F^{2}ED-LEARNING: Good fences make good neighbors

Lun Wang
University of California, Berkeley
wanglun@berkeley.edu

Qi Pang
Hong Kong University of Science and Technology
qpangaa@cse.ust.hk

Shuai Wang
Hong Kong University of Science and Technology
shuaiw@cse.ust.hk

Dawn Song
University of California, Berkeley
dawnsong@cs.berkeley.edu

Abstract

In this paper, we present F^{2}ED-LEARNING, the first federated learning protocol simultaneously defending against both semi-honest server and Byzantine malicious clients. Using a robust mean estimator called FilterL2, F^{2}ED-LEARNING is the first FL protocol providing dimension-free estimation error against Byzantine malicious clients. Besides, F^{2}ED-LEARNING leverages secure aggregation to protect the clients from a semi-honest server who wants to infer the clients' information from the legitimate updates. The main challenge stems from the incompatibility between FilterL2 and secure aggregation. Specifically, to run FilterL2, the server needs to access individual updates from clients while secure aggregation hides those updates from it. We propose to split the clients into shards, securely aggregate each shard's updates and run FilterL2 on the updates from different shards. The evaluation shows that F^{2}ED-LEARNING consistently achieves optimal or sub-optimal performance under three attacks among five robust FL protocols.

1 Introduction

Federated learning (FL) has drawn numerous attention in the past few years as a new distributed learning paradigm. In federated learning, the users collaboratively train a model with the help of a centralized server when all the data is held locally to preserve the users' privacy. The privacy guarantee can be further enhanced using secure aggregation technique [5] which hides the individual local updates and only reveals the aggregated global update. The graceful balance between utility and privacy popularizes federated learning in a variety of sensitive applications such as Google GBoard, healthcare service and self-driving cars.

The above threat model assumes that all the users honestly upload their local updates. However, it is likely that a small number of clients are malicious in a large-scale FL system with tens of thousands of clients. Besides, in most SGD-based FL algorithms used today [14], the centralized server averages the local updates to obtain the global update, which is vulnerable to even only one malicious client. Therefore, a malicious client can arbitrarily craft its update to either prevent the global model from converging or lead it to a sub-optimal minimum. This kind of attack in federated learning is well-studied by [3, 11, 2, 18].

To mitigate these attacks, various Byzantine-robust FL protocols [4, 21, 12, 16] are proposed to reduce the impact of the contaminated updates. These protocols replace trivial aggregators such as averaging with well-designed Byzantine-robust mean estimators. These estimators suppress the influence of the malicious updates and output a mean estimation as accurate as possible. Nevertheless, almost all of these aggregators suffer from the curse of dimensionality. Specifically, the estimation error scales up with the size of the model in a square-root fashion. As a concrete example, a

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

three-layer MLP on MNIST contains more than 50,000 parameters and leads to a 223-fold increase of the estimation error, which is prohibitive in practice. Draco [7] and BULYAN [15] are the only two works that claim to yield dimension-free estimation error. However, Draco is designed for distributed learning and is incompatible with federated learning because it requires redundant updates from each worker. On the other hand, although Bulyan [15] claims to provide dimension-free estimation error, it is based on much stronger assumptions than other works. When the assumptions are relaxed to the common case, Bulyan's estimation error still scales up with the square root of the model size as discussed in Section 2.

In addition, these robust FL protocols have incompatible implementation with secure aggregation techniques. The robust estimators have to access local updates while secure aggregation hides them from the server. Consequently, the system cannot simultaneously protect the server and the clients, but has to place complete trust in either of them. The lack of two-way protection severely harms the people's confidence in FL system and prevents federated learning from being used in many sensitive applications such as home monitoring and self-driving cars.

In this paper, we propose **FED**ERATED **LEARNING** WITH **F**ENCE, abbreviately F^{2}ED-LEARNING. F^{2}ED-LEARNING integrates a robust mean estimator with dimension-free error [17] and secure aggregation [5] to defend against both the Byzantine malicious clients and the semi-honest server. In particular, F^{2}ED-LEARNING is the first Byzantine-robust FL system with dimension-free estimation error. To address the incompatibility, the clients are split into multiple shards, the local updates from the same shard are securely aggregated at the centralized server, and the robust estimator is run on the aggregated local updates from different shards.

2 Loophole in Bulyan & Related Work

Byzantine-robust aggregation has drawn enormous attention in the past few years due to the emergence of various distributed attacks in federated learning. [11] formalize the attack as an optimization problem and successfully migrate the data poisoning attack to federated learning. The proposed attacks even work under Byzantine-robust federated learning. [18] manage to launch data poisoning attack on the multi-task federated learning framework. [3] and [2] even manage to insert backdoor functionalities into the model via local model poisoning or local model replacement.

A variety of Byzantine-robust FL protocols are proposed to defend against these attacks. Krum [4] picks the subset of updates with enough close neighbors and averages the subset. [21] leverage traditional robust estimators like trimmed mean or median to achieve order-optimal statistical error rate under strongly convex assumptions. [20] propose to use robust mean estimators to defend against saddle point attack. [15] pointed out that Krum, trimmed mean and median all suffers from $\mathcal{O}(\sqrt{d})$ (d is the model size) estimation error and proposed a general framework Bulyan to reduce the error to $\mathcal{O}(1)$. However, we point out that the improvement of Bulyan actually comes from its stronger assumption. In particular, Bulyan assumes that expectation of the distance between two benign updates is bounded by a constant σ_1, while Krum assumes that the distance is bounded by $\sigma_2\sqrt{d}$. We can easily see that if $\sigma_1 = \sigma_2\sqrt{d}$, Bulyan falls back to the same order of estimation error as Krum. The same loophole exists in the analysis of ByzantineSGD [1]. Consequently, there is no known federated learning protocol with dimension-free estimation error against Byzantine adversaries.

3 Problem Setup

In this section, we review the general pipeline of federated learning, introduce the threat model, and establish the notation system. We use bold lower-case letters (*e.g.* **a**,**b**,**c**) to denote vectors, and bold upper-case letters (*e.g.* **A**, **B**, **C**) for matrices. We denote $1 \cdots n$ with $[n]$.

Federated Learning Pipeline. In a federated learning system, there are one server $\mathcal{S}$ and m clients $\mathcal{C}_i, i \in [m]$. Each client holds data samples drawn from some unknown distribution $\mathcal{D}$. Let $\ell(\mathbf{w}; \mathbf{z})$ be the loss function on the model parameter $\mathbf{w} \in \mathbb{R}^d$ and a data sample $\mathbf{z}$. Let $\mathcal{L}(\mathbf{w}) = \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}[\ell(\mathbf{w}; \mathbf{z})]$ be the population loss function. Our goal is to learn the model $\mathbf{w}$ such that the population loss function is minimized:

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

To learn $\mathbf{w}^*$, the whole system runs a T-round federated learning protocol. Initially, the server stores a global model $\mathbf{w}_0$. In the t^{th} round, $\mathcal{S}$ broadcasts the global model $\mathbf{w}_{t-1}$ to the m clients. The clients then run the local optimizers (*e.g.* SGD, Adam, and RMSprop), compute the difference $\mathbf{g}_t^{(i)}$ between the optimized model and the global model, and upload the difference to $\mathcal{S}$. In the t^{th} round, the server takes the average of the differences and updates the global model $\mathbf{w}_t = \mathbf{w}_{t-1} + \frac{1}{m}\sum_{i=1}^{m} \mathbf{g}_t^{(i)}$.

Threat Model. We assume that the centralized server $\mathcal{S}$ is semi-honest. The server can launch whatever attacks such as inference attack using legitimate updates from the clients as the only inputs. However, the server cannot deviate from the protocol for the sake of regulation or reputation pressure. On the other hand, we assume that the clients are ϵ-Byzantine malicious, which means at most ϵm clients can be malicious. Malicious clients can arbitrarily deviate from the protocol and tamper with their own updates without being detected.

4 F^{2}ED-LEARNING: Robust Privacy-Preserving Distributed FL

In this section, we formally present our main protocol: F^{2}ED-LEARNING. We introduce F^{2}ED-LEARNING step by step and formally establish the robustness and security guarantees.

4.1 F^{2}ED-LEARNING: Byzantine-Robust Privacy-Preserving Federated Learning

The complete F^{2}ED-LEARNING protocol is presented in Algorithm 1. F^{2}ED-LEARNING iteratively executes the following steps: (1) the server broadcasts the global model to the clients; (2) clients train the global model with their local data; (3) clients in the same shard run secure aggregation protocol to upload the mean of their updates to the server; (4) the server aggregates the received updates using robust mean estimation; (5) the server updates the global model with the aggregated global update. We highlight step (3) and (4) newly proposed in F^{2}ED-LEARNING.

Algorithm 1: F^{2}ED-LEARNING: Robust Privacy-Preserving Sharded Federated Learning.

1 **for** $t \leftarrow [T]$ **do**
2 **Server:**
3 Split m clients into p shards $\{H_j\}_{j\in[p]}$
4 Broadcast $\{H_j\}_{j\in[p]}$ and the global model $\mathbf{w}_{t-1}$ to all the clients
5 **Client:**
6 **foreach** *client* $i \in [m]$ **do**
7 Locate its own shard j
8 Generate random masks $\mathbf{u}_{ik}^{(j)}, k \in H_j/i$
9 **foreach** $k \in H_j/i$ **do**
10 Send $\mathbf{u}_{ik}$ to k
11 Train the local model $\mathbf{w}_t^{(i)}$ using $\mathbf{w}_t$ as initialization
12 $\mathbf{g}_t^{(i)} = \mathbf{w}_t^{(i)} - \mathbf{w}_{t-1} + \sum_{k\neq i, i\in H_j, k\in H_j} \mathbf{u}_{ik}^{(j)} - \sum_{k\neq i, i\in H_j, k\in H_j} \mathbf{u}_{ki}^{(j)}$
13 Send $\mathbf{g}_t^{(i)}$ to the server
14 **Server:**
15 **foreach** $H_j \in \{H_j\}_{j\in[p]}$ **do**
16 $\mathbf{g}_t^{H_j} = \sum_{k\in H_j} \mathbf{g}_t^{(k)}$
17 $\mathbf{g}_t = \text{FilterL2}(\{\mathbf{g}_t^{H_j}\}_{j\in[p]})$
18 $\mathbf{w}_t = \mathbf{w}_{t-1} + \mathbf{g}_t$

Sharded Secure Aggregation (line 8-10, 12, 16). Secure aggregation is developed by [5] to defend against the honest but curious server in federated learning. Secure aggregation allows the server to obtain the sum of the clients' updates but hides the individual updates cryptographically. We introduce an oversimplified version of secure aggregation as follow for the ease of clarification. As the first step, each client samples random values for the other clients and send the values to the corresponding clients (line 8-10). After receiving all the values from other clients, each client sums up

the received values and subtracts the values generated by itself to produce a random mask (line 12). Each client blinds its local update with the random mask and sends the blinded update to the server (line 13). The server then sums up all the blinded updates obtains the summed update in plaintext (line 16). Obviously, all the masks cancel out during aggregation and the server receives the plaintext sum. Secure aggregation provides strong privacy guarantee for the clients that the server cannot see anything but the aggregated global update and each client is hidden in thousands of other clients.

However, in our threat model, vanilla secure aggregation is insufficient since it provides no protection for the server. As the individual updates are completely hidden, there is no way that the server can identify the malicious clients even after detecting the attack. To address the issue, we propose to split the clients into multiple shards and run secure aggregation within each shard. The size of the shards provides a trade-off between the protection for the server and the protection for the clients. The smaller the size is, the more information is revealed to the server, thus the easier to defend against Byzantine malicious clients and the harder to fight off the semi-honest server.

Robust Mean Estimation (line 17). The core step in Byzantine-robust federated learning is to estimate the true mean of the benign updates as accurate as possible even with some malicious clients. The most commonly used aggregator, averaging, is proven to be vulnerable to even only one malicious client. All other works addressing the issue such as Krum [4] and Bulyan [15] suffer from a dimension-dependent estimation error. Such error is unacceptable even for training a 3-layer MLP on MNIST, not to mention more complicated tasks and models such as VGG16 or ResNet50.

Actually, the above problem is well studied in statistics under the name "robust mean estimation" and there already exist several robust mean estimators with dimension-free estimation error ([9, 6, 17, 8, 10]). Therefore, instead of reinventing the wheel, we choose to leverage a representative robust mean estimator: FilterL2 (Algorithm 2).

Algorithm 2: FilterL2: dimension-free robust mean estimation [17].

Input: $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d$, $\eta > 1$

1 Let $c_1, \cdots, c_n = 1$

2 $\hat{\boldsymbol{\mu}}_c = (\sum_{i=1}^n c_i \mathbf{x}_i)/(\sum_{i=1}^n c_i)$

3 $\hat{\boldsymbol{\Sigma}}_c = (\sum_{i=1}^n c_i (\mathbf{x}_i - \hat{\mu}_c)(\mathbf{x}_i - \hat{\mu}_c)^\top)/(\sum_{i=1}^n c_i)$

4 Let v be the maximum eigenvector of $\hat{\boldsymbol{\Sigma}}_c$, and let $\hat{\sigma}_c^2 = \mathbf{v}^\top \hat{\boldsymbol{\Sigma}}_c \mathbf{v}$

5 **if** $\hat{\sigma}_c^2 \leq \eta\sigma^2$ **then** return $\hat{\boldsymbol{\mu}}_c$

6 **else** let $\tau = \langle \mathbf{x}_i - \hat{\boldsymbol{\mu}}_c \rangle^2$, and update $c_i \leftarrow c_i \cdot (1 - \tau_i/\tau_{\max})$, where $\tau_{\max} = \max_i \tau_i$

7 Go back to line 2

Specifically, FilterL2 assigns each update a weight and iteratively updates the weights until the weights for the malicious updates are small enough. As mentioned, FilterL2 provides dimension-free error rate formally presented as follow.

Theorem 1 ([17])**.** *Let $\mathcal{D}$ be the honest dataset and $\mathcal{D}^*$ be the contaminated version of $\mathcal{D}$ by inserting malicious samples. Suppose that $|\mathcal{D}^*| \leq |\mathcal{D}|/(1-\epsilon), \epsilon \leq \frac{1}{12}$, and further suppose that $\textsc{Mean}[\mathcal{D}] = \boldsymbol{\mu}$ and $\|\textsc{Cov}[\mathcal{D}]\|_{op} \leq \sigma^2$. Then given $\mathcal{D}^*$, Algorithm 2 outputs $\hat{\boldsymbol{\mu}}$ s.t. $\|\hat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2 = \mathcal{O}(\sigma\sqrt{\epsilon})$ using $\textsc{poly}(n, d)$ time.*

Although Algorithm 2 only takes polynomial time to run, the per-round time complexity is $\mathcal{O}(nd^2)$ if implemented with power iteration. Given d is large, the running time is still quite expensive in practice. To address the issue, we cut the update vectors into k sections and apply the robust estimator to each of the sections. The acceleration scheme reduces the per-round running time to $\mathcal{O}(nd^2/k)$ but increases the estimation error to $\mathcal{O}(\sigma\sqrt{k})$. For instance, if we take $k = \sqrt{d}$, the per-round running time becomes $\mathcal{O}(nd)$ while the estimation error grows to $\mathcal{O}(\sigma\sqrt[4]{\sigma^2 d})$. Despite the compromise for acceleration, FilterL2 still gives the known optimal estimation error and outperforms other robust FL protocols by multiple magnitudes.

4.2 Robustness & Security Analysis

In this section, we rigorously present the security and robustness guarantee of $\text{F}^2\textsc{ed-Learning}$.

Security Guarantee. We first give the security guarantee of F^2ED-LEARNING as follow. Intuitively, no more information about the clients except the averaged updates from the shards is revealed to the centralized server. Thus, each client's update is hidden in all the other clients in its shard.

Theorem 2 (Security against honest-but-curious server; Informal)**.** *There exists a PPT (probabilistic polynomial Turing machine) simulator which can only see the averaged updates from the shards and its output is computationally indistinguishable from the transcript of* F^2ED-LEARNING.

Robustness Guarantee. We now give the formal robustness guarantee of F^2ED-LEARNING. Intuitively, if the number of shards containing malicious clients is small enough, F^2ED-LEARNING can provide mean estimation with dimension-free error (or quad-root error with the acceleration).

Theorem 3 (Robustness against Byzantine adversaries)**.** *Given the number of clients* m*, the number of shards* p *and the fraction of corrupted clients* ϵ*,* F^2ED-LEARNING *provides a mean estimation with dimension-free error as long as* $12\epsilon m < p$*.*

5 Evaluation

In this section, we want to answer the following questions using empirical evaluation: (1) Does FilterL2 outperforms other aggregators when used alone? (2) Does F^2ED-LEARNING outperform other robust FL protocols augmented with sharded secure aggregation?

5.1 Attacks

To answer the above questions, we evaluated the robust estimators without attack and with several representative attacks. We focus on three of these attacks.

The first and second attacks we used are the model poisoning attacks from [11]. The aim of the model poisoning attacks is to increase the error rate of the converged model even facing Byzantine-robust protocols. In these attacks, the malicious clients search for poisoning updates by solving an optimization problem. We employ two attacks proposed in their work targeting at Krum and Trimmed Mean. These two attacks are henceforth referred to as Krum attack (KA) and trimmed mean attack (TMA).

The third attack we considered is a backdoor attack from [3]. The attack aims to insert a backdoor functionality while preserving high accuracy on the validation set. Similarly, the search for the attack gradient is formalized as an optimization problem and the authors tweak the objective function with some stealth metrics to make the attack gradient hard to detect. We refer to the attack as Model Poisoning Attack (MPA) in the rest of the section.

5.2 Experimental Setup

We selected two datasets: MNIST [13] and FashionMNIST [19], and three other Byzantine-robust federated learning protocols to compare with: (1) Krum [4]; (2) Trimmed Mean [21]; and (3) Bulyan [15]. Note that Bulyan acts like a wrapper around other robust estimators so in the evaluation we have two versions of Bulyan: Bulyan Krum and Bulyan Trimmed Mean. We ran all the protocols on the two datasets and present the attack performance under these protocols. Attack performance is measured differently according to the different attack targets. For KA and TMA, we use the model accuracy as the metric for characterizing attack performance. Higher model accuracy indicates stronger robustness. For MPA, we use the percentage of the remembered backdoors to represent the attack performance. The fewer backdoors remembered, the more robust the estimator is.

5.3 Evaluation Results

FilterL2 Performance. To answer question (1), we evaluated 6 aggregators on MNIST and FashionMNIST as shown in Figure 1. We ran the protocols with 20 clients with IID data distributions. 5 of the clients are malicious under attacks. Not surprisingly, FilterL2 achieves optimal performance among all 6 aggregators. Besides, FilterL2 is the only aggregator that consistently achieves good performance under all three attacks. The superiority of FilterL2 is owed to its quad-root estimation

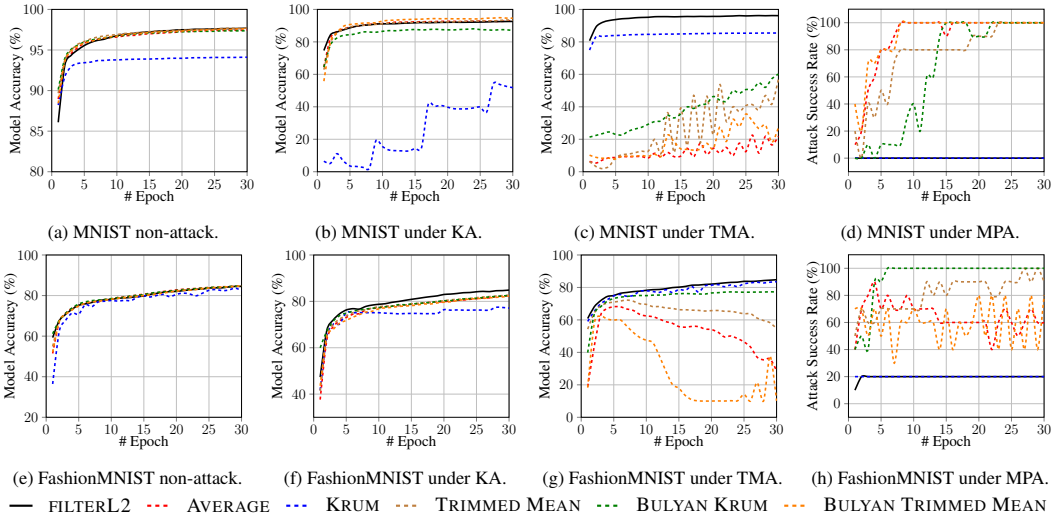

Figure 1: Attack performance under different Byzantine-robust estimators with IID data distribution.

error. Due to the theoretically stronger robustness, it is extremely hard to design targeted attacks for FilterL2 like Krum or trimmed mean.

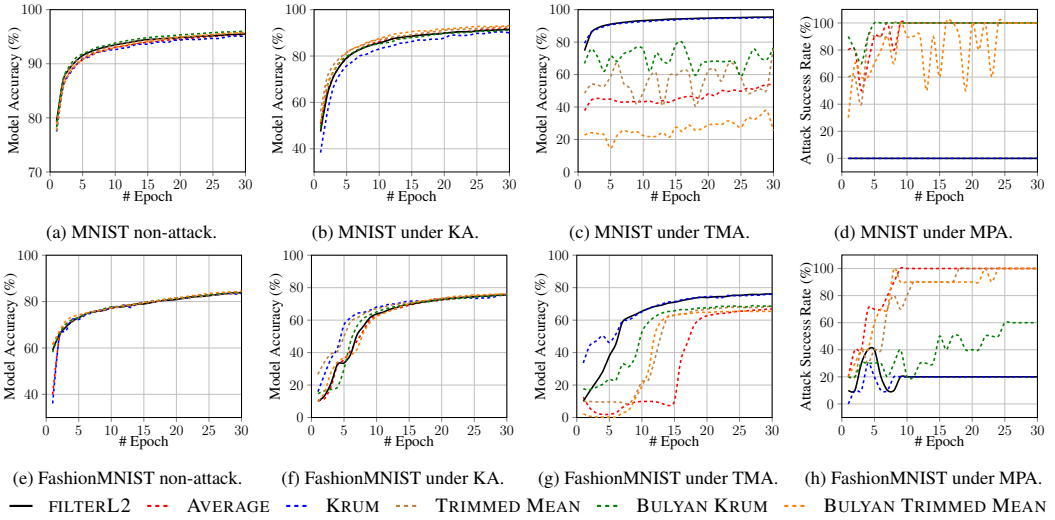

Figure 2: Attack performance under different estimators with sharded secure aggregation with IID data distribution.

F^{2}ED-LEARNING **Performance.** To answer question (2), we evaluated six aggregators with sharding on MNIST and FashionMNIST as shown in Figure 2. We ran the protocols with 100 clients, ten of which are malicious under attacks. The 100 clients were randomly split into 25 shards.

For the experiments without attack, with TMA or with MPA (Figure 2a,2c,2d,2e.2g,2h), F^{2}ED-LEARNING still achieves optimal or close-to-optimal performance. An interesting phenomenon is that KA can be successfully defended by all aggregators when the clients are sharded (Figure 2b,2f). The reason is that KA is targeted at Krum without sharding and wants to maximize the probability that a malicious update is chosen by Krum.

Once integrated with sharding, Krum selects from the averaged updates from the shards, and thus the effect of the malicious update is diluted. This demonstrates that sharding itself can defend against some attacks by diluting the effect of malicious updates.

References

[1] Dan Alistarh, Zeyuan Allen-Zhu, and Jerry Li. Byzantine stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 4613–4623, 2018.

[2] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2938–2948, 2020.

[3] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*, pages 634–643, 2019.

[4] Peva Blanchard, Rachid Guerraoui, Julien Stainer, et al. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Advances in Neural Information Processing Systems*, pages 119–129, 2017.

[5] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

[6] Moses Charikar, Jacob Steinhardt, and Gregory Valiant. Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 47–60, 2017.

[7] Lingjiao Chen, Hongyi Wang, Zachary Charles, and Dimitris Papailiopoulos. Draco: Byzantine-resilient distributed training via redundant gradients. *arXiv preprint arXiv:1803.09877*, 2018.

[8] Yu Cheng, Ilias Diakonikolas, and Rong Ge. High-dimensional robust mean estimation in nearly-linear time. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2755–2771. SIAM, 2019.

[9] Ilias Diakonikolas, Gautam Kamath, Daniel Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robust estimators in high-dimensions without the computational intractability. *SIAM Journal on Computing*, 48(2):742–864, 2019.

[10] Yihe Dong, Samuel Hopkins, and Jerry Li. Quantum entropy scoring for fast robust mean estimation and improved outlier detection. In *Advances in Neural Information Processing Systems*, pages 6067–6077, 2019.

[11] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Local model poisoning attacks to byzantine-robust federated learning. *arXiv preprint arXiv:1911.11815*, 2019.

[12] Shuhao Fu, Chulin Xie, Bo Li, and Qifeng Chen. Attack-resistant federated learning with residual-based reweighting. *arXiv preprint arXiv:1912.11464*, 2019.

[13] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[14] Brendan McMahan and Daniel Ramage. Federated learning: Collaborative machine learning without centralized training data. *Google Research Blog*, 3, 2017.

[15] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Rouault. The hidden vulnerability of distributed learning in byzantium. *arXiv preprint arXiv:1802.07927*, 2018.

[16] Krishna Pillutla, Sham M Kakade, and Zaid Harchaoui. Robust aggregation for federated learning. *arXiv preprint arXiv:1912.13445*, 2019.

[17] Jacob Steinhardt. *Robust learning: Information theory and algorithms*. PhD thesis, Stanford University, 2018.

[18] Gan Sun, Yang Cong, Jiahua Dong, Qiang Wang, and Ji Liu. Data poisoning attacks on federated machine learning. *arXiv preprint arXiv:2004.10020*, 2020.

[19] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[20] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Defending against saddle point attack in byzantine-robust distributed learning. In *International Conference on Machine Learning*, pages 7074–7084. PMLR, 2019.

[21] Dong Yin, Yudong Chen, Kannan Ramchandran, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. *arXiv preprint arXiv:1803.01498*, 2018.