression task. We use these datasets since they are representative and have been used in previous works on machine learning with heavy-tailed data e.g., [35]. In particular, to test our algorithms for stochastic non-convex optimization, we train a fully-connected neural network with one hidden layer to solve the tasks on real-world datasets.

Since the precise evaluation of the population risk, i.e., $R_{\mathcal{D}}(w) - R_{\mathcal{D}}(w^*)$, is impossible, we will use the empirical risk to approximate the population risk. Specifically, we measure the performance of different algorithm in terms of test loss, i.e., excess empirical risk on the test dataset. For all experiments, we run each algorithm for at least 10 times and report the average results over all the repetitions.

B. Results and Discussion

1) Comparison With Baseline Methods: To show the Byzantine-resilience of BHGD (Algorithm 1), we compare the performance of BHGD with previous methods. Specifically, we consider the distributed gradient descent algorithms with averaging rules being empirical mean (E-Mean) [1], coordinate-wise trimmed mean (CWT-Mean) [8], coordinate-wise median (CW-Median) [8], geometric median (G-Median) [22], Krum [31], Bulyan [32], and momentum Krum (M-Krum) [10] respectively. For experiments on synthetic datasets, we set $m = 10$, $n = 100$, $d = 10$, $\alpha = 0.2$ and generate 200 test data for each model. For experiments on real-world datasets, we set $m = 10$, $n = 40$, $\alpha = 0.2$ and $m = 5$, $n = 10$, $\alpha = 0.2$ for Boston dataset and Adult dataset respectively and pick 100 data uniformly at random as the test data for each. The experimental results are shown in Fig. 1. As we can see from Fig. 1, BHGD (Algorithm 1) achieves lower test loss than all the baseline methods on both synthetic and real-world datasets. In addition, the convergence of BHGD is stabler than the other methods, which indicates that BHGD is more robust to heavy-tailed data as well as Byzantine devices.

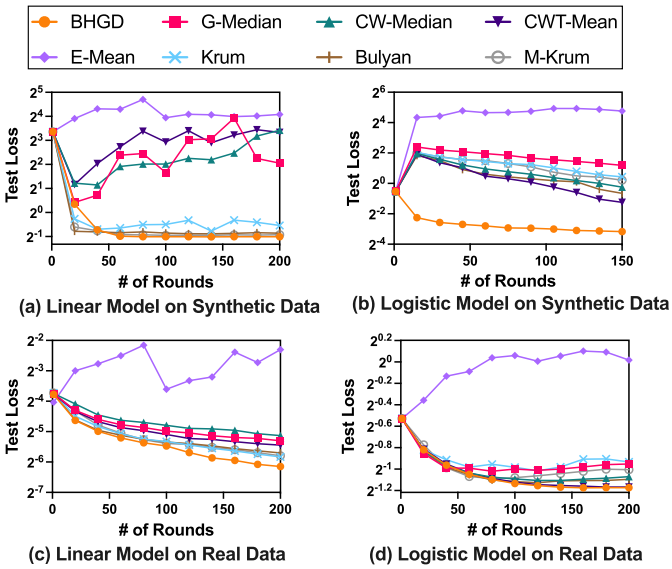

Fig. 1. Comparison with Baseline Methods.

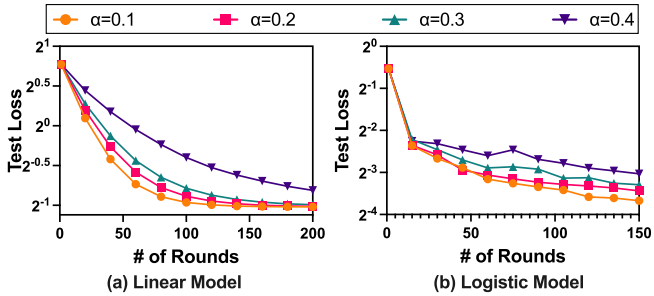

Fig. 2. Impact of Byzantine Devices.

2) Impact of Byzantine Devices: We conduct experiments on synthetic datasets to study the impact of Byzantine devices. We present the performance of BHGD (Algorithm 1) in presence of different fractions of Byzantine devices in Fig. 2, where α denotes the fraction of Byzantine devices among all learning devices. Our BHGD algorithm converge in all cases, which confirms that BHGD can tolerate various fraction of Byzantine devices. Moreover, the convergence is subject to the fraction of Byzantine devices. As shown in Fig. 2, a larger fraction of Byzantine devices may not only cause a larger excess empirical risk, but also slow the convergence down.

3) Impact of Heavy-Tailed Distributions: We conduct experiments on synthetic datasets with different types of heavy-tailed label noises and different tail weights of heavy-tailed feature vectors to show the impact of heavy-tailed distributions on the learning performance of BHGD algorithm. First, we adopt LogNormal distribution and Pareto Distribution to generate the label noises. The main difference between these two heavy-tailed

4.http://lib.stat.cmu.edu/datasets/boston

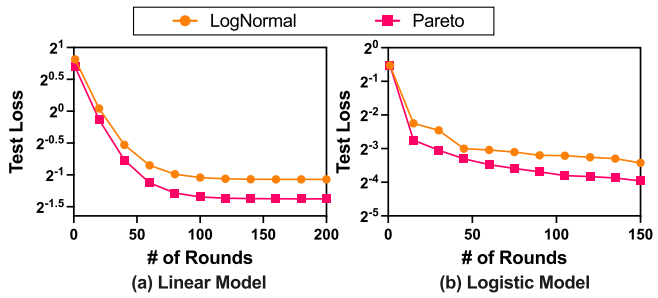

Fig. 3. Performance under Different Heavy-tailed Label Noises.

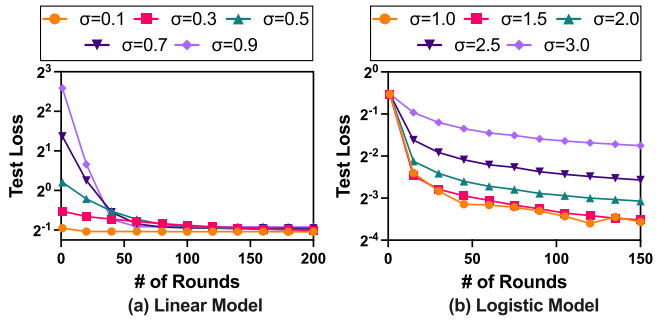

Fig. 4. Performance under Different Tail Weights of Heavy-Tailed Feature Vectors.

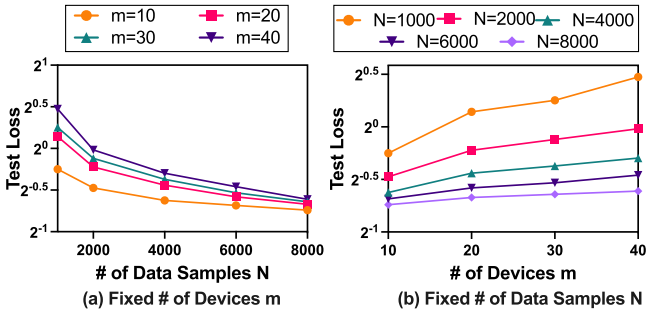

Fig. 5. Impact of the System Scale.

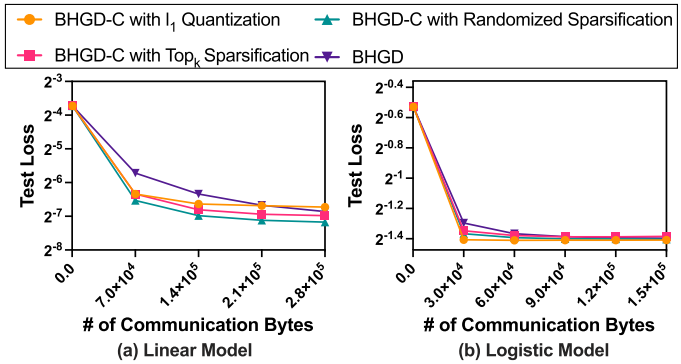

Fig. 6. Impact of Gradient Compression.

distributions is that LogNormal distribution is symmetric while Pareto distribution is one-sided. For Pareto label noise in each task, we set the scale parameter to be 1 and the shape parameter to be 3.26953 such that its second moment is identical to that of $\mathrm{LogNormal}(0, 0.55848)$. The experimental results are shown in Fig. 3. It can be seen that, BHGD converges under both types of noises, and performs better on Pareto noises. We think the main reason lies in that Pareto noise is one-sided and thus its influence is smaller. Next, we vary the tail weight of heavy-tailed feature vector. We adopt the LogNormal distribution $\mathrm{LogNormal}(\mu, \sigma)$, where μ is set to be 0 and σ varies in different settings. Note that, a larger σ corresponds to a larger variance. For linear model, we let σ vary in $\{0.1, 0.3, 0.5, 0.7, 0.9\}$, while for logistic model, we let σ vary in $\{1.0, 1.5, 2.0, 2.5, 3.0\}$. The experimental results are shown in Fig. 4. From these results, we can conclude that more test loss would be incurred when the training data become more heavy-tailed.

4) Impact of System Scale: We study the impact of system scale on the performance of BHGD. Specifically, we consider the scale of learning devices and the scale of training data samples. In this experiment, we let m (i.e., the number of learning devices) vary in $\{10, 20, 30, 40\}$ and let N (i.e., the number of training data samples) vary in $\{1000, 2000, 4000, 6000, 8000\}$. We run the BHGD algorihtm on synthetic dataset for logistic model under each combination of m and N. The results are shown in Fig. 5. In Fig. 5 (a), we fix m and present the test loss after 200 training rounds for various N. Since $N = mn$, when m is fixed, a larger N means a larger size of local data sample n for each device. The results in Fig. 5 (a) shows that when each learning device has more training data, the test loss would get smaller. Intuitively, this means that when each device has more local data, it learns more accurately, which is quite reasonable. In Fig. 5 (b), we fix N and present the test loss after 200 training rounds for various m.When N is fixed, a larger m means a smaller size of local data n for each device. Hence the test loss grows larger as m increases in Fig. 5 (b). Note that, these results corroborate our previous theoretical upper bounds on statistical error rates.

5) Communication Efficiency: We show the communication efficiency of our BHGD-C algorithm (Algorithm 2).

In the first part, we study the impact of different gradient compression techniques. To this end, we equip BHGD-C with different compressor, and then compare their performances with BHGD on real-world datasets. Specifically, we adopt ℓ_1 quantization [44], Top_k sparsification [41] and randomized sparsification [46]. In ℓ_1 quantization, the compressor $Q(\cdot)$ can be summarized as $\mathcal{Q}(x) = \{\frac{\|x\|_1}{d}, \mathrm{sign}(x)\}$ for $\forall x \in \mathbb{R}^d$, where $\mathrm{sign}(x)$ is the quantized vector with each coordinate $i \in [d]$ being either $+1$ (for positive x_i) or -1 (for negative x_i) and $\frac{\|x\|_1}{d}$ is the scaling factor. In Top_k sparsification, for any $x \in \mathbb{R}^d$, the compressor compresses x by retaining the top k largest coordinates of x and sets the others to zero. In randomized sparsification, for any $x \in \mathbb{R}^d$ and each coordinate $i \in [d]$, the compressor set x_i to 1 with probability p and to 0 with probability $1 - p$. The experimental results are shown in Fig. 6. Note that, in Fig. 6, we present how the test loss varies with respect to the total communication costs (in byte) during the training process. It can be seen that, to achieve the same test loss, BHGD-C entails

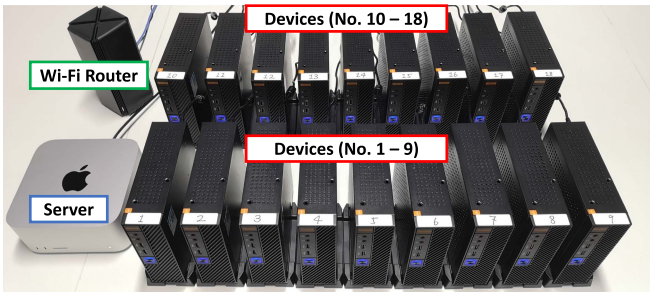

Fig. 7. The hardware prototype system for edge federated learning.

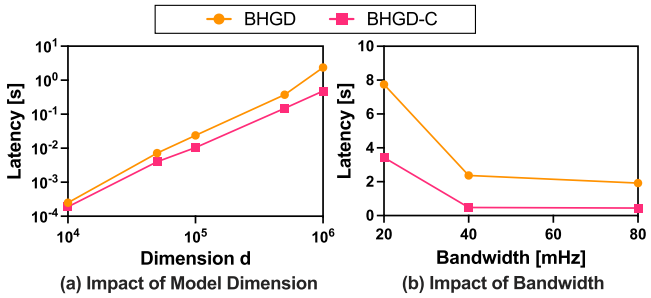

Fig. 8. Transmission Latency in the Real System.

fewer bytes to transmit from devices to the sever and thus provide communication efficiency.

Next, in the second part, we conduct experiment on a networked hardware prototype system for edge federated learning. Our system, as illustrated in Fig. 7, consists of $m = 18$ Teclast mini PCs serving as learning devices and a Mac Studio serving as the central server. All the machines are interconnected via a Wi-Fi router and the bandwidth is set to 40mHz by default. We implement our BHGD and BHGD-C algorithms on the system, where BHGD-C algorithm use the Top_k sparsification with k being half of the model dimension. We present the average communication latency across all the learning devices and all the training rounds in Fig. 8. Overall, the BHGD-C incurs less latency during the learning process. From Fig. 8 (a), we can see that, with the model dimension increases, the latency saved by BHGD-C also grows. From Fig. 8 (b), we can see that, the bandwidth affects the communication latency a lot and BHGD-C achieves a much lower latency than BHGD even when the bandwidth is limited.

VI. Conclusion

In this paper, we studied how to retain Byzantine resilience and communication efficiency when training machine learning model with heavy-tailed data in a distributed manner. Specifically, we first presented an algorithm that is robust against both Byzantine worker nodes and heavy-tailed data. Then by adopting the gradient compression technique, we further proposed a robust learning algorithm with reduced communication overhead. Our theoretical analysis demonstrated that our proposed algorithms achieve optimal error rates for strongly convex population risk case. We also conducted extensive experiments, which yield consistent results with the theoretical analysis. It is worth noting that, our results may not have optimal dependence on the dimension d. Some recent work also attempt to incorporate recent breakthroughs in robust high-dimensional aggregators into the distributed learning scenarios, e.g., [47], [48], [49], [50]. How to achieve Byzantine resilience, communication efficiency and heavy-tailed data robustness simultaneously in high dimensions is an interesting and significant problem and we leave it as the future work.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, 2017, pp. 1273–1282.
[2] J. Zhang et al., "Adaptive federated learning on non-IID data with resource constraint," *IEEE Trans. Comput.*, vol. 71, no. 7, pp. 1655–1667, Jul. 2022.
[3] L. Feng, Y. Zhao, S. Guo, X. Qiu, W. Li, and P. Yu, "BAFL: A blockchain-based asynchronous federated learning framework," *IEEE Trans. Comput.*, vol. 71, no. 5, pp. 1092–1103, May 2022.
[4] W. Wu, L. He, W. Lin, R. Mao, C. Maple, and S. A. Jarvis, "SAFA: A semi-asynchronous protocol for fast federated learning with low overhead," *IEEE Trans. Comput.*, vol. 70, no. 5, pp. 655–668, May 2021.
[5] L. Lamport, R. E. Shostak, and M. C. Pease, "The byzantine generals problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, 1982.
[6] G. S. Veronese, M. Correia, A. N. Bessani, L. C. Lung, and P. Verissimo, "Efficient byzantine fault-tolerance," *IEEE Trans. Comput.*, vol. 62, no. 1, pp. 16–30, Jan. 2013.
[7] T. Distler, C. Cachin, and R. Kapitza, "Resource-efficient byzantine fault tolerance," *IEEE Trans. Comput.*, vol. 65, no. 9, pp. 2807–2819, Sep. 2016.
[8] D. Yin, Y. Chen, K. Ramchandran, and P. L. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 5650–5659.
[9] S. Farhadkhani, R. Guerraoui, N. Gupta, R. Pinot, and J. Stephan, "Byzantine machine learning made easy by resilient averaging of momentums," in *Proc. 39th Int. Conf. Mach. Learn.*, 2022, pp. 6246–6283.
[10] E. M. El Mhamdi, R. Guerraoui, and S. L. A. Rouault, "Distributed momentum for byzantine-resilient stochastic gradient descent," in *Proc. 9th Int. Conf. Learn. Representations*, 2021, pp. 1–37.
[11] D. Data and S. Diggavi, "Byzantine-resilient SGD in high dimensions on heterogeneous data," in *Proc. IEEE Int. Symp. Inf. Theory*, 2021, pp. 2310–2315.
[12] Y. Shi, K. Yang, T. Jiang, J. Zhang, and K. B. Letaief, "Communication-efficient edge ai: Algorithms and systems," *IEEE Commun. Surv. Tuts.*, vol. 22, no. 4, pp. 2167–2191, Fourth Quarter 2020.
[13] L. Yan, C. Di, Q. J. Wu, and Y. Xia, "Sequential fusion for multirate multisensor systems with heavy-tailed noises and unreliable measurements," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 52, no. 1, pp. 523–532, Jan. 2022.
[14] L. Zhao, X. Cao, L. Li, and H. Yang, "Event-triggered distributed fusion for multirate multisensor systems with heavy-tailed noises," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 52, no. 5, pp. 3137–3150, May 2022.
[15] H. Zhu, K. Zou, Y. Li, and H. Leung, "Robust sensor fusion with heavy-tailed noises," *Signal Process.*, vol. 175, 2020, Art. no. 107659.
[16] R. F. Woolson and W. R. Clarke, *Statistical Methods for the Analysis of Biomedical Data*. Hoboken, NJ, USA: Wiley, 2011.
[17] A. Biswas, S. Datta, J. P. Fine, and M. R. Segal, *Statistical Advances in the Biomedical Sciences: Clinical Trials, Epidemiology, Survival Analysis, and Bioinformatics*. Hoboken, NJ, USA: Wiley, 2007.
[18] M. Ibragimov, R. Ibragimov, and J. Walden, *Heavy-Tailed Distributions and Robustness in Economics and Finance*. Berlin, Germany: Springer, 2015, vol. 214.
[19] B. O. Bradley and M. S. Taqqu, "Financial risk and heavy tails," in *Handbook of Heavy Tailed Distributions in Finance*, vol. 1, Amsterdam, The Netherlands: North-Holland, 2003, pp. 35–103.
[20] M. J. Holland, "Robust descent using smoothed multiplicative noise," in *Proc. 22nd Int. Conf. Artif. Intell. Statist.*, 2019, pp. 703–711.
[21] M. J. Holland and K. Ikeda, "Efficient learning with robust gradient descent," *Mach. Learn.*, vol. 108, no. 8, pp. 1523–1560, 2019.
[22] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 1, no. 2, pp. 44:1–44:25, 2017.

[23] A. Ghosh, R. K. Maity, S. Kadhe, A. Mazumdar, and K. Ramchandran, "Communication-efficient and byzantine-robust distributed learning with error feedback," *IEEE J. Sel. Areas Inf. Theory*, vol. 2, no. 3, pp. 942–953, Sep. 2021.
[24] S. P. Karimireddy, L. He, and M. Jaggi, "Learning from history for byzantine robust optimization," in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 5311–5319.
[25] Y. Dong, S. B. Hopkins, and J. Li, "Quantum entropy scoring for fast robust mean estimation and improved outlier detection," in *Proc. 32nd Conf. Neural Inf. Process. Syst.*, 2019, pp. 6065–6075.
[26] I. Diakonikolas, D. M. Kane, and A. Pensia, "Outlier robust mean estimation with subgaussian rates via stability," in *Proc. 34th Conf. Neural Inf. Process. Syst.*, 2020, pp. 1830–1840.
[27] L. Hu and O. Reingold, "Robust mean estimation on highly incomplete data with arbitrary outliers," in *Proc. 24th Int. Conf. Artif. Intell. Statist.*, 2021, pp. 1558–1566.
[28] K. Pillutla, S. M. Kakade, and Z. Harchaoui, "Robust aggregation for federated learning," *IEEE Trans. Signal Process.*, vol. 70, pp. 1142–1154, 2022.
[29] J. Bernstein, Y. Wang, K. Azizzadenesheli, and A. Anandkumar, "SIGNSGD: Compressed optimisation for non-convex problems," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 559–568.
[30] R. Jin, Y. Huang, X. He, H. Dai, and T. Wu, "Stochastic-sign SGD for federated learning with theoretical guarantees," 2020. [Online]. Available: https://arxiv.org/abs/2002.10940
[31] P. Blanchard, E. M. E. Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proc. 31st Conf. Neural Inf. Process. Syst.*, 2017, pp. 118–128.
[32] R. Guerraoui et al., "The hidden vulnerability of distributed learning in byzantium," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 3518–3527.
[33] U. Simsekli, L. Sagun, and M. Gurbuzbalaban, "A tail-index analysis of stochastic gradient noise in deep neural networks," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 5827–5837.
[34] J. Bernstein, J. Zhao, K. Azizzadenesheli, and A. Anandkumar, "signSGD with majority vote is communication efficient and fault tolerant," in *Proc. 7th Int. Conf. Learn. Representations*, 2019, pp. 1–20.
[35] D. Wang, H. Xiao, S. Devadas, and J. Xu, "On differentially private stochastic convex optimization with heavy-tailed data," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 10081–10091.
[36] L. Hu, S. Ni, H. Xiao, and D. Wang, "High dimensional differentially private stochastic optimization with heavy-tailed data," in *Proc. 41st ACM SIGMOD-SIGACT-SIGAI Symp. Princ. Database Syst.*, 2022, pp. 227–236.
[37] G. Kamath, X. Liu, and H. Zhang, "Improved rates for differentially private stochastic convex optimization with heavy-tailed data," in *Proc. 39th Int. Conf. Mach. Learn.*, 2022, pp. 10633–10660.
[38] O. Catoni and I. Giulini, "Dimension-free PAC-Bayesian bounds for matrices, vectors, and linear least squares regression," 2017. [Online]. Available: https://arxiv.org/abs/1712.02747
[39] V. M. Tikhomirov, *Entropy and -Capacity of Sets in Functional Spaces*. Dordrecht, Netherlands: Springer, 1993, pp. 86–170.
[40] Y. Tao et al., "Byzantine-resilient federated learning at edge," 2023. [Online]. Available: https://arxiv.org/abs/2303.10434
[41] S. U. Stich, J. Cordonnier, and M. Jaggi, "Sparsified SGD with memory," in *Proc. 32nd Conf. Neural Inf. Process. Syst.*, 2018.
[42] H. Wang, S. Sievert, S. Liu, Z. B. Charles, D. S. Papailiopoulos, and S. J. Wright, "ATOMO: Communication-efficient learning via atomic sparsification," in *Proc. 32nd Conf. Neural Inf. Process. Syst.*, 2018, pp. 4452–4463.
[43] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "QSGD: Communication-efficient SGD via gradient quantization and encoding," in *Proc. 31st Conf. Neural Inf. Process. Syst.*, 2017, pp. 1709–1720.
[44] S. P. Karimireddy, Q. Rebjock, S. U. Stich, and M. Jaggi, "Error feedback fixes signsgd and other gradient compression schemes," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 3252–3261.
[45] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml
[46] J. Wangni, J. Wang, J. Liu, and T. Zhang, "Gradient sparsification for communication-efficient distributed optimization," in *Proc. 32nd Conf. Neural Inf. Process. Syst.*, 2018, pp. 1306–1316.
[47] I. Diakonikolas, G. Kamath, D. Kane, J. Li, J. Steinhardt, and A. Stewart, "Sever: A Robust Meta-Algorithm for Stochastic Optimization," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2019, pp. 1596–1606.
[48] L. Su and J. Xu, "Securing distributed gradient descent in high dimensional statistical learning," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 3, no. 1, pp. 12:1–12:41, 2019.
[49] E.-M. El-Mhamdi, R. Guerraoui, and S. Rouault, "Fast and robust distributed learning in high dimension," in *Proc. 39th Int. Symp. Reliable Distrib. Syst.*, 2020, pp. 71–80.
[50] D. Data and S. N. Diggavi, "Byzantine-resilient high-dimensional SGD with local iterations on heterogeneous data," in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 2478–2488.

Youming Tao (Student Member, IEEE) received the bachelor's degree (with distinction) in computer science from Shandong University, in 2021, where he is currently working toward the PhD degree in computer science. He is interested in building private, efficient and reliable collaborative learning algorithms with applications to edge intelligence.

Sijia Cui received the BE degree in computer science from Shandong University, in 2022. He is currently working toward the PhD degree in computer science with the University of Chinese Academy of Sciences. He is interested in reinforcement learning.

Wenlu Xu received the bachelor's degree in mathematics from Shandong University. She is currently working toward the PhD degree with the Department of Statistics, University of California, Los Angeles (UCLA). She is under the guidance of Dr. Xiaowu Dai. Her research interests include recommender system, causal inference and machine learning in economics. She is currently a member of the Lab for Statistics, Computing, Algorithm, Learning, and Economics (SCALE).

Haofei Yin received the BE degree in computer science from Shandong University, in 2022. He is currently working toward the master's degree in computer science with Shandong University. He is interested in computer networking.

Dongxiao Yu (Senior Member, IEEE) received the BSc degree from the School of Mathematics, Shandong University, in 2006, and the PhD degree from the Department of Computer Science, The University of Hong Kong, in 2014. He became an associate professor with the School of Computer Science and Technology, Huazhong University of Science and Technology, in 2016. He is currently a professor with the School of Computer Science and Technology, Shandong University. His research interests include wireless networks, distributed computing, and graph algorithms.

Weifa Liang (Senior Member, IEEE) received the BSc degree in computer science from Wuhan University, China, in 1984, and the ME and PhD degrees in computer science from Australian National University, in 1989 and 1998, respectively. He is currently a professor with the Department of Computer Science, City University of Hong Kong. Prior to that, he was a professor with the Australian National University. His research interests include design and analysis of energy efficient routing protocols for wireless ad hoc and sensor networks, mobile edge computing (MEC), network function virtualization (NFV), Internet of Things, software-defined networking (SDN), design and analysis of parallel and distributed algorithms, approximation algorithms, combinatorial optimization, and graph theory. He is currently an associate editor for the editorial board of *IEEE Transactions on Communications*.

Xiuzhen Cheng (Fellow, IEEE) received the MS and PhD degrees in computer science from the University of Minnesota, Twin Cities, in 2000 and 2002, respectively. She was a faculty member with the Department of Computer Science, The George Washington University, from 2002-2020. Currently she is a professor of computer science with Shandong University, Qingdao, China. Her research focuses on blockchain computing, security and privacy, and Internet of Things.