Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates

Dong Yin [1] **Yudong Chen** [2] **Kannan Ramchandran** [1] **Peter Bartlett** [1 3]

Abstract

In this paper, we develop distributed optimization algorithms that are provably robust against Byzantine failures—arbitrary and potentially adversarial behavior, in distributed computing systems, with a focus on achieving optimal statistical performance. A main result of this work is a sharp analysis of two robust distributed gradient descent algorithms based on median and trimmed mean operations, respectively. We prove statistical error rates for all of strongly convex, non-strongly convex, and smooth non-convex population loss functions. In particular, these algorithms are shown to achieve order-optimal statistical error rates for strongly convex losses. To achieve better communication efficiency, we further propose a median-based distributed algorithm that is provably robust, and uses only one communication round. For strongly convex quadratic loss, we show that this algorithm achieves the same optimal error rate as the robust distributed gradient descent algorithms.

1. Introduction

Many tasks in computer vision, natural language processing and recommendation systems require learning complex prediction rules from large datasets. As the scale of the datasets in these learning tasks continues to grow, it is crucial to utilize the power of distributed computing and storage. In such large-scale distributed systems, robustness and security issues have become a major concern. In particular, individual computing units—known as worker machines—may exhibit abnormal behavior due to crashes, faulty hardware, stalled computation or unreliable communication channels. Security issues are only exacerbated in the so-called *Federated Learning* setting, a modern distributed learning paradigm that is more decentralized, and that uses the data owners' devices (such as mobile phones and personal computers) as worker machines (McMahan & Ramage, 2017; Konečnỳ et al., 2016). Such machines are often more unpredictable, and in particular may be susceptible to malicious and coordinated attacks.

Due to the inherent unpredictability of this abnormal (sometimes adversarial) behavior, it is typically modeled as *Byzantine failure* (Lamport et al., 1982), meaning that some worker machines may behave completely arbitrarily and can send any message to the master machine that maintains and updates an estimate of the parameter vector to be learned. Byzantine failures can incur major degradation in learning performance. It is well-known that standard learning algorithms based on naive aggregation of the workers' messages can be arbitrarily skewed by a single Byzantine-faulty machine. Even when the messages from Byzantine machines take only moderate values—hence difficult to detect—and when the number of such machines is small, the performance loss can still be significant. We demonstrate such an example in our experiments in Section 7.

In this paper, we aim to develop distributed statistical learning algorithms that are provably robust against Byzantine failures. While this objective is considered in a few recent works (Feng et al., 2014; Blanchard et al., 2017; Chen et al., 2017), a fundamental problem remains poorly understood, namely the *optimal statistical performance* of a robust learning algorithm. A learning scheme in which the master machine always outputs zero regardless of the workers' messages, is certainly not affected by Byzantine failures, but it will not return anything statistically useful either. On the other hand, many standard distributed algorithms that achieve good statistical performance in the absence of Byzantine failures, become completely unreliable otherwise. Therefore, a main goal of this work is to understand the following questions: what is the best achievable statistical performance while being Byzantine-robust, and how to design an algorithm that achieves such performance?

To formalize this question, we consider a standard statistical setting of empirical risk minimization (ERM). Here nm data points are sampled independently from some distribution and distributed evenly among m machines, αm of which are Byzantine. The goal is to learn a parametric model by minimizing some loss function defined by the data. In this statistical setting, one expects that the error in learning

[1]Department of EECS, UC Berkeley [2]School of ORIE, Cornell University [3]Department of Statistics, UC Berkeley. Correspondence to: Dong Yin <dongyin@berkeley.edu>.

Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018. Copyright 2018 by the author(s).

the parameter, measured in an appropriate metric, should decrease when the amount of data nm becomes larger and the fraction of Byzantine machines α becomes smaller. In fact, we can show that, at least for strongly convex problems, no algorithm can achieve an error lower than

$$\widetilde{\Omega}\left(\frac{\alpha}{\sqrt{n}}+\frac{1}{\sqrt{nm}}\right)=\widetilde{\Omega}\left(\frac{1}{\sqrt{n}}\left(\alpha+\frac{1}{\sqrt{m}}\right)\right),$$

regardless of communication costs;[1] see Observation 1 in Section 6. Intuitively, the above error rate is the optimal rate that one should target for, as $\frac{1}{\sqrt{n}}$ is the effective standard deviation for each machine with n data points, α is the bias effect of Byzantine machines, and $\frac{1}{\sqrt{m}}$ is the averaging effect of m normal machines. When there are no or few Byzantine machines, we see the usual scaling $\frac{1}{\sqrt{mn}}$ with the total number of data points; when some machines are Byzantine, their influence remains bounded, and moreover is proportional to α. If an algorithm is guaranteed to attain this bound, we are assured that we do not sacrifice the quality of learning when trying to guard against Byzantine failures—we pay a price that is unavoidable, but otherwise achieve the best possible statistical accuracy in the presence of Byzantine failures.

Another important consideration for us is *communication efficiency*. As communication between machines is costly, one cannot simply send all data to the master machine. This constraint precludes direct application of standard robust learning algorithms (such as M-estimators (Huber, 2011)), which assume access to all data. Instead, a desirable algorithm should involve a small number of communication rounds as well as a small amount of data communicated per round. We consider a setting where in each round a worker or master machine can only communicate a vector of size $\mathcal{O}(d)$, where d is the dimension of the parameter to be learned. In this case, the total communication cost is proportional to the number of communication rounds.

To summarize, we aim to develop distributed learning algorithms that simultaneously achieve two objectives:

- **Statistical optimality:** attain an $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}}+\frac{1}{\sqrt{nm}})$ rate.
- **Communication efficiency:** $\mathcal{O}(d)$ communication per round, with as few rounds as possible.

To the best of our knowledge, *no existing algorithm achieves these two goals simultaneously*. In particular, previous robust algorithms either have unclear or sub-optimal statistical guarantees, or incur a high communication cost and hence not applicable in a distributed setting—we discuss related work in more detail in Section 2.

[1]Throughout the paper, unless otherwise stated, $\Omega(\cdot)$ and $\mathcal{O}(\cdot)$ hide universal multiplicative constants; $\widetilde{\Omega}(\cdot)$ and $\widetilde{\mathcal{O}}(\cdot)$ further hide terms that are independent of α, n, m or logarithmic in n, m.

Our Contributions We propose two robust distributed gradient descent (GD) algorithms, one based on coordinate-wise median, and the other on coordinate-wise trimmed mean. We establish their statistical error rates for strongly convex, non-strongly convex, and non-convex *population* loss functions. Particularly for strongly convex losses, we show that these algorithms achieve order-optimal statistical rates under mild conditions. We further propose a median-based robust algorithm that only requires one communication round, and show that it also achieves the optimal rate for strongly convex quadratic losses. The statistical error rates of these three algorithms are summarized as follows.

- **Median-based GD:** $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}}+\frac{1}{\sqrt{nm}}+\frac{1}{n})$, order-optimal for strongly convex loss if $n \gtrsim m$.
- **Trimmed-mean-based GD:** $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}}+\frac{1}{\sqrt{nm}})$, order-optimal for strongly convex loss.
- **Median-based one-round algorithm:** $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}}+\frac{1}{\sqrt{nm}}+\frac{1}{n})$, order-optimal for strongly convex quadratic loss if $n \gtrsim m$.

A major technical challenge in our statistical setting here is as follows: the nm data points are sampled once and fixed, and each worker machine has access to the same set of data throughout learning process. This creates complicated probabilistic dependency across the iterations of the algorithms. Worse yet, Byzantine machines may create further unspecified dependency. We overcome this difficulty by proving certain *uniform* bounds via careful covering arguments. Furthermore, for the analysis of median-based algorithms, we cannot simply adapt standard techniques (such as those in Minsker et al. (2015)), which can only show that the output of the master machine is as accurate as that of *one* normal machine, leading to a sub-optimal $\mathcal{O}(\frac{1}{\sqrt{n}})$ rate even without Byzantine failures. Instead, we make use of a more delicate argument based on normal approximation and Berry-Esseen-type inequalities.

2. Related Work

Outlier-robust estimation in non-distributed settings is a classical topic in statistics (Huber, 2011). Particularly relevant to us is the so-called *median-of-means* method, in which one partitions the data m subsets, computes an estimate from each sub-dataset, and finally takes the median of these m estimates. This idea is studied in Nemirovskii et al. (1983); Jerrum et al. (1986); Alon et al. (1999); Lerasle & Oliveira (2011); Minsker et al. (2015), and has been applied to bandit and least square regression problems (Bubeck et al., 2013; Lugosi & Mendelson, 2016; Kogler & Traxler, 2016) as well as problems involving heavy-tailed distributions (Hsu & Sabato, 2016; Lugosi & Mendelson, 2017). In a very recent work, Minsker & Strawn (2017) provide new analysis of median-of-means using normal approximation. We borrow some techniques from this paper, but need to address a significant harder problem: 1) we deal with the Byzantine setting with arbitrary/adversarial outliers, which is not considered in their paper; 2) we study iterative algorithms for general multi-dimensional problems with convex and non-convex losses, while they mainly focus on one-shot algorithms for mean-estimation-type problems.

The median-of-means method is used in the context of Byzantine-robust distributed learning in two recent papers. In particular, the work of Feng et al. (2014) considers a simple one-shot application of median-of-means, and only proves a sub-optimal $\widetilde{\mathcal{O}}(\frac{1}{\sqrt{n}})$ error rate as mentioned. The work of Chen et al. (2017) considers only strongly convex losses, and seeks to circumvent the above issue by grouping the worker machines into mini-batches; however, their rate $\widetilde{\mathcal{O}}(\frac{\sqrt{\alpha}}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$ still falls short of being optimal, and in particular their algorithm fails even when there is only one Byzantine machine in each mini-batch.

Other methods have been proposed for Byzantine-robust distributed learning and optimization; e.g., Su & Vaidya (2016a;b). These works consider optimizing fixed functions and do not provide guarantees on statistical error rates. Most relevant is the work by Blanchard et al. (2017), who propose to aggregate the gradients from worker machines using a robust procedure. Their optimization setting—which is at the level of stochastic gradient descent and assumes unlimited, independent access to a strong stochastic gradient oracle—is fundamentally different from ours; in particular, they do not provide a characterization of the statistical errors given a fixed number of data points.

Communication efficiency has been studied extensively in non-Byzantine distributed settings (McMahan et al., 2016; Yin et al., 2017). An important class of algorithms are based on one-round aggregation methods (Zhang et al., 2012; 2015; Rosenblatt & Nadler, 2016). More sophisticated algorithms have been proposed in order to achieve better accuracy than the one-round approach while maintaining lower communication costs; examples include DANE (Shamir et al., 2014), Disco (Zhang & Lin, 2015), distributed SVRG (Lee et al., 2015) and their variants (Reddi et al., 2016; Wang et al., 2017). Developing Byzantine-robust versions of these algorithms is an interesting future direction.

For outlier-robust estimation in non-distributed settings, much progress has been made recently in terms of improved performance in high-dimensional problems (Diakonikolas et al., 2016; Lai et al., 2016; Bhatia et al., 2015) as well as developing list-decodable and semi-verified learning schemes when a majority of the data points are adversarial (Charikar et al., 2017). These results are not directly applicable to our distributed setting with general loss functions, but it is nevertheless an interesting future problem to investigate their potential extension for our problem.

3. Problem Setup

In this section, we formally set up our problem and introduce a few concepts key to our the algorithm design and analysis. Suppose that training data points are sampled from some unknown distribution $\mathcal{D}$ on the sample space $\mathcal{Z}$. Let $f(\mathbf{w};\mathbf{z})$ be a loss function of a parameter vector $\mathbf{w} \in \mathcal{W} \subseteq \mathbb{R}^d$ associated with the data point $\mathbf{z}$, where $\mathcal{W}$ is the parameter space, and $F(\mathbf{w}) := \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}[f(\mathbf{w};\mathbf{z})]$ is the corresponding population loss function. Our goal is to learn a model defined by the parameter that minimizes the population loss:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w}). \tag{1}$$

The parameter space $\mathcal{W}$ is assumed to be convex and compact with diameter D, i.e., $\|\mathbf{w}-\mathbf{w}'\|_2 \le D, \forall \mathbf{w}, \mathbf{w}' \in \mathcal{W}$. We consider a distributed computation model with one master machine and m worker machines. Each worker machine stores n data points, each of which is sampled independently from $\mathcal{D}$. Denote by $\mathbf{z}^{i,j}$ the j-th data on the i-th worker machine, and $F_i(\mathbf{w}) := \frac{1}{n}\sum_{j=1}^{n} f(\mathbf{w};\mathbf{z}^{i,j})$ the empirical risk function for the i-th worker. We assume that an α fraction of the m worker machines are Byzantine, and the remaining $1-\alpha$ fraction are normal. With the notation $[N] := \{1,2,\ldots,N\}$, we index the set of worker machines by $[m]$, and denote the set of Byzantine machines by $\mathcal{B} \subset [m]$ (thus $|\mathcal{B}| = \alpha m$). The master machine communicates with the worker machines using some predefined protocol. The Byzantine machines need not obey this protocol and can send arbitrary messages to the master; in particular, they may have complete knowledge of the system and learning algorithms, and can collude with each other.

We introduce the coordinate-wise median and trimmed mean operations, which serve as building blocks for our algorithm.

Definition 1 (Coordinate-wise median)**.** *For vectors $\mathbf{x}^i \in \mathbb{R}^d$, $i \in [m]$, the coordinate-wise median $\mathbf{g} := \mathsf{med}\{\mathbf{x}^i : i \in [m]\}$ is a vector with its k-th coordinate being $g_k = \mathsf{med}\{x_k^i : i \in [m]\}$ for each $k \in [d]$, where med is the usual (one-dimensional) median.*

Definition 2 (Coordinate-wise trimmed mean)**.** *For $\beta \in [0, \frac{1}{2})$ and vectors $\mathbf{x}^i \in \mathbb{R}^d$, $i \in [m]$, the coordinate-wise β-trimmed mean $\mathbf{g} := \mathsf{trmean}_\beta\{\mathbf{x}^i : i \in [m]\}$ is a vector with its k-th coordinate being $g_k = \frac{1}{(1-2\beta)m}\sum_{x\in U_k} x$ for each $k \in [d]$. Here U_k is a subset of $\{x_k^1,\ldots,x_k^m\}$ obtained by removing the largest and smallest β fraction of its elements.*

For the analysis, we need several standard definitions concerning random variables/vectors.

Definition 3 (Variance of random vectors)**.** *For a random vector $\mathbf{x}$, define its variance as $\mathrm{Var}(\mathbf{x}) := \mathbb{E}[\|\mathbf{x}-\mathbb{E}[\mathbf{x}]\|_2^2]$.*

Definition 4 (Absolute skewness)**.** *For a one-dimensional random variable X, define its absolute skewness[2] as*

[2]Note the difference with the usual skewness $\frac{\mathbb{E}[(X-\mathbb{E}[X])^3]}{\mathrm{Var}(X)^{3/2}}$.

$\gamma(X) := \frac{\mathbb{E}[|X-\mathbb{E}[X]|^3]}{\mathrm{Var}(X)^{3/2}}$. *For a d-dimensional random vector $\mathbf{x}$, we define its absolute skewness as the vector of the absolute skewness of each coordinate of $\mathbf{x}$, i.e.,* $\gamma(\mathbf{x}) := [\gamma(x_1)\ \gamma(x_2)\ \cdots\ \gamma(x_d)]^\top$.

Definition 5 (Sub-exponential random variables)**.** *A random variable X with $\mathbb{E}[X]=\mu$ is called v-sub-exponential if* $\mathbb{E}[e^{\lambda(X-\mu)}] \le e^{\frac{1}{2}v^2\lambda^2}$, $\forall\, |\lambda| < \frac{1}{v}$.

Finally, we need several standard concepts from convex analysis regarding a differentiable function $h(\cdot) : \mathbb{R}^d \to \mathbb{R}$.

Definition 6 (Lipschitz)**.** *h is L-Lipschitz if*
$|h(\mathbf{w}) - h(\mathbf{w}')| \le L\|\mathbf{w}-\mathbf{w}'\|_2, \forall\, \mathbf{w}, \mathbf{w}'$.

Definition 7 (Smoothness)**.** *h is L'-smooth if*
$\|\nabla h(\mathbf{w}) - \nabla h(\mathbf{w}')\|_2 \le L'\|\mathbf{w}-\mathbf{w}'\|_2, \forall\, \mathbf{w}, \mathbf{w}'$.

Definition 8 (Strong convexity)**.** *h is λ-strongly convex if* $h(\mathbf{w}') \ge h(\mathbf{w}) + \langle \nabla h(\mathbf{w}), \mathbf{w}'-\mathbf{w}\rangle + \frac{\lambda}{2}\|\mathbf{w}'-\mathbf{w}\|_2^2, \forall\, \mathbf{w}, \mathbf{w}'$.

4. Robust Distributed Gradient Descent

We describe two robust distributed gradient descent algorithms, one based on coordinate-wise median and the other on trimmed mean. These two algorithms are formally given in Algorithm 1 as Option I and Option II, respectively, where the symbol $*$ represents an arbitrary vector.

In each parallel iteration of the algorithms, the master machine broadcasts the current model parameter to all worker machines. The normal worker machines compute the gradients of their local loss functions and then send the gradients back to the master machine. The Byzantine machines may send any messages of their choices. The master machine then performs a gradient descent update on the model parameter with step-size η, using either the coordinate-wise median or trimmed mean of the received gradients. The Euclidean projection $\Pi_{\mathcal{W}}(\cdot)$ ensures that the model parameter stays in the parameter space $\mathcal{W}$.

Below we provide statistical guarantees on the error rates of these algorithms, and compare their performance. Throughout we assume that each loss functions $f(\mathbf{w};\mathbf{z})$ and the population loss function $F(\mathbf{w})$ are smooth:

Assumption 1 (Smoothness of f and F)**.** *For any $\mathbf{z} \in \mathcal{Z}$, the partial derivative of $f(\cdot;\mathbf{z})$ with respect to the k-th coordinate of its first argument, denoted by $\partial_k f(\cdot;\mathbf{z})$, is L_k-Lipschitz for each $k \in [d]$, and the function $f(\cdot;\mathbf{z})$ is L-smooth. Let $\widehat{L} := (\sum_{k=1}^d L_k^2)^{1/2}$. Also assume that the population loss function $F(\cdot)$ is L_F-smooth.*

It is easy to see hat $L_F \le L \le \widehat{L}$. We note that $\widehat{L}$ appears because we have coordinate-wise operations and the $\widehat{L}$ quantity combines the smoothness parameter of all the d partial derivatives. When the dimension of $\mathbf{w}$ is high, the quantity $\widehat{L}$ may be large. However, we will soon see that $\widehat{L}$ only appears in the logarithmic factors in our bounds and thus does not have a significant impact.

Algorithm 1 Robust Distributed Gradient Descent

```
Require: Initialize parameter vector $\mathbf{w}^0 \in \mathcal{W}$, algorithm
    parameters $\beta$ (for Option II), $\eta$ and $T$.
  for $t = 0, 1, 2, \dots, T-1$ do
    Master machine: send $\mathbf{w}^t$ to all the worker machines.
    for $i \in [m]$ in parallel do
      Worker machine $i$: compute local gradient
        $\mathbf{g}^i(\mathbf{w}^t) \leftarrow \begin{cases} \nabla F_i(\mathbf{w}^t) & \text{normal machines,} \\ * & \text{Byzantine machines,} \end{cases}$
      send $\mathbf{g}^i(\mathbf{w}^t)$ to master machine.
    end for
    Master machine: compute aggregate gradient
      $\mathbf{g}(\mathbf{w}^t) \leftarrow \begin{cases} \mathsf{med}\{\mathbf{g}^i(\mathbf{w}^t) : i \in [m]\} & \text{Option I} \\ \mathsf{trmean}_\beta\{\mathbf{g}^i(\mathbf{w}^t) : i \in [m]\} & \text{Option II} \end{cases}$
    update model parameter $\mathbf{w}^{t+1} \leftarrow \Pi_{\mathcal{W}}(\mathbf{w}^t - \eta\mathbf{g}(\mathbf{w}^t))$.
  end for
```

In addition, when $F(\cdot)$ is convex, we assume that $\mathbf{w}^*$, the minimizer of $F(\cdot)$ in $\mathcal{W}$, is also the minimizer of $F(\cdot)$ in $\mathbb{R}^d$. Formally, we have

Assumption 2 (minimizer in $\mathcal{W}$)**.** *Suppose that $F(\mathbf{w})$ is convex, and let $\mathbf{w}^* = \arg\min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w})$. We assume that* $\nabla F(\mathbf{w}^*) = 0$.

4.1. Median-based Gradient Descent

We first consider our median-based algorithm, namely Algorithm 1 with Option I. We impose the assumptions that the gradient of the loss function f has bounded variance, and each coordinate of the gradient has coordinate-wise bounded absolute skewness:

Assumption 3 (Bounded variance of gradient)**.**
For any $\mathbf{w} \in \mathcal{W}$, $\mathrm{Var}(\nabla f(\mathbf{w};\mathbf{z})) \le V^2$.

Assumption 4 (Bounded skewness of gradient)**.**
For any $\mathbf{w} \in \mathcal{W}$, $\|\gamma(\nabla f(\mathbf{w};\mathbf{z}))\|_\infty \le S$.

These assumptions are satisfied in many learning problems with small values of V^2 and S. Below we provide a concrete example in terms of a linear regression problem.

Proposition 1. *Suppose that each data point $\mathbf{z} = (\mathbf{x}, y) \in \mathbb{R}^d \times \mathbb{R}$ is generated by $y = \mathbf{x}^\top\mathbf{w}^* + \xi$ with some $\mathbf{w}^* \in \mathcal{W}$. Assume that the elements of $\mathbf{x}$ are independent and uniformly distributed in $\{-1, 1\}$, and that the noise $\xi \sim \mathcal{N}(0,\sigma^2)$ is independent of $\mathbf{x}$. With the quadratic loss function $f(\mathbf{w};\mathbf{x},y) = \frac{1}{2}(y - \mathbf{x}^\mathrm{T}\mathbf{w})^2$, we have* $\mathrm{Var}(\nabla f(\mathbf{w};\mathbf{x},y)) = (d-1)\|\mathbf{w}-\mathbf{w}^*\|_2^2 + d\sigma^2$*, and* $\|\gamma(\nabla f(\mathbf{w};\mathbf{x},y))\|_\infty \le 480$.

We prove Proposition 1 in Appendix A.1. In this example, the upper bound V on $\mathrm{Var}(\nabla f(\mathbf{w};\mathbf{x},y))$ depends on dimension d and the diameter of the parameter space, and if the diameter is a constant, we have $V = \mathcal{O}(\sqrt{d})$. Moreover, the gradient skewness is bounded by a universal constant S regardless of the size of the parameter space.

We now state our main technical results on the median-based algorithm, namely statistical error guarantees for strongly convex, non-strongly convex, and smooth non-convex population loss functions F.

Strongly Convex Losses: We first consider the case where the population loss function $F(\cdot)$ is strongly convex. Note that we do not require strong convexity of the individual loss function $f(\cdot;\mathbf{z})$.

Theorem 1. *Consider Option I in Algorithm 1. Suppose that Assumptions 1, 2, 3, and 4 hold, $F(\cdot)$ is λ_F-strongly convex, and the fraction α of Byzantine machines satisfies*

$$\alpha + \sqrt{\frac{d\log(1+nm\widehat{L}D)}{m(1-\alpha)}} + 0.4748\frac{S}{\sqrt{n}} \leq \frac{1}{2} - \epsilon \tag{2}$$

for some $\epsilon > 0$. Choose step-size $\eta = 1/L_F$. Then, with probability at least $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$, after T parallel iterations, we have

$$\|\mathbf{w}^T - \mathbf{w}^*\|_2 \leq (1 - \frac{\lambda_F}{L_F+\lambda_F})^T\|\mathbf{w}^0 - \mathbf{w}^*\|_2 + \frac{2}{\lambda_F}\Delta,$$

where

$$\Delta := \mathcal{O}\Big(C_\epsilon V\big(\frac{\alpha}{\sqrt{n}} + \sqrt{\frac{d\log(nm\widehat{L}D)}{nm}} + \frac{S}{n}\big)\Big), \tag{3}$$

and C_ϵ is defined as

$$C_\epsilon := \sqrt{2\pi}\exp\Big(\frac{1}{2}(\Phi^{-1}(1-\epsilon))^2\Big), \tag{4}$$

with $\Phi^{-1}(\cdot)$ being the inverse of the cumulative distribution function of the standard Gaussian distribution $\Phi(\cdot)$.

We prove Theorem 1 in Appendix B. In (3), we hide universal constants and a higher order term that scales as $\frac{1}{nm}$, and the factor C_ϵ is a function of ϵ; as a concrete example, $C_\epsilon \approx 4$ when $\epsilon = \frac{1}{6}$. Theorem 1 together with the inequality $\log(1-x) \leq -x$, guarantees that after running $T \geq \frac{L_F+\lambda_F}{\lambda_F}\log(\frac{\lambda_F}{2\Delta}\|\mathbf{w}^0 - \mathbf{w}^*\|_2)$ parallel iterations, with high probability we can obtain a solution $\widehat{\mathbf{w}} = \mathbf{w}^T$ with error $\|\widehat{\mathbf{w}} - \mathbf{w}^*\|_2 \leq \frac{4}{\lambda_F}\Delta$.

Here we achieve an the error rate (defined as the distance between $\widehat{\mathbf{w}}$ and the optimal solution $\mathbf{w}^*$) of the form $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$. In Section 6, we provide a lower bound showing that the error rate of any algorithm is $\widetilde{\Omega}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$. Therefore the first two terms in the upper bound cannot be improved. The third term $\frac{1}{n}$ is due to the dependence of median on the skewness of the gradients. When each worker machine has a sufficient amount of data, more specifically $n \gtrsim m$, we achieve an order-optimal error rate up to logarithmic factors.

Non-strongly Convex Losses: We next consider the case where the population risk function $F(\cdot)$ is convex, but not necessarily strongly convex. In this case, we need a mild technical assumption on the size of the parameter space $\mathcal{W}$.

Assumption 5 (Size of $\mathcal{W}$)**.** *The parameter space $\mathcal{W}$ contains the following ℓ_2 ball centered at $\mathbf{w}^*$: $\{\mathbf{w} \in \mathbb{R}^d : \|\mathbf{w} - \mathbf{w}^*\|_2 \leq 2\|\mathbf{w}^0 - \mathbf{w}^*\|_2\}$.*

This assumption (and Assumption 6 below) ensures that the iterates $\mathbf{w}^t$ always stay in $\mathcal{W}$ without projection. Doing so streamlines our analysis, as our main focus is on robustness. We then have the following result on the convergence rate in terms of the value of the population risk function.

Theorem 2. *Consider Option I in Algorithm 1. Suppose that Assumptions 1, 2, 3, 4 and 5 hold, and that the population loss $F(\cdot)$ is convex, and α satisfies (2) for some $\epsilon > 0$. Define Δ as in (3), and choose step-size $\eta = 1/L_F$. Then, with probability at least $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$, after $T = \frac{L_F}{\Delta}\|\mathbf{w}^0 - \mathbf{w}^*\|_2$ parallel iterations, we have*

$$F(\mathbf{w}^T) - F(\mathbf{w}^*) \leq 16\|\mathbf{w}^0 - \mathbf{w}^*\|_2\Delta\Big(1 + \frac{1}{2L_F}\Delta\Big).$$

We prove Theorem 2 in Appendix C. We observe that the error rate, defined as the excess risk $F(\mathbf{w}^T) - F(\mathbf{w}^*)$, again has the form $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$.

Non-convex Losses: When $F(\cdot)$ is non-convex but smooth, we need a somewhat different technical assumption on the size of $\mathcal{W}$.

Assumption 6 (Size of $\mathcal{W}$)**.** *Suppose that $\forall\ \mathbf{w} \in \mathcal{W}$, $\|\nabla F(\mathbf{w})\|_2 \leq M$. We assume that $\mathcal{W}$ contains the ℓ_2 ball $\{\mathbf{w} \in \mathbb{R}^d : \|\mathbf{w} - \mathbf{w}^0\|_2 \leq \frac{2}{\Delta^2}(M+\Delta)(F(\mathbf{w}^0) - F(\mathbf{w}^*))\}$, where Δ is defined as in (3).*

We have the following guarantees on the rate of convergence to a critical point of the population loss $F(\cdot)$.

Theorem 3. *Consider Option I in Algorithm 1. Suppose that Assumptions 1 3, 4 and 6 hold, and α satisfies (2) for some $\epsilon > 0$. Define Δ as in (3), and choose step-size $\eta = 1/L_F$. With probability at least $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$, after $T = \frac{2L_F}{\Delta^2}(F(\mathbf{w}^0) - F(\mathbf{w}^*))$ parallel iterations, we have*

$$\min_{t=0,1,\ldots,T}\|\nabla F(\mathbf{w}^t)\|_2 \leq \sqrt{2}\Delta.$$

We prove Theorem 3 in Appendix D. We again obtain an $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$ error rate in terms of the gap to a critical point of $F(\mathbf{w})$.

4.2. Trimmed-mean-based Gradient Descent

We next analyze the robust distributed gradient descent algorithm based on coordinate-wise trimmed mean, namely Option II in Algorithm 1. Here we need stronger assumptions on the tail behavior of the partial derivatives of the loss functions—in particular, sub-exponentiality.

Assumption 7 (Sub-exponential gradients)**.** *We assume that for all $k \in [d]$ and $\mathbf{w} \in \mathcal{W}$, the partial derivative of $f(\mathbf{w};\mathbf{z})$ with respect to the k-th coordinate of $\mathbf{w}$, $\partial_k f(\mathbf{w};\mathbf{z})$, is v-sub-exponential.*

The sub-exponential property implies that all the moments of the derivatives are bounded. This is a stronger assumption than the bounded absolute skewness (hence bounded third moments) required by the median-based GD algorithm.

We use the same example as in Proposition 1 and show that the derivatives of the loss are indeed sub-exponential.

Proposition 2. *Consider the regression problem in Proposition 1. For all $k \in [d]$ and $\mathbf{w} \in \mathcal{W}$, the partial derivative $\partial_k f(\mathbf{w};\mathbf{z})$ is $\sqrt{\sigma^2 + \|\mathbf{w} - \mathbf{w}^*\|_2^2}$-sub-exponential.*

Proposition 2 is proved in Appendix A.3. We now proceed to establish the statistical guarantees of the trimmed-mean-based algorithm, for different loss function classes. When the population loss $F(\cdot)$ is convex, we again assume that the minimizer of $F(\cdot)$ in $\mathcal{W}$ is also its minimizer in $\mathbb{R}^d$. The next three theorems are analogues of Theorems 1–3 for the median-based GD algorithm.

Strongly Convex Losses: We have the following result.

Theorem 4. *Consider Option II in Algorithm 1. Suppose that Assumptions 1, 2, and 7 hold, $F(\cdot)$ is λ_F-strongly convex, and $\alpha \le \beta \le \frac{1}{2} - \epsilon$ for some $\epsilon > 0$. Choose step-size $\eta = 1/L_F$. Then, with probability at least $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$, after T parallel iterations, we have*

$$\|\mathbf{w}^T - \mathbf{w}^*\|_2 \le \Big(1 - \frac{\lambda_F}{L_F + \lambda_F}\Big)^T \|\mathbf{w}^0 - \mathbf{w}^*\|_2 + \frac{2}{\lambda_F}\Delta',$$

where

$$\Delta' := \mathcal{O}\Big(\frac{vd}{\epsilon}\big(\frac{\beta}{\sqrt{n}} + \frac{1}{\sqrt{nm}}\big)\sqrt{\log(nm\widehat{L}D)}\Big). \quad (5)$$

We prove Theorem 4 in Appendix E. In (5), we hide universal constants and higher order terms that scale as $\frac{\beta}{n}$ or $\frac{1}{nm}$. By running $T \ge \frac{L_F + \lambda_F}{\lambda_F}\log(\frac{\lambda_F}{2\Delta'}\|\mathbf{w}^0 - \mathbf{w}^*\|_2)$ parallel iterations, we can obtain a solution $\widehat{\mathbf{w}} = \mathbf{w}^T$ satisfying $\|\widehat{\mathbf{w}} - \mathbf{w}^*\|_2 \le \widetilde{\mathcal{O}}(\frac{\beta}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$. Note that one needs to choose the parameter for trimmed mean to satisfy $\beta \ge \alpha$. If we set $\beta = c\alpha$ for some universal constant $c \ge 1$, we can achieve an order-optimal error rate $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$.

Non-strongly Convex Losses: Again imposing Assumption 5 on the size of $\mathcal{W}$, we have the following guarantee.

Theorem 5. *Consider Option II in Algorithm 1. Suppose that Assumptions 1, 2, 5 and 7 hold, $F(\cdot)$ is convex, and $\alpha \le \beta \le \frac{1}{2} - \epsilon$ for some $\epsilon > 0$. Choose step-size $\eta = 1/L_F$, and define Δ' as in (5). Then, with probability at least $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$, after $T = \frac{L_F}{\Delta'}\|\mathbf{w}^0 - \mathbf{w}^*\|_2$ parallel iterations, we have*

$$F(\mathbf{w}^T) - F(\mathbf{w}^*) \le 16\|\mathbf{w}^0 - \mathbf{w}^*\|_2\Delta'\Big(1 + \frac{1}{2L_F}\Delta'\Big).$$

The proof of Theorem 5 is similar to that of Theorem 2, and we refer readers to Remark 1 in Appendix E. Again, by choosing $\beta = c\alpha$ $(c \ge 1)$, we obtain the $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$ error rate in the function value of $F(\mathbf{w})$.

Non-convex Losses: In this case, imposing a version of Assumption 6 on the size of $\mathcal{W}$, we have the following.

Theorem 6. *Consider Option II in Algorithm 1, and define Δ' as in (5). Suppose that Assumptions 1 and 7 hold, Assumption 6 holds with Δ replaced by Δ', and $\alpha \le \beta \le \frac{1}{2} - \epsilon$ for some $\epsilon > 0$. Choose step-size $\eta = 1/L_F$. Then, with probability at least $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$, after $T = \frac{2L_F}{\Delta'^2}(F(\mathbf{w}^0) - F(\mathbf{w}^*))$ parallel iterations, we have*

$$\min_{t=0,1,\ldots,T}\|\nabla F(\mathbf{w}^t)\|_2 \le \sqrt{2}\Delta'.$$

The proof of Theorem 6 is similar to that of Theorem 3; see Remark 1 in Appendix E. By choosing $\beta = c\alpha$ with $c \ge 1$, we again achieve the statistical rate $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$.

4.3. Comparisons

We compare the performance guarantees of the above two robust distribute GD algorithms. The trimmed-mean-based algorithm achieves the statistical error rate $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$, which is order-optimal for strongly convex loss. In comparison, the rate of the median-based algorithm is $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$, which has an additional $\frac{1}{n}$ term and is only optimal when $n \gtrsim m$. In particular, the trimmed-mean-based algorithm has better rates when each worker machine has small local sample size—the rates are meaningful even in the extreme case $n = \mathcal{O}(1)$. On the other hand, the median-based algorithm requires milder tail/moment assumptions on the loss derivatives (bounded skewness) than its trimmed-mean counterpart (sub-exponentiality). Finally, the trimmed-mean operation requires an additional parameter β, which can be any upper bound on the fraction α of Byzantine machines in order to guarantee robustness. Using an overly large β may lead to a looser bound and sub-optimal performance. In contrast, median-based GD does not require knowledge of α. We summarize these observations in Table 1. We see that the two algorithms are complementary to each other, and our experiment results corroborate this point.

	median GD	trimmed mean GD
Rate	$\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$	$\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$
$\partial_k f(\mathbf{w};\mathbf{z})$	Bounded skewness	Sub-exponential
α known?	No	Yes

Table 1. Comparison between the two robust distributed gradient descent algorithms.

5. Robust One-round Algorithm

As mentioned, in our distributed computing framework, the communication cost is proportional to the number of parallel iterations. The above two GD algorithms both require a number iterations depending on the desired accuracy. Can we further reduce the communication cost while keeping the algorithm Byzantine-robust and statistically optimal?

A natural candidate is the so-called one-round algorithm. Previous work has considered a standard one-round scheme where each local machine computes the empirical risk minimizer (ERM) using its local data and the master machine receives all workers' ERMs and computes their *average* (Zhang et al., 2012). Clearly, a single Byzantine machine can arbitrary skew the output of this algorithm. We instead consider a Byzantine-robust one-round algorithm. As detailed in Algorithm 2, we employ the coordinate-wise median operation to aggregate all the ERMs.

Algorithm 2 Robust One-round Algorithm

```
for $i \in [m]$ in parallel do
    Worker machine $i$: compute & send to master machine:
        $\widehat{\mathbf{w}}^i \leftarrow \begin{cases} \arg\min_{\mathbf{w}\in\mathcal{W}} F_i(\mathbf{w}) & \text{normal machines} \\ * & \text{Byzantine machines} \end{cases}$
end for
Master machine: compute $\widehat{\mathbf{w}} \leftarrow \mathsf{med}\{\widehat{\mathbf{w}}^i : i \in [m]\}$.
```

Our main result is a characterization of the error rate of Algorithm 2 in the presence of Byzantine failures. We are only able to establish such a guarantee when the loss functions are quadratic and $\mathcal{W} = \mathbb{R}^d$. However, one can implement this algorithm in problems with other loss functions.

Definition 9 (Quadratic loss function). *The loss function $f(\mathbf{w};\mathbf{z})$ is quadratic if it can be written as*

$$f(\mathbf{w};\mathbf{z}) = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{H}\mathbf{w} + \mathbf{p}^{\mathrm{T}}\mathbf{w} + c,$$

where $\mathbf{z} = (\mathbf{H}, \mathbf{p}, c)$, $\mathbf{H}$, and $\mathbf{p}$, and c are drawn from the distributions $\mathcal{D}_H$, $\mathcal{D}_p$, and $\mathcal{D}_c$, respectively.

Denote by $\mathbf{H}_F$, $\mathbf{p}_F$, and c_F the expectations of $\mathbf{H}$, $\mathbf{p}$, and c, respectively. Thus the population risk function takes the form $F(\mathbf{w}) = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{H}_F\mathbf{w} + \mathbf{p}_F^{\mathrm{T}}\mathbf{w} + c_F$.

We need a technical assumption which guarantees that each normal worker machine has unique ERM.

Assumption 8 (Strong convexity of F_i). *With probability 1, the empirical risk minimization function $F_i(\cdot)$ on each normal machine is strongly convex.*

Note that this assumption is imposed on $F_i(\mathbf{w})$, rather than on the individual loss $f(\mathbf{w};\mathbf{z})$ associated with a single data point. This assumption is satisfied, for example, when all $f(\cdot;\mathbf{z})$'s are strongly convex, or in the linear regression problems with the features $\mathbf{x}$ drawn from some continuous distribution (e.g. isotropic Gaussian) and $n \geq d$. We have the following guarantee for the robust one-round algorithm.

Theorem 7. *Suppose that $\forall\, \mathbf{z} \in \mathcal{Z}$, the loss function $f(\cdot;\mathbf{z})$ is convex and quadratic, $F(\cdot)$ is λ_F-strongly convex, and Assumption 8 holds. Assume that α satisfies*

$$\alpha + \sqrt{\frac{\log(nmd)}{2m(1-\alpha)}} + \frac{\widetilde{C}}{\sqrt{n}} \leq \frac{1}{2} - \epsilon$$

for some $\epsilon > 0$, where $\widetilde{C}$ is a quantity that depends on $\mathcal{D}_H$, $\mathcal{D}_p$, λ_F and is monotonically decreasing in n. Then, with probability at least $1 - \frac{4}{nm}$, the output $\widehat{\mathbf{w}}$ of the robust one-round algorithm satisfies

$$\|\widehat{\mathbf{w}} - \mathbf{w}^*\|_2 \leq \frac{C_\epsilon}{\sqrt{n}}\widetilde{\sigma}\big(\alpha + \sqrt{\frac{\log(nmd)}{2m(1-\alpha)}} + \frac{\widetilde{C}}{\sqrt{n}}\big),$$

where C_ϵ is defined as in (4) and

$$\widetilde{\sigma}^2 := \mathbb{E}\big[\|\mathbf{H}_F^{-1}\big((\mathbf{H} - \mathbf{H}_F)\mathbf{H}_F^{-1}\mathbf{p}_F - (\mathbf{p} - \mathbf{p}_F)\big)\|_2^2\big],$$

with $\mathbf{H}$ and $\mathbf{p}$ drawn from $\mathcal{D}_H$ and $\mathcal{D}_p$, respectively.

We prove Theorem 7 and provide an explicit expression of $\widetilde{C}$ in Appendix F. In terms of the dependence on α, n, and m, the robust one-round algorithm achieves the same error rate as the robust gradient descent algorithm based on coordinate-wise median, i.e., $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$, for quadratic problems. Again, this rate is optimal when $n \gtrsim m$. Therefore, at least for quadratic loss functions, the robust one-round algorithm has similar theoretical performance as the robust gradient descent algorithm with significantly less communication cost. Our experiments show that the one-round algorithm has good empirical performance for other losses as well.

6. Lower Bound

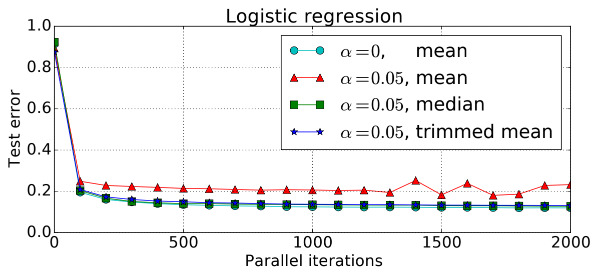
In this section, we provide a lower bound on the error rate for strongly convex losses, which implies that the $\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}}$ term is unimprovable. This lower bound is derived using a mean estimation problem, and is an extension of the lower bounds in the robust mean estimation literature such as Chen et al. (2015); Lai et al. (2016).

We consider the problem of estimating the mean $\boldsymbol{\mu}$ of some random variable $\mathbf{z} \sim \mathcal{D}$, which is equivalent to solving the following minimization problem:

$$\boldsymbol{\mu} = \arg\min_{\mathbf{w}\in\mathcal{W}} \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}[\|\mathbf{w} - \mathbf{z}\|_2^2], \tag{6}$$

Note that this is a special case of the general learning problem (1). We consider the same distributed setting as in Section 4, with a minor technical difference regarding the Byzantine machines. We assume that each of the m worker machines is Byzantine with probability α, independently of each other. The parameter α is therefore the *expected* fraction of Byzantine machines. In this setting we have the lower bound in Observation 1. In Appendix G, we also discuss how we can translate this average-case bound to a lower bound holds under the setting of our main theorems, that is, an unknown set of αm Byzantine machines are selected without any assumption.

Observation 1. *Consider the distributed mean estimation problem in (6) with Byzantine failure probability α, and suppose that $\mathcal{Z}$ is Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\sigma^2\mathbf{I}$ ($\sigma = \mathcal{O}(1)$). Then, any algorithm*



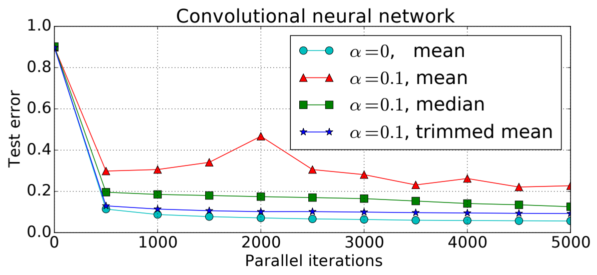


Figure 1. Test error vs the number of parallel iterations. For logistic regression, we set $m = 40$, and for trimmed mean, we choose $\beta = 0.05$; for CNN, we set $m = 10$, and for trimmed mean, we choose $\beta = 0.1$.

that computes an estimation $\widehat{\boldsymbol{\mu}}$ of the mean from the data has a constant probability of error $\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2 = \Omega(\frac{\alpha}{\sqrt{n}} + \sqrt{\frac{d}{nm}})$.

We prove Observation 1 in Appendix G. According to this observation, we see that the $\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}}$ dependence cannot be avoided, which in turn implies the order-optimality of the results in Theorem 1 (when $n \gtrsim m$) and Theorem 4.

7. Experiments

We conduct experiments to show the effectiveness of the median and trimmed mean operations. Our experiments are implemented with Tensorflow (Abadi et al., 2016) on Microsoft Azure system. We use the MNIST (LeCun et al., 1998) dataset and randomly partition the 60,000 training data into m subsamples with equal sizes. We use these subsamples to represent the data on m machines.

In the first experiment, we compare the performance of distributed gradient descent algorithms in the following 4 settings: 1) $\alpha = 0$ (no Byzantine machines), using vanilla distributed gradient descent (aggregating the gradients by taking the mean), 2) $\alpha > 0$, using vanilla distributed gradient descent, 3) $\alpha > 0$, using median-based algorithm, and 4) $\alpha > 0$, using trimmed-mean-based algorithm. We generate the Byzantine machines in the following way: we replace every training label y on these machines with $9 - y$, e.g., 0 is replaced with 9, 1 is replaced with 8, etc, and the Byzantine machines simply compute gradients based on these data. We also note that when generating the Byzantine machines, we do not simply add extreme values in the features or gradients; instead, the Byzantine machines send messages to the master machine with moderate values.

We train a multi-class logistic regression model and a convolutional neural network (CNN) using distributed gradient descent, and for each model, we compare the test accuracies in the aforementioned 4 settings. For the convolutional neural network model, we use the stochastic version of the distributed gradient descent algorithm; more specifically, in every iteration, each worker machine computes the gradient using 10% of its local data. We plot the test error as a function of the number of parallel iterations (i.e., communication rounds) in Figure 1. The final test accuracies are presented in Tables 2 and 3.

α	0	0.05		
Algorithm	mean	mean	median	trimmed mean
Acc. (%)	88.0	76.8	87.2	86.9

Table 2. Test accuracy on the logistic regression model using gradient descent. We set $m = 40$, and for trimmed mean, we choose $\beta = 0.05$.

α	0	0.1		
Algorithm	mean	mean	median	trimmed mean
Acc. (%)	94.3	77.3	87.4	90.7

Table 3. Test accuracy on CNN using gradient descent. We set $m = 10$, and for trimmed mean, we choose $\beta = 0.1$.

As we can see, in the adversarial settings, the vanilla distributed gradient descent algorithm suffers from severe performance loss, and using the median and trimmed mean operations, we observe significant improvement in test accuracy. This shows these two operations indeed have strong ability in defense against Byzantine failures.

In the second experiment, we compare the performance of distributed one-round algorithms in the following 3 settings: 1) $\alpha = 0$, mean aggregation, 2) $\alpha > 0$, mean aggregation, and 3) $\alpha > 0$, median aggregation. To show that our algorithm is able to defend against different types of adversarial behavior, we generate the Byzantine machines differently from the first experiment—the training labels are i.i.d. uniformly sampled from $\{0, \ldots, 9\}$, and these machines train models using the faulty data. We choose the multi-class logistic regression model, and the test accuracies are presented in Table 4.

α	0	0.1	
Algorithm	mean	mean	median
Acc. (%)	91.8	83.7	89.0

Table 4. Test accuracy on the logistic regression model using one-round algorithm. We set $m = 10$.

As we can see, for the one-round algorithm, although the theoretical guarantee is only proved for quadratic loss, in practice, the median-based one-round algorithm still improves the test accuracy in problems with other loss functions, such as the logistic loss here.

Acknowledgements

We gratefully acknowledge the support of the NSF through grant IIS-1619362, CIF award 1703678, CRII award 1657420 and grant 1704828. We also acknowledge the support of Berkeley DeepDrive Industry Consortium and Gift award from Huawei. Cloud computing resources are provided by a Microsoft Azure for Research award.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pp. 265–283, 2016.

Alon, N., Matias, Y., and Szegedy, M. The space complexity of approximating the frequency moments. *Journal of Computer and system sciences*, 58(1):137–147, 1999.

Bhatia, K., Jain, P., and Kar, P. Robust regression via hard thresholding. In *Advances in Neural Information Processing Systems*, pp. 721–729, 2015.

Blanchard, P., Mhamdi, E. M. E., Guerraoui, R., and Stainer, J. Byzantine-tolerant machine learning. *arXiv preprint arXiv:1703.02757*, 2017.

Bubeck, S., Cesa-Bianchi, N., and Lugosi, G. Bandits with heavy tail. *IEEE Transactions on Information Theory*, 59 (11):7711–7717, 2013.

Charikar, M., Steinhardt, J., and Valiant, G. Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 47–60. ACM, 2017.

Chen, M., Gao, C., and Ren, Z. Robust covariance matrix estimation via matrix depth. *arXiv preprint arXiv:1506.00691*, 2015.

Chen, Y., Su, L., and Xu, J. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *arXiv preprint arXiv:1705.05491*, 2017.

Diakonikolas, I., Kamath, G., Kane, D. M., Li, J., Moitra, A., and Stewart, A. Robust estimators in high dimensions without the computational intractability. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pp. 655–664. IEEE, 2016.

Feng, J., Xu, H., and Mannor, S. Distributed robust learning. *arXiv preprint arXiv:1409.5937*, 2014.

Hsu, D. and Sabato, S. Loss minimization and parameter estimation with heavy tails. *The Journal of Machine Learning Research*, 17(1):543–582, 2016.

Huber, P. J. Robust statistics. In *International Encyclopedia of Statistical Science*, pp. 1248–1251. Springer, 2011.

Jerrum, M. R., Valiant, L. G., and Vazirani, V. V. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43:169–188, 1986.

Kogler, A. and Traxler, P. Efficient and robust median-of-means algorithms for location and regression. In *Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2016 18th International Symposium on*, pp. 206–213. IEEE, 2016.

Konečný, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

Lai, K. A., Rao, A. B., and Vempala, S. Agnostic estimation of mean and covariance. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pp. 665–674. IEEE, 2016.

Lamport, L., Shostak, R., and Pease, M. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3):382–401, 1982.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lee, J. D., Lin, Q., Ma, T., and Yang, T. Distributed stochastic variance reduced gradient methods and a lower bound for communication complexity. *arXiv preprint arXiv:1507.07595*, 2015.

Lerasle, M. and Oliveira, R. I. Robust empirical mean estimators. *arXiv preprint arXiv:1112.3914*, 2011.

Lugosi, G. and Mendelson, S. Risk minimization by median-of-means tournaments. *arXiv preprint arXiv:1608.00757*, 2016.

Lugosi, G. and Mendelson, S. Sub-gaussian estimators of the mean of a random vector. *arXiv preprint arXiv:1702.00482*, 2017.

McMahan, B. and Ramage, D. Federated learning: Collaborative machine learning without centralized training data. `https://research.googleblog.com/2017/04/federated-learning-collaborative.html`, 2017.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., et al. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*, 2016.

Minsker, S. and Strawn, N. Distributed statistical estimation and rates of convergence in normal approximation. *arXiv preprint arXiv:1704.02658*, 2017.

Minsker, S. et al. Geometric median and robust estimation in banach spaces. *Bernoulli*, 21(4):2308–2335, 2015.

Nemirovskii, A., Yudin, D. B., and Dawson, E. R. Problem complexity and method efficiency in optimization. 1983.

Reddi, S. J., Konečnỳ, J., Richtárik, P., Póczós, B., and Smola, A. Aide: Fast and communication efficient distributed optimization. *arXiv preprint arXiv:1608.06879*, 2016.

Rosenblatt, J. D. and Nadler, B. On the optimality of averaging in distributed statistical learning. *Information and Inference: A Journal of the IMA*, 5(4):379–404, 2016.

Shamir, O., Srebro, N., and Zhang, T. Communication-efficient distributed optimization using an approximate newton-type method. In *International conference on machine learning*, pp. 1000–1008, 2014.

Su, L. and Vaidya, N. H. Fault-tolerant multi-agent optimization: optimal iterative distributed algorithms. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, pp. 425–434. ACM, 2016a.

Su, L. and Vaidya, N. H. Non-bayesian learning in the presence of byzantine agents. In *International Symposium on Distributed Computing*, pp. 414–427. Springer, 2016b.

Wang, S., Roosta-Khorasani, F., Xu, P., and Mahoney, M. W. Giant: Globally improved approximate newton method for distributed optimization. *arXiv preprint arXiv:1709.03528*, 2017.

Yin, D., Pananjady, A., Lam, M., Papailiopoulos, D., Ramchandran, K., and Bartlett, P. Gradient diversity: a key ingredient for scalable distributed learning. *arXiv preprint arXiv:1706.05699*, 2017.

Zhang, Y. and Lin, X. Disco: Distributed optimization for self-concordant empirical loss. In *International conference on machine learning*, pp. 362–370, 2015.

Zhang, Y., Wainwright, M. J., and Duchi, J. C. Communication-efficient algorithms for statistical optimization. In *Advances in Neural Information Processing Systems*, pp. 1502–1510, 2012.

Zhang, Y., Duchi, J., and Wainwright, M. Divide and conquer kernel ridge regression: A distributed algorithm with minimax optimal rates. *The Journal of Machine Learning Research*, 16(1):3299–3340, 2015.