Byzantine-Robust Decentralized Learning via Remove-then-Clip Aggregation

Caiyi Yang, Javad Ghaderi

Department of Electrical Engineering, Columbia University, USA
{cy2621, jghaderi}@columbia.edu

Abstract

We consider decentralized learning over a network of workers with heterogeneous datasets, in the presence of Byzantine workers. Byzantine workers may transmit arbitrary or malicious values to neighboring workers, leading to degradation in overall performance. The heterogeneous nature of the training data across various workers complicates the identification and mitigation of Byzantine workers. To address this complex problem, we introduce a resilient decentralized learning approach that combines the gradient descent algorithm with a novel robust aggregator. Specifically, we propose a remove-then-clip aggregator, whereby each benign worker meticulously filters the neighbors' values and subsequently projects the remaining values to a sphere centered at its local value, with an appropriately selected radius. We prove that our proposed method converges to a neighborhood of a stationary point for non-convex objectives under standard assumptions. Furthermore, empirical evaluations are provided to demonstrate the superior performance of our method in comparison to existing algorithms, under various Byzantine attack models.

Introduction

Modern machine learning (ML) applications rely on processing massive amounts of data, which are often generated and held by devices and users worldwide. Training machine learning models on distributed data provides many advantages over traditional centralized approaches in core aspects such as data ownership, privacy and fault-tolerance (Kairouz et al. 2021). In this direction, federated learning (FL) has gained significant traction (Konečnỳ et al. 2016; McMahan et al. 2017), where edge devices collaborate to train a model without sharing their local data. In centralized FL, devices send their model updates to a centralized server (called parameter server) which aggregates the updates and coordinates the devices during the learning process. The parameter server poses a single point of failure and may cause a communication bottleneck as the number of devices increases. This problem has motivated the design of decentralized machine learning methods (Lian et al. 2017; Lalitha et al. 2018; Roy et al. 2019), that can achieve comparable model accuracy as the state-of-the-art FL approaches while boasting several significant advantages over FL, such as fault-resilience and self-scalability.

Despite the benefits of decentralized training approaches, their performance can be significantly degraded due to vulnerability to malicious devices or data heterogeneity across the devices. In particular, some devices can be faulty, referred to as *Byzantine workers* (Hegedűs, Danner, and Jelasity 2021), due to software/hardware errors or getting hacked, and send arbitrary or malicious model updates to other devices, thus severely degrading the overall performance. To address Byzantine attacks in the training process, a few Byzantine-robust decentralized learning algorithms have been introduced recently (Yang and Bajwa 2019; Guo et al. 2022; Fang, Yang, and Bajwa 2022; He, Karimireddy, and Jaggi 2022), where benign workers attempt to combine the updates received from their neighbors by using robust aggregation rules to mitigate the impact of potential Byzantine workers. Most current algorithms deal with Byzantine attacks under independent and identically distributed (IID) data across the devices; however, in reality, the data can vary dramatically across the devices in terms of quantity, label distribution, and feature distribution (Zhao et al. 2018; Hsieh et al. 2020). Data heterogeneity makes the detection of Byzantine workers more challenging compared to the IID data setting, as the divergent local models might be attributed to either Byzantine workers or the heterogeneous nature of the data. Most proposed Byzantine-robust decentralized learning algorithms cannot handle such non-IID data settings and result in slow convergence or degraded accuracy of the global model (Zhang and Yang 2021).

In this paper, we study the problem of decentralized learning in non-IID and Byzantine environments. We propose a robust decentralized learning algorithm to address the deficiencies of the prior algorithms. Specifically, the main contributions of our work can be summarized as follows:

- We present a decentralized learning method by utilizing the distributed stochastic gradient descent (DSGD) algorithm combined with a novel robust aggregator. The proposed robust aggregator, based on a *Remove-then-Clip (RTC)* idea, enables the application of DSGD algorithm to Byzantine non-IID settings.
- Given a general weighted network topology, we prove the convergence to a neighborhood of a stationary point for non-convex objectives in the Byzantine and non-IID

setting.

- We provide extensive evaluations that demonstrate the superior performance of our method compared to the prior approaches, under various Byzantine attacks. In particular, our method can lead to 3%-35% improvement over the prior approaches in the specific scenarios analyzed.

Related Work

There has been a tremendous amount of work on Byzantine-robust learning in the parameter-server model (Blanchard et al. 2017; Yin et al. 2018; Chen, Su, and Xu 2017; Xie, Koyejo, and Gupta 2019, 2020b; Regatti, Chen, and Gupta 2020). Most of the proposed robust aggregation rules are distance-based rules, including Krum (Blanchard et al. 2017), trimmed mean (Yin et al. 2018) and geometric median (Chen, Su, and Xu 2017), which filter the gradients/-parameters far from the average. It has been shown that these methods are vulnerable to some sophisticated attacks (Baruch, Baruch, and Goldberg 2019). On the other hand, there have been performance-based filtering strategies (Xie, Koyejo, and Gupta 2019, 2020b; Regatti, Chen, and Gupta 2020), which evaluate the model received from each worker on data sampled by the server and filter the abnormal one. However, such methods depend on auxiliary datasets at the server and cannot defend against time-coupled attacks (Xie, Koyejo, and Gupta 2020a). To address this issue, a clipping-based method has been proposed (Karimireddy, He, and Jaggi 2021), where the server projects all received values onto a sphere centered at its last updated value with a proper radius. This process, referred to as clipping, acts as a defense mechanism against sophisticated attacks including time-coupled attacks.

In the context of decentralized machine learning, there is a rich literature on compression techniques for reducing the size of messages exchanged between the workers (Koloskova et al. 2020), optimizing the communication topology (Assran et al. 2019; Ying et al. 2021), and addressing data heterogeneity (Shi et al. 2015; Di Lorenzo and Scutari 2016; Mateos, Bazerque, and Giannakis 2010; Alghunaim and Sayed 2020). In the Byzantine setting, there has been work on decentralized machine learning that uses trimmed mean aggregators at benign workers to aggregate models received from their neighbors (Su and Vaidya 2016; Yang and Bajwa 2019). Guo et al. (2022) have demonstrated that such an aggregation method is defeated by the so-called hidden attack (Guerraoui, Rouault et al. 2018). To solve this limitation, Guo et al. (2022) propose an algorithm called uniform Byzantine-resilient aggregation rule (UBAR) that combines distance-based and performance-based aggregators. However, this method will fail in the non-IID data setting (He, Karimireddy, and Jaggi 2022). There are only a few works on Byzantine-robust decentralized learning in the non-IID data setting (He, Karimireddy, and Jaggi 2022; Wu, Chen, and Ling 2023; Li et al. 2017). He, Karimireddy, and Jaggi (2022) propose a self-centered clipping (SCCLIP) aggregation rule, yet obtaining the clipping threshold in SCCLIP relies on actually knowing the set of benign workers in each neighborhood that is not feasible in a Byzantine environment. Wu, Chen, and Ling (2023) propose an iterative filtering-based aggregation rule, however, this rule encounters computational difficulties especially when the number of Byzantine workers is proportional to the total number of workers. Li et al. (2017) show that alternating direction method of multipliers (ADMM) converges linearly to a neighborhood of the optimal solution, under certain conditions, and proposes a robust variant of ADMM; however, no proof of convergence is provided for the robust variant. To fill the research gap, we propose a Byzantine-robust decentralized learning method, based on the combination of DSGD algorithm and a novel robust aggregator RTC. We prove that the proposed method converges to a neighborhood of a stationary point in non-IID and Byzantine settings.

Problem Formulation

Network Model

The network is modeled as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a set of n workers $\mathcal{V} = \{1, \cdots, n\}$ and a set of undirected edges $\mathcal{E}$. If $(i, j) \in \mathcal{E}$, workers i and j are neighbors and can communicate with each other. Let $\boldsymbol{W} = [w_{ij}]_{i,j\in\mathcal{V}} \in \mathbb{R}^{n\times n}$ be a non-negative symmetric matrix whose positive entry w_{ij} is the weight of edge $(i, j) \in \mathcal{E}$. It is assumed that not all workers are benign. An *unknown* group of workers are *Byzantine* which can send faulty/malicious values to their neighbors during the learning process. We use $\mathcal{R}$ and $\mathcal{B}$ to denote the sets of benign workers and Byzantine workers, respectively. We define the numbers of benign and Byzantine workers by $R = |\mathcal{R}|$ and $B = |\mathcal{B}|$. For worker i, we use $\mathcal{N}_i$ to denote the set of all its neighbors and further define $\overline{\mathcal{N}}_i := \mathcal{N}_i \cup \{i\}$. We next provide a few standard definitions and assumptions.

Definition 1. *A subgraph $\mathcal{G}_\mathcal{R}$ of $\mathcal{G}$ is called the Byzantine-free graph if it is generated by removing all Byzantine workers along with their edges from $\mathcal{G}$.*

Assumption 1. *The Byzantine-free graph $\mathcal{G}_\mathcal{R}$ generated from $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is connected.*

Compared to the setting in prior works (Su and Vaidya 2016; Yang and Bajwa 2019) that assume each benign worker receives messages with an honest majority, Assumption 1 is weaker and has also been used before (Peng and Ling 2020; He, Karimireddy, and Jaggi 2022). The above assumption implies that the original graph $\mathcal{G}$ is also connected.

Byzantine-Robust Decentralized Learning

We consider the following optimization problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}) := \frac{1}{R}\sum_{i\in\mathcal{R}} \left\{ f_i(\boldsymbol{x}) := \mathbb{E}_{\xi_i\sim\mathcal{D}_i} F_i\left(\boldsymbol{x}; \xi_i\right)\right\}, \quad (1)$$

where $\boldsymbol{x} \in \mathbb{R}^d$ is the model parameter, f_i is the local objective function of benign worker i and $F_i\left(\boldsymbol{x}; \xi_i\right)$ denotes the loss function for model parameter $\boldsymbol{x}$ for the random data sample ξ_i drawn from distribution $\mathcal{D}_i$. Note that the data distribution is non-IID across the workers, i.e., $\mathcal{D}_i \neq \mathcal{D}_j$ for $i, j \in \mathcal{V}, i \neq j$.

We assume that the gradients and the local objective functions satisfy the following properties.

Assumption 2 (L-smooth)**.** *For $i \in \mathcal{R}$, $f_i(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$ is differentiable and there exists a constant $L \geq 0$ such that for any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$:*

$$\|\nabla f_i(\boldsymbol{x}) - \nabla f_i(\boldsymbol{y})\|_2 \leq L\|\boldsymbol{x} - \boldsymbol{y}\|_2. \tag{2}$$

Assumption 3 (Bounded gradient noise)**.** *There is a $\sigma < \infty$ such that for all $i \in \mathcal{R}$ and $\boldsymbol{x} \in \mathbb{R}^d$,*

$$\max_{i \in \mathcal{R}} \|\nabla F_i(\boldsymbol{x}; \xi_i) - \nabla f_i(\boldsymbol{x})\|_2^2 \leq \sigma^2. \tag{3}$$

Assumption 4 (Bounded heterogeneity)**.** *There is a $\zeta < \infty$ such that for all $i \in \mathcal{R}$ and $\boldsymbol{x} \in \mathbb{R}^d$,*

$$\max_{i \in \mathcal{R}} \|\nabla f_i(\boldsymbol{x}) - \nabla f(\boldsymbol{x})\|_2^2 \leq \zeta^2. \tag{4}$$

Assumption 2 is the commonly used smoothness condition. Assumption 3 limits the divergence of the stochastic gradient $\nabla F_i(\boldsymbol{x})$ from the true gradient $\nabla f_i(\boldsymbol{x})$. Assumption 4, on the other hand, restricts the deviation of the local gradient $\nabla f_i(\boldsymbol{x})$ from the averaged gradient $\nabla f(\boldsymbol{x})$.

To solve the optimization problem (1) in a distributed manner, each benign worker will update its local model parameter and collaborate with neighboring workers according to a specific decentralized learning algorithm. However, it is crucial to note that some workers within the system may be Byzantine. Byzantine workers may broadcast malicious parameters to their neighbors to make the learning process fail. We make the following assumption on Byzantine workers.

Assumption 5. *There is a fixed set of Byzantine workers which are omniscient.*

Under this assumption, Byzantine workers know the entire state of all benign workers and can take advantage of this information to manipulate the models generated by the learning algorithms. Defense against such powerful Byzantine workers is very challenging. During the learning process, each benign worker communicates with its neighbors. One method of facilitating this collaboration is through the mean weighted aggregation, which can be modeled as $\boldsymbol{x}_i^{t+1} = \sum_{j \in \overline{\mathcal{N}}_i} w_{ij} \boldsymbol{x}_j^t$, where $\boldsymbol{x}_i^t$ denotes the model parameter vector of worker i at time t. A few definitions and assumptions on the matrix $\boldsymbol{W}$ and weights are provided below.

Assumption 6. *We assume that $\boldsymbol{W} \in [0,1]^{n \times n}$ is a symmetric doubly stochastic matrix.*

Note that $\sum_{j \in \mathcal{V}} w_{ij} = 1$ for any worker $i \in \mathcal{V}$ and $w_{ij} > 0$ if and only if $j \in \overline{\mathcal{N}}_i$.

Definition 2. *The total weight of Byzantine workers in the neighborhood of benign worker $i \in \mathcal{R}$ is defined by $\delta_i := \sum_{j \in \mathcal{B} \cap \mathcal{N}_i} w_{ij}$. We further define $\delta_{\max} := \max_{i \in \mathcal{R}} \delta_i$.*

In this work, we consider a more general setting where there is a bound $\delta_{\max}$ on the total weight of Byzantine workers in the neighborhood of any benign worker, rather than a bound on the number of Byzantine workers in any such neighborhood, similar to the setting in He, Karimireddy, and Jaggi (2022).

Definition 3. *Given the Byzantine-free graph $\mathcal{G}_{\mathcal{R}}$ of $\mathcal{G}$, we define a Byzantine-free weight matrix $\widetilde{\boldsymbol{W}} \in \mathbb{R}^{(n-B)\times(n-B)}$, where $\widetilde{w}_{ij}$, for $i, j \in \mathcal{R}$, is defined by*

$$\widetilde{w}_{ij} = \begin{cases} w_{ij}, & \text{if} \quad i \neq j, \\ w_{ii} + \delta_i, & \text{otherwise.} \end{cases} \tag{5}$$

By construction, $\widetilde{\boldsymbol{W}}$ is doubly stochastic and has a positive spectral gap γ guaranteed by Assumption 1. $\widetilde{\boldsymbol{W}}$ will be used in the analysis and upper bound calculations.

Preliminary: Decentralized Stochastic Gradient Descent

Following a decentralized learning method, workers in $\mathcal{V}$ iteratively optimize their local model parameters and collaborate to reach consensus. As a preliminary, we first describe a widely used method, namely, decentralized stochastic gradient descent (DSGD) (Nedic and Ozdaglar 2009). In general, DSGD contains three steps at each iteration, including computation, communication and aggregation. Specifically, at the t-th iteration, worker i has its local model parameter $\boldsymbol{x}_i^t$ and updates an intermediate variable $\boldsymbol{x}_i^{t+\frac{1}{2}}$ as

$$\boldsymbol{x}_i^{t+\frac{1}{2}} = \boldsymbol{x}_i^t - \eta \boldsymbol{g}_i^t, \tag{6}$$

where $\boldsymbol{g}_i^t = \nabla F_i(\boldsymbol{x}_i^t; \xi_i^t)$ is the stochastic gradient of F_i at $\boldsymbol{x}_i^t$ and η is the step size. Then, worker i broadcasts $\boldsymbol{x}_i^{t+\frac{1}{2}}$ to its neighbors. When receiving the model parameters from the neighbors, worker i will update its model by the mean weighted aggregation:

$$\boldsymbol{x}_i^{t+1} = \sum_{j \in \overline{\mathcal{N}}_i} w_{ij} \boldsymbol{x}_j^{t+\frac{1}{2}}. \tag{7}$$

However, in the presence of Byzantine workers, DSGD will fail since messages sent from Byzantine workers can prevent the convergence and such an aggregation rule is vulnerable to Byzantine attacks. To solve this problem, we need to replace the weighted averaging in (7) by more *robust* aggregation rules, in order to mitigate the impact of the Byzantine workers (Yang and Bajwa 2019; Guo et al. 2022; Fang, Yang, and Bajwa 2022; He, Karimireddy, and Jaggi 2022). Specifically, benign worker i updates its estimate at the $(t+1)$-th iteration using a *robust aggregation rule* $\mathrm{AGG}_i\left(\boldsymbol{x}_j^{t+\frac{1}{2}}, j \in \overline{\mathcal{N}}_i\right)$, under which (7) is replaced by

$$\boldsymbol{x}_i^{t+1} = \mathrm{AGG}_i\left(\boldsymbol{x}_j^{t+\frac{1}{2}}, j \in \overline{\mathcal{N}}_i\right). \tag{8}$$

Proposed Robust Decentralized Learning Method: DSGD-RTC

In this section, we introduce our decentralized learning approach, by incorporating our novel robust aggregator in the DSGD scheme with momentum.

Algorithm 1: RTC aggregator at benign worker i

Input: Model parameter $\boldsymbol{x}_i \in \mathbb{R}^d$ for worker $i \in \mathcal{R}$, worker i's neighbor set $\overline{\mathcal{N}}_i$, the set of weights $\{w_{ij} | j \in \overline{\mathcal{N}}_i\}$ and bound $\delta_{\max}$.
Output: $\mathrm{RTC}_i\left(\boldsymbol{x}_j, j \in \overline{\mathcal{N}}_i\right)$.

1: Initialize a removing set $U_i^0 = \emptyset$ and $k = 0$
2: **while** $\sum_{j \in U_i^k} w_{ij} \leq \delta_{\max}$ **do**
3: Set $k = k + 1$
4: Choose index $j = \underset{m \in \overline{\mathcal{N}}_i}{\arg\max} \|\boldsymbol{x}_m - \boldsymbol{x}_i\|_2$
5: Add j into the removing set U_i^k
6: **end while**
7: Construct the set of remaining neighbors $S_i = \overline{\mathcal{N}}_i \backslash U_i^{k-1}$
8: Centered at $\boldsymbol{x}_i$, clip the remaining neighbors' values to obtain $\mathrm{RTC}_i\left(\boldsymbol{x}_j, j \in \overline{\mathcal{N}}_i\right)$ according to (9)
9: **return** $\mathrm{RTC}_i\left(\boldsymbol{x}_j, j \in \overline{\mathcal{N}}_i\right)$.

Algorithm 2: DSGD with RTC aggregator

Input: Initial model parameter $\boldsymbol{x}_i^0 \in \mathbb{R}^d$ for worker $i \in \mathcal{R}$, communication graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and mixing matrix W.
Parameter: Step sizes η and α.
Output: $\boldsymbol{x}_i^T$ for worker $i \in \mathcal{R}$.

1: Initialize $\boldsymbol{x}_i^0 = \boldsymbol{m}_i^0 = 0 \in \mathbb{R}^d$ in parallel for $i \in \mathcal{R}$
2: **for** $t = 0, 1, ..., T-1$ **do**
3: **for** $i \in \mathcal{R}$ **do**
4: Sample ξ_i^t, compute gradient $\boldsymbol{g}_i^t = \nabla F_i(\boldsymbol{x}_i^t; \xi_i^t)$
5: $\boldsymbol{m}_i^{t+1} = (1-\alpha)\boldsymbol{m}_i^t + \alpha \boldsymbol{g}_i^t$
6: $\boldsymbol{x}_i^{t+1/2} = \boldsymbol{x}_i^t - \eta \boldsymbol{m}_i^{t+1}$
7: Worker i exchanges $\boldsymbol{x}_i^{t+\frac{1}{2}}$ with its neighbors
8: $\boldsymbol{x}_i^{t+1} = \mathrm{RTC}_i\left(\boldsymbol{x}_j^{t+\frac{1}{2}}, j \in \overline{\mathcal{N}}_i\right)$
9: **end for**
10: **end for**
11: **return** $\boldsymbol{x}_i^T$ for worker $i \in \mathcal{R}$.

Robust Aggregation via Remove-then-Clip (RTC)

We present a novel robust aggregator, referred to as *RTC*, designed explicitly to enhance the robustness and efficiency of the DSGD algorithm. The RTC aggregator is described in Algorithm 1. RTC involves a careful filtering phase and a clipping phase. In the filtering phase, benign worker i receives messages from its neighbors and sequentially eliminates the most distant ones. Specifically, it initializes a removal set $U_i^0 = \varnothing$ at the beginning of the filtering process. Then, iteratively, worker i identifies the neighbor whose model is the farthest from its own (worker i's) model and adds it to the set U_i^k (Lines 2-6 in Algorithm 1). This process continues until the total weight of the removed neighbors reaches $\delta_{\max}$ (Definition 2).

After the filtering phase, a clipping process is applied to the set of remaining neighbors S_i that have survived the filtering phase. The output of the aggregation rule $\mathrm{RTC}_i\left(\boldsymbol{x}_j, j \in \overline{\mathcal{N}}_i\right)$ at worker i is obtained by clipping the values of workers in S_i centered at $\boldsymbol{x}_i$ and replacing the values of removed neighbors with $\boldsymbol{x}_i$, i.e.,

$$\begin{aligned}&\mathrm{RTC}_i\left(\boldsymbol{x}_j, j \in \overline{\mathcal{N}}_i\right)\\&= \sum_{j \in S_i} w_{ij}\left(\boldsymbol{x}_i + \mathrm{CLIP}(\boldsymbol{x}_j - \boldsymbol{x}_i, \tau_i)\right) + \sum_{j \in \overline{\mathcal{N}}_i \backslash S_i} w_{ij} \boldsymbol{x}_i,\end{aligned} \tag{9}$$

where τ_i is a carefully chosen parameter defined as

$$\tau_i = \sqrt{\frac{1}{\delta_{\max}} \sum_{j \in S_i} w_{ij} \mathbb{E}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|_2^2}, \tag{10}$$

and CLIP is defined as

$$\mathrm{CLIP}\left(\boldsymbol{x}, \tau\right) = \boldsymbol{x} \cdot \min(1, \tau / \|\boldsymbol{x}\|_2^2). \tag{11}$$

Below, we describe how RTC distinguishes itself from the existing methods and outline its key features. Compared with methods such as robust federated aggregation (RFA) (Pillutla, Kakade, and Harchaoui 2022), which primarily focus on finding a median value, our proposed RTC applies clipping method to multiple vectors rather than seeking a single central point, which takes advantage of more information from neighbors. SCCLIP method (He, Karimireddy, and Jaggi 2022) indiscriminately applies a clipping radius $\tau_i' = \sqrt{\frac{1}{\delta_i} \sum_{j \in \mathcal{R}} w_{ij} \mathbb{E}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|_2^2}$ to *all* received neighbor vectors. However, this radius includes the set $\mathcal{R}$, which comprises all benign workers and is *unknown*. In comparison, our proposed RTC method introduces a more sophisticated aggregation. On one hand, by integrating the meticulous filtering scheme with the clipping process, RTC reduces the impact of potential Byzantine attacks without losing significant information from benign neighbors' updates. On the other hand, to deal with the problem of finding clipping radius, RTC constructs the set S_i by employing the filtering process and tactically replaces the unknown set $\mathcal{R}$ in τ_i' with the purposefully constructed set S_i in τ_i. This balanced consideration of feasibility and robustness in integrating RTC offers an enhancement to decentralized learning.

Algorithm 2 describes DSGD with the robust aggregator RTC. Specifically, each regular worker i calculates the stochastic gradient $\boldsymbol{g}_i^t$ in Line 4. Following this step, worker i employs the momentum method in Line 5, which is advantageous for accelerating the convergence rate of DSGD and defending against time-coupled attacks (Karimireddy, He, and Jaggi 2021). Subsequently, worker i advances by calculating the intermediate variable $\boldsymbol{x}_i^{t+\frac{1}{2}}$, which is then exchanged with neighboring workers. The final step involves the application of the RTC aggregator to mitigate potential adversarial influences from Byzantine workers. This process culminates in the derivation of the updated model parameter $\boldsymbol{x}_i^{t+1}$, as presented in Line 8. Note that Byzantine workers can send arbitrary vectors to their neighbors.

Remark 1. *Similar to SCCLIP, our approach is easy to implement, and the computational complexity is $O(|\mathcal{N}_i| d)$ for each worker at each iteration, where d is the dimension of the model parameter vector. At the same time, implementation of methods like Krum requires $O(|\mathcal{N}_i|^2 d)$ computation*

per step (Blanchard et al. 2017).

Remark 2. *RTC only needs an estimate of the upper bound on $\delta_{\max}$. In the absence of an accurate estimate, we can employ an adaptive strategy. We initiate the process with a conservative estimate of $\delta_{\max}$, setting it to a relatively small value. The learning algorithm is then allowed for a number of iterations. If it fails to converge to a local neighborhood, or converges but yields accuracy that falls below the desired threshold, we incrementally adjust $\delta_{\max}$ upwards. This process is repeated until the algorithm consistently demonstrates satisfactory performance. Through this adaptive method, we can effectively estimate a suitable $\delta_{\max}$, while also accommodating the dynamic range of Byzantine behaviors in a practical scenario.*

Analysis of the Proposed DSGD-RTC Method

We first present a lemma showing that the difference between the value returned by RTC and the weighted average of the values from regular neighbors is bounded. Then we provide the main theorem regarding the convergence properties of DSGD with RTC.

Lemma 1. *Assume Assumption 5, 6 hold and choose τ_i according to (10). Let $\hat{\boldsymbol{x}}_i = \mathrm{RTC}_i\left(\boldsymbol{x}_j, j \in \overline{\mathcal{N}}_i\right)$. Then for all $i \in \mathcal{R}$, we have*

$$\mathbb{E}\left\|\hat{\boldsymbol{x}}_i - \sum_{j\in\mathcal{R}} \widetilde{w}_{ij}\boldsymbol{x}_j\right\|_2^2 \leq 10\delta_{\max} \max_{j\in\mathcal{R}\cap\overline{\mathcal{N}}_i} \mathbb{E}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|_2^2. \tag{12}$$

Recall that by Definition 3, $\sum_{j\in\mathcal{R}} \widetilde{w}_{ij}\boldsymbol{x}_j$ is the weighted average of the values of *regular* neighbors of worker i and worker i itself. Note that in Lemma 1 if there are no Byzantine workers ($\delta_{\max} = 0$), RTC aggregator outputs the same value as the weighted average. Also if regular workers have reached consensus ($\max_{j\in\mathcal{R}} \mathbb{E}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|_2^2 = 0$), RTC aggregator maintains the consensus. Note that if we apply SCCLIP aggregator (He, Karimireddy, and Jaggi 2022) and set $\tau_i' = \sqrt{\frac{1}{\delta_i}\sum_{j\in\mathcal{R}} w_{ij}\mathbb{E}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|_2^2}$, then we have,

$$\mathbb{E}\left\|\mathrm{SCCLIP}_i\left(\boldsymbol{x}_j, j \in \overline{\mathcal{N}}_i\right) - \sum_{j\in\mathcal{R}} \widetilde{w}_{ij}\boldsymbol{x}_j\right\|_2^2 \leq 4\delta_i \sum_{j\in\mathcal{R}} w_{ij}\mathbb{E}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|_2^2. \tag{13}$$

Compared to (12), the right-hand side of inequality (13) is constrained by the weighted average of distances $\sum_{j\in\mathcal{R}} w_{ij}\mathbb{E}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|_2^2$, whereas in Lemma 1, RTC aggregator employs the maximum term $\max_{j\in\mathcal{R}} \mathbb{E}\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|_2^2$. This distinction arises from SCCLIP's use of inaccessible clipping thresholds which result in a tighter bound.

Theorem 1. *Assume Assumption 1-6 hold, $\delta_{\max} = \mathcal{O}\left(\gamma^2\right)$ and $\alpha := 4\eta L$. Then for Algorithm 2, we have*

$$\frac{1}{T+1}\sum_{t=0}^{T}\left\|\nabla f\left(\bar{\boldsymbol{x}}^t\right)\right\|_2^2 \leq \mathcal{O}\left(\frac{\delta_{\max}\zeta^2}{\gamma^2} + \left(\frac{(\frac{\delta_{\max}}{\gamma^2} + \frac{1}{R})\sigma^2}{T}\right)^{\frac{1}{2}} + \left(\frac{\zeta}{T\gamma}\right)^{\frac{2}{3}} + \left(\frac{\sigma^{\frac{2}{3}}}{T\gamma^{\frac{2}{3}}}\right)^{\frac{3}{4}} + \frac{1}{T}\right). \tag{14}$$

The above theorem shows that DSGD-RTC converges to $\mathcal{O}\left(\delta_{\max}\zeta^2/\gamma^2\right)$-neighborhood of a stationary point after a sufficiently large time T. Note that $\Omega\left(\delta_{\max}\zeta^2\right)$ is the lower bound on the convergence results in the Byzantine-robust distributed learning (Karimireddy, He, and Jaggi 2022). Theorem 1 establishes that even though RTC aggregator adopts a feasible clipping threshold, leading to a less stringent bound as detailed in Lemma 1, Algorithm 2 reliably attains convergence in the non-convex and non-IID setting.

Experiments

In this section, we empirically show the superior performance of our proposed method and compare it with existing robust aggregators as well as an upper bound based on an ideal aggregator which knows the identity of Byzantine workers.

Experiment Setup

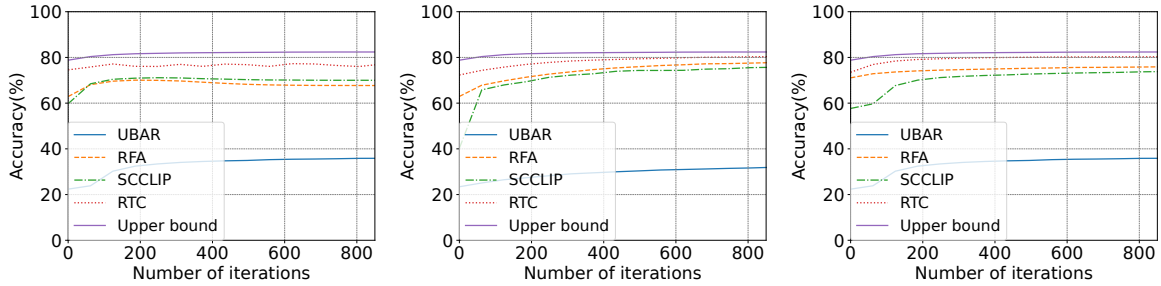
Network Topology. The network topology of the decentralized system in our experiments is an Erdos–Renyi (ER) graph with 30 nodes, where each edge is included in the graph with connection probability p. To ensure the connectivity assumption, we first generate a decentralized network of benign nodes in which all the workers strictly follow the learning procedure. Then we randomly add different numbers of Byzantine workers to the network and connect them to the benign nodes with the same connection probability p. Consistent with the setting adopted in the recent study (He, Karimireddy, and Jaggi 2022), we construct w_{ij} as follows,

$$w_{ij} = \begin{cases} \frac{1}{d_{\max}+1}, & \text{if} \quad j \in \mathcal{N}_i, \\ 1 - \frac{|\mathcal{N}_i|}{d_{\max}+1}, & \text{if} \quad j = i, \\ 0, & \text{otherwise}, \end{cases} \tag{15}$$

where $d_{\max}$ is the maximum degree of nodes in the graph.

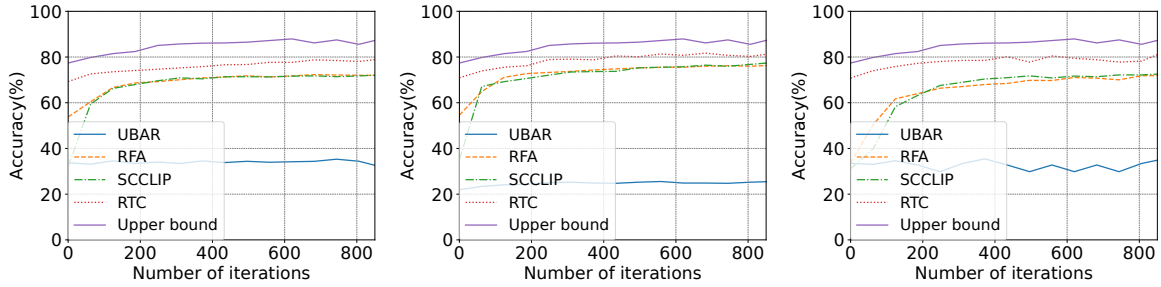
Dataset. We consider MNIST (LeCun and Cortes 2010), which contains ten handwritten digits from 0 to 9, with 60,000 training images and 10,000 testing images. For IID data distribution, the training dataset is partitioned equally among all workers, while for non-IID data distribution, the dataset is sorted by labels and sequentially divided into equal parts for all workers.

Optimization Objective. We consider both convex and non-convex loss functions. In the convex setting, we use



(a) IPM (b) ALIE (c) DISSENSUS

Figure 1: Accuracy of different aggregators in the ER graph with 5 Byzantine workers under three attacks in the convex setting



(a) IPM (b) ALIE (c) DISSENSUS

Figure 2: Accuracy of different aggregators in the ER graph with 5 Byzantine workers under three attacks in the non-convex setting

the multi-class logistic regression problem to test the performance of algorithms. The loss function is given by,

$$f(\boldsymbol{x}) = -\left[\sum_{i=1}^{N}\sum_{k=1}^{K} 1\left\{b_i = k\right\} \log \frac{\exp\left(\boldsymbol{x}^{(k)\top} a_i\right)}{\sum_{j=1}^{K} \exp\left(\boldsymbol{x}^{(j)\top} a_i\right)}\right] + \frac{\theta}{2}\|\boldsymbol{x}\|_2^2, \tag{16}$$

where $1\{\}$ is the indicator function, $a_i \in \mathbb{R}^m$ and $b_i \in \{1, \cdots, K\}$ are the input vector and the label of data samples, K is the number of classes, N is the total number of data samples, $\boldsymbol{x}$ is an $m \times K$ matrix, of which the j-th column is $\boldsymbol{x}^{(j)}$ and θ is the regularization parameter. In the decentralized setting, all the benign workers use the loss function (16) but with different data samples. In the setting of non-convex, we use a 6-layer neural network as the training model.

Benchmarks. We consider existing robust aggregators with GD including: (1) RFA (Pillutla, Kakade, and Harchaoui 2022), (2) UBAR (Guo et al. 2022), and (3) SCCLIP (He, Karimireddy, and Jaggi 2022). Since coordinate-wise trimmed mean (Yang and Bajwa 2019) and Krum (Blanchard et al. 2017) usually perform worse than RFA, we exclude them in the experiments.

In addition, we introduce an upper bound on the performance of any Byzantine-robust algorithm. To establish the upper bound, we use the mean weighted aggregation from (7) *based on the Byzantine-free weight matrix* $\widetilde{\boldsymbol{W}}$, as defined in Definition 3.

Byzantine Attacks. We evaluate the performance of algorithms under several widely used attacks:

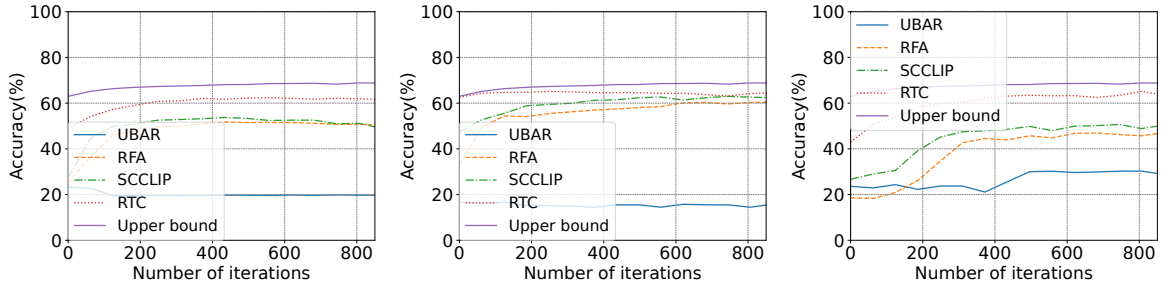
- *Inner Product Manipulation (IPM)* attack (Xie, Koyejo, and Gupta 2020a): IPM attack is proposed to make all Byzantine workers send the same corrupted gradient $\boldsymbol{u}$ based on the benign gradients in the distributed settings. Thus, in the decentralized settings, following the modification detailed in (He, Karimireddy, and Jaggi 2022), if $j \in \mathcal{B}$ is a Byzantine neighbor of the benign worker i, $\boldsymbol{x}_j$ is computed as

$$\boldsymbol{x}_j = \boldsymbol{x}_i - \varepsilon \sum_{k \in \mathcal{R} \cap \mathcal{N}_i} w_{ik} \left(\boldsymbol{x}_k - \boldsymbol{x}_i\right), \tag{17}$$

  where ε is a tuning parameter.
- *A Little Is Enough (ALIE)* attack (Baruch, Baruch, and Goldberg 2019): The Byzantine workers estimate the mean μ_i and standard deviation σ_i of the benign models, and send $\mu_i - z\sigma_i$ to benign worker i where z is a constant controlling the strength of the attack. The parameter z is computed as

$$z = \max_z \left(\phi(z) < \frac{n - B - s}{n - B}\right), \tag{18}$$

  where $s = \left\lfloor \frac{n}{2} + 1 \right\rfloor - B$ and ϕ is the cumulative standard normal function.



(a) IPM (b) ALIE (c) DISSENSUS

Figure 3: Accuracy of different aggregators in the ER graph with 10 Byzantine workers under three attacks in the non-convex setting

- *DISSENSUS* attack (He, Karimireddy, and Jaggi 2022): Let $\varepsilon_i > 0$ for all $i \in \mathcal{R}$. The Byzantine worker $j \in \mathcal{B} \cap \mathcal{N}_i$ sends

$$\boldsymbol{x}_j := \boldsymbol{x}_i - \varepsilon_i \frac{\sum_{k \in \mathcal{R} \cap \mathcal{N}_i} w_{ik} \left(\boldsymbol{x}_k - \boldsymbol{x}_i\right)}{\sum_{l \in \mathcal{B} \cap \mathcal{N}_i} w_{il}}. \tag{19}$$

Evaluation Results

We evaluate the defense efficacy of various aggregators under three attacks and non-IID datasets in the convex and non-convex settings.

Performance comparison with 5 Byzantine workers under different Byzantine attacks in the convex settings. The network is structured as an ER graph comprising 30 workers, among which 5 are Byzantine workers. It features a connection ratio $p = 0.5$ and $\delta_{\max} = 0.2$. To evaluate the performance of various methods, Fig. 1 is presented. This figure features the number of iterations on the x-axis and the accuracy achieved on test datasets on the y-axis. In the convex setting, as shown in Fig. 1, our method, RTC performs best under all three attacks and is relatively close to the upper bound. More specifically, under IPM attack, our algorithm improves accuracy by 10% over SCCLIP and 14% over RFA. Under ALIE attack, our algorithm's performance exceeds that of SCCLIP and RFA by 6% and 3%, respectively. Under DISSENSUS attack, our proposed algorithm surpasses SCCLIP by an accuracy margin of 9%, and outperforms RFA by 6%. Additionally, UBAR remains with an accuracy of less than 40% under three attacks.

Performance comparison with 5 Byzantine workers under different Byzantine attacks in the non-convex settings. Utilizing the established ER graph framework with 30 workers, including 5 Byzantine workers, we further investigate the performance of various methods in the non-convex setting, as shown in Fig. 2. We observe that our method ranks as the best under three attacks. Specifically, under IPM attack, our proposed algorithm exhibits an accuracy improvement of 10% over both SCCLIP and RFA. Under ALIE attack, our algorithm gains 5% over SCCLIP and 7% over RFA in accuracy. Under DISSENSUS attack, our algorithm gains 13% over SCCLIP and 14% over RFA in accuracy. It is noteworthy that UBAR consistently exhibits the least effectiveness under all three attacks.

Performance comparison with 10 Byzantine workers under different Byzantine attacks in the non-convex settings. We modify our network setting to ER graph with 30 workers, of which 10 are Byzantine workers. This network is characterized by a connection ratio $p = 0.5$ and $\delta_{\max} = 0.3$. In this adjusted setting, we evaluate the performance of various methods against 10 Byzantine workers in the non-convex environment, as depicted in Fig. 3. Our results indicate that our method consistently achieves the best performance across all three distinct attack scenarios. Specifically, under IPM attack, our algorithm shows a 26% improvement in accuracy over SCCLIP and 22% over RFA. In the case of ALIE attack, there is a 4% gain over SCCLIP and 7% over RFA. Under DISSENSUS attack, the improvement is even more pronounced, with 35% over SCCLIP and 26% over RFA. Notably, UBAR shows the least effectiveness in all three attack scenarios. Moreover, in scenarios with 10 Byzantine workers, the improvement of our method over other methods is significantly magnified compared to the setting of 5 Byzantine workers, especially under IPM and DISSENSUS attacks. This notable enhancement in performance under more intensive Byzantine attacks demonstrates our method's heightened resilience and adaptability to stronger attack environments compared to the past aggregators.

Conclusion

In this paper, we introduced a decentralized learning method that leverages DSGD in conjunction with a novel robust aggregator. We provided theoretical guarantees on the convergence of our algorithm. Extensive evaluations demonstrated that our method offers significant improvements over existing methods in heterogeneous and Byzantine environments. A possible future work could include more careful analysis or other approaches to relax the dependence of convergence guarantees on heterogeneity.

Acknowledgements

This research was supported by ARO Grant W911NF1910379.

References

Alghunaim, S. A.; and Sayed, A. H. 2020. Linear Convergence of Primal–Dual Gradient Methods and Their Performance in Distributed Optimization. *Automatica*, 117: 109003.

Assran, M.; Loizou, N.; Ballas, N.; and Rabbat, M. 2019. Stochastic Gradient Push for Distributed Deep Learning. In *International Conference on Machine Learning*, 344–353. PMLR.

Baruch, G.; Baruch, M.; and Goldberg, Y. 2019. A Little is Enough: Circumventing Defenses for Distributed Learning. *Advances in Neural Information Processing Systems*, 32.

Blanchard, P.; El Mhamdi, E. M.; Guerraoui, R.; and Stainer, J. 2017. Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. *Advances in Neural Information Processing Systems*, 30.

Chen, Y.; Su, L.; and Xu, J. 2017. Distributed Statistical Machine Learning in Adversarial Settings: Byzantine Gradient Descent. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 1(2): 1–25.

Di Lorenzo, P.; and Scutari, G. 2016. Next: In-Network Nonconvex Optimization. *IEEE Transactions on Signal and Information Processing over Networks*, 2(2): 120–136.

Fang, C.; Yang, Z.; and Bajwa, W. U. 2022. BRIDGE: Byzantine-Resilient Decentralized Gradient Descent. *IEEE Transactions on Signal and Information Processing over Networks*, 8: 610–626.

Guerraoui, R.; Rouault, S.; et al. 2018. The Hidden Vulnerability of Distributed Learning in Byzantium. In *International Conference on Machine Learning*, 3521–3530. PMLR.

Guo, S.; Zhang, T.; Yu, H.; Xie, X.; Ma, L.; Xiang, T.; and Liu, Y. 2022. Byzantine-Resilient Decentralized Stochastic Gradient Descent. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(6): 4096–4106.

He, L.; Karimireddy, S. P.; and Jaggi, M. 2022. Byzantine-Robust Decentralized Learning via Self-Centered Clipping. *arXiv preprint arXiv:2202.01545*.

Hegedűs, I.; Danner, G.; and Jelasity, M. 2021. Decentralized Learning Works: An Empirical Comparison of Gossip Learning and Federated Learning. *Journal of Parallel and Distributed Computing*, 148: 109–124.

Hsieh, K.; Phanishayee, A.; Mutlu, O.; and Gibbons, P. 2020. The Non-IID Data Quagmire of Decentralized Machine Learning. In *International Conference on Machine Learning*, 4387–4398. PMLR.

Kairouz, P.; McMahan, H. B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A. N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. 2021. Advances and Open Problems in Federated Learning. *Foundations and Trends® in Machine Learning*, 14(1–2): 1–210.

Karimireddy, S. P.; He, L.; and Jaggi, M. 2021. Learning from History for Byzantine Robust Optimization. In *International Conference on Machine Learning*, 5311–5319. PMLR.

Karimireddy, S. P.; He, L.; and Jaggi, M. 2022. Byzantine-Robust Learning on Heterogeneous Datasets via Bucketing. In *9th International Conference on Learning Representations (ICLR)*.

Koloskova, A.; Lin, T.; Stich, S. U.; and Jaggi, M. 2020. Decentralized Deep Learning with Arbitrary Communication Compression. In *International Conference on Learning Representations*.

Konečnỳ, J.; McMahan, H. B.; Ramage, D.; and Richtárik, P. 2016. Federated Optimization: Distributed Machine Learning for On-Device Intelligence. *arXiv preprint arXiv:1610.02527*.

Lalitha, A.; Shekhar, S.; Javidi, T.; and Koushanfar, F. 2018. Fully Decentralized Federated Learning. In *Third workshop on Bayesian Deep Learning (NeurIPS)*.

LeCun, Y.; and Cortes, C. 2010. MNIST Handwritten Digit Database.

Li, Q.; Kailkhura, B.; Goldhahn, R.; Ray, P.; and Varshney, P. K. 2017. Robust Federated Learning Using ADMM in the Presence of Data Falsifying Byzantines. *arXiv preprint arXiv:1710.05241*.

Lian, X.; Zhang, C.; Zhang, H.; Hsieh, C.-J.; Zhang, W.; and Liu, J. 2017. Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent. *Advances in Neural Information Processing Systems*, 30.

Mateos, G.; Bazerque, J. A.; and Giannakis, G. B. 2010. Distributed Sparse Linear Regression. *IEEE Transactions on Signal Processing*, 58(10): 5262–5276.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Artificial intelligence and statistics*, 1273–1282. PMLR.

Nedic, A.; and Ozdaglar, A. 2009. Distributed Subgradient Methods for Multi-Agent Optimization. *IEEE Transactions on Automatic Control*, 54(1): 48–61.

Peng, J.; and Ling, Q. 2020. Byzantine-Robust Decentralized Stochastic Optimization. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5935–5939. IEEE.

Pillutla, K.; Kakade, S. M.; and Harchaoui, Z. 2022. Robust Aggregation for Federated Learning. *IEEE Transactions on Signal Processing*, 70: 1142–1154.

Regatti, J.; Chen, H.; and Gupta, A. 2020. ByGARS: Byzantine SGD with Arbitrary Number of Attackers. *arXiv preprint arXiv:2006.13421*.

Roy, A. G.; Siddiqui, S.; Pölsterl, S.; Navab, N.; and Wachinger, C. 2019. Braintorrent: A Peer-to-Peer Environment for Decentralized Federated Learning. *arXiv preprint arXiv:1905.06731*.

Shi, W.; Ling, Q.; Wu, G.; and Yin, W. 2015. Extra: An Exact First-Order Algorithm for Decentralized Consensus Optimization. *SIAM Journal on Optimization*, 25(2): 944–966.

Su, L.; and Vaidya, N. 2016. Multi-Agent Optimization in the Presence of Byzantine Adversaries: Fundamental Limits.

In *2016 American Control Conference (ACC)*, 7183–7188. IEEE.

Wu, Z.; Chen, T.; and Ling, Q. 2023. Byzantine-Resilient Decentralized Stochastic Optimization With Robust Aggregation Rules. *IEEE Transactions on Signal Processing*, 71: 3179–3195.

Xie, C.; Koyejo, O.; and Gupta, I. 2020a. Fall of Empires: Breaking Byzantine-Tolerant SGD by Inner Product Manipulation. In *Uncertainty in Artificial Intelligence*, 261–270. PMLR.

Xie, C.; Koyejo, S.; and Gupta, I. 2019. Zeno: Distributed Stochastic Gradient Descent with Suspicion-based Fault-Tolerance. In *International Conference on Machine Learning*, 6893–6901. PMLR.

Xie, C.; Koyejo, S.; and Gupta, I. 2020b. Zeno++: Robust Fully Asynchronous SGD. In *International Conference on Machine Learning*, 10495–10503. PMLR.

Yang, Z.; and Bajwa, W. U. 2019. Byrdie: Byzantine-Resilient Distributed Coordinate Descent for Decentralized Learning. *IEEE Transactions on Signal and Information Processing over Networks*, 5(4): 611–627.

Yin, D.; Chen, Y.; Kannan, R.; and Bartlett, P. 2018. Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates. In *International Conference on Machine Learning*, 5650–5659. PMLR.

Ying, B.; Yuan, K.; Chen, Y.; Hu, H.; Pan, P.; and Yin, W. 2021. Exponential Graph is Provably Efficient for Decentralized Deep Training. *Advances in Neural Information Processing Systems*, 34: 13975–13987.

Zhang, Y.; and Yang, Q. 2021. A Survey on Multi-Task Learning. *IEEE Transactions on Knowledge and Data Engineering*.

Zhao, Y.; Li, M.; Lai, L.; Suda, N.; Civin, D.; and Chandra, V. 2018. Federated Learning with Non-IID Data. *arXiv preprint arXiv:1806.00582*.