An Experimental Study of Byzantine-Robust Aggregation Schemes in Federated Learning

Shenghui Li, *Student Member, IEEE*, Edith C.-H. Ngai, *Senior Member, IEEE*, and Thiemo Voigt, *Member, IEEE*

***Abstract*—Byzantine-robust federated learning aims at mitigating Byzantine failures during the federated training process, where malicious participants (known as Byzantine clients) may upload arbitrary local updates to the central server in order to degrade the performance of the global model. In recent years, several robust aggregation schemes have been proposed to defend against malicious updates from Byzantine clients and improve the robustness of federated learning. These solutions were claimed to be Byzantine-robust, under certain assumptions. Other than that, new attack strategies are emerging, striving to circumvent the defense schemes. However, there is a lack of systematical comparison and empirical study thereof. In this paper, we conduct an experimental study of Byzantine-robust aggregation schemes under different attacks using two popular algorithms in federated learning, *FedSGD* and *FedAvg*. We first survey existing Byzantine attack strategies, as well as Byzantine-robust aggregation schemes that aim to defend against Byzantine attacks. We also propose a new scheme, `ClippedClustering`, to enhance the robustness of a clustering-based scheme by automatically clipping the updates. Then we provide an experimental evaluation of eight aggregation schemes in the scenario of five different Byzantine attacks. Our experimental results show that these aggregation schemes sustain relatively high accuracy in some cases, but they are not effective in all cases. In particular, our proposed `ClippedClustering` successfully defends against most attacks under independent and identically distributed (IID) local datasets. However, when the local datasets are Non-IID, the performance of all the aggregation schemes significantly decreases. With Non-IID data, some of these aggregation schemes fail even in the complete absence of Byzantine clients. Based on our experimental study, we conclude that the robustness of all the aggregation schemes is limited, highlighting the need for new defense strategies, in particular for Non-IID datasets.**

***Index Terms*—Byzantine attacks, distributed learning, federated learning, neural networks, robustness.**

Manuscript received 1 April 2022; revised 18 November 2022; accepted 3 January 2023. Date of publication 16 January 2023; date of current version 13 November 2024. This research was supported in part by the RGC General Research Funds under Grants 17203320 and 17209822 from Hong Kong, in part by Swedish Research Council project under Grant 2017-04543, in part by the HKU-TCL joint research centre for artificial intelligence seed funding, and in part by the European Union's Horizon 2020 research and innovation programme under Grant 101015922. Recommended for acceptance by Special Issue On Trustable, Verifiable, and Auditable Federated Learning. *(Corresponding author: Edith C.-H. Ngai.)*

Shenghui Li is with the Department of Information Technology, Uppsala University, 752 36 Uppsala, Sweden (e-mail: shenghui.li@it.uu.se).

Edith C.-H. Ngai is with the Department of Electrical and Electronic Engineering, University of Hong Kong, Pok Fu Lam, Hong Kong (e-mail: chngai@eee.hku.hk).

Thiemo Voigt is with the Department of Electrical Engineering, Uppsala University, 752 36 Uppsala, Sweden, and also with the RISE, Research Institutes of Sweden, 111 21 Stockholm, Sweden (e-mail: thiemo.voigt@it.uu.se).

Digital Object Identifier 10.1109/TBDATA.2023.3237397

I. Introduction

FEDERATED learning (FL) [1], [2], [3] is a machine learning paradigm for distributed model training on decentralized data across a set of client devices (e.g., desktops, mobile phones, IoT devices). Specifically, FL repeatedly performs the following steps: the server broadcasts the current global model to client devices; the clients then perform one or several local steps of stochastic gradient descent (SGD) using private training sets, and send the updates back to the server; then the server generates a new global model by aggregating the local updates to enable the next round of training. This paradigm allows the client devices to perform most of the computation, without requiring any of the participants to reveal their private training data to a centralized entity or each other. In addition, benefiting from multiple steps of local updates before uploading local updates, FL improves communication efficiency compared to traditional distributed learning [4].

Despite the achievements of FL in terms of data privacy and communication efficiency, it also opens up the parameter updating process to manipulation by the clients, which brings serious security threats to model training [5]. An important class of security threats in this context is known as Byzantine failures [5], where some of the participants are not rigorously following the protocol, but upload arbitrary parameters to the central server, for example, due to faulty communication [6], or even worse, adversaries, where malicious attackers modify the update vectors to their desire and upload them to the server [7]. We use the term "Byzantine attack" to refer to the attacks where malicious attackers upload arbitrary updates to the server in order to degrade the overall performance of the global model in FL. In typical FL algorithms (e.g., *FedAvg*) [1], the server aggregates the uploaded updates by calculating their sample mean and adds the result to the global model. However, it is well-known that the result of such an aggregation scheme can be arbitrarily skewed even by a single Byzantine client [8]. The server thus requires Byzantine-robust solutions to defend against malicious clients.

In recent years, a number of Byzantine-robust techniques have been proposed [9]. They can be classified into three categories: redundancy-based schemes that assign each client redundant updates and use this redundancy to eliminate the effect of Byzantine failures [10], [11], [12], [13]; trust-based schemes that assume some of the clients or datasets are trusted for filtering and re-weighting the local model updates [14], [15], [16]; robust aggregation schemes that estimate the updates according to some robust aggregation algorithms [8], [17], [18], [19], [20], [21]. For the first category, redundancy-based schemes, in the worst case, require each node to compute $\Omega(M)$ times more updates, where M is the number of Byzantine clients [10]. This overhead is prohibitive in settings with large numbers of Byzantine clients. For the second category, the trusted clients/datasets are not always available to the server due to the concern of user data privacy.

Robust aggregation schemes, in contrast, aggregate the updates efficiently, without requiring trusted clients or datasets. However, typical schemes including `GeoMed` [18], `Krum` [17], `TrimmedMean` [8], `Median` [8] and `CC` [20], often come with limited guarantees of Byzantine robustness (e.g., only establishing convergence to a limit, or only guaranteeing that the output of the aggregation scheme has a positive inner product with the true gradient [17], [22]) and often require other strong assumptions, such as bounded absolute skewness [8]. More importantly, recent studies reveal the vulnerability of some schemes to new attacks. For instance, the A Little Is Enough (ALIE) attack can circumvent `TrimmedMean` and `Krum` by taking advantage of empirical variance between the updates of clients if such variance is high enough [23]. The Inner Product Manipulation (IPM) attack poses a significant threat to `Median` and `Krum` by manipulating the inner product between the true gradient and the robust aggregated gradients to be negative [24]. Other schemes, such as `AutoGM` [19] and `Clustering` [21], were proposed with only empirical evaluations.

These existing aggregation schemes are evaluated using different datasets, attack types, and hyper-parameters. There is a lack of empirical studies that compare different schemes of utilizing the same settings. Furthermore, the impact of data heterogeneity on robustness schemes is rarely evaluated as those schemes usually assume that all clients' local data are independent and identically distributed (IID). Therefore, there is a clear need for a comparative experimental study that offers in-depth insight into the performance of the existing Byzantine-robust schemes for FL.

To meet this need, we conduct an experimental study on the Byzantine attack and defense problem in FL based on two well-known algorithms, *FedSGD* and *FedAvg* [1], [23]. We first survey existing attack strategies and robust aggregation schemes in the literature. We further propose a new aggregation scheme `ClippedClustering` to address the weakness of an existing clustering-based scheme. Then we design experiments to evaluate the robustness of eight representative Byzantine-robust aggregation rules by applying five state-of-the-art attacking strategies. Our experimental results show that those aggregation rules sustain relatively high accuracy in some cases. However, they are not effective in all cases. Moreover, when the local datasets are not independent and identically distributed (Non-IID), the capability of all the aggregation rules decreases significantly. With Non-IID data, some of these aggregation rules fail even in the complete absence of Byzantine clients. Furthermore, our proposed scheme performs the best in most attack scenarios when the datasets are IID. From the evaluation, we conclude that existing aggregation rules are insufficient to meet the need for Byzantine robustness, highlighting the demand for new defense strategies in FL, especially with training on Non-IID datasets.

Our key contributions can be summarized as follows:

- We survey existing Byzantine attack strategies to compromise FL, as well as Byzantine-robust aggregation schemes that aim to defend against Byzantine attacks.
- Based on an existing clustering-based aggregation scheme, we further propose an enhanced scheme called `ClippedClustering`, by applying an automatical clipping technique to mitigate the effect of amplified local updates.
- We evaluate eight robust aggregation schemes (including the proposed `ClippedClustering`) under five representative Byzantine attack strategies. Our experimental results show that the aggregation schemes sustain high accuracy in some cases, but have limited success in other cases, especially in the presence of Non-IID data.

The rest of this paper is organized as follows: Section II first formulates the problem of FL and introduces two optimization algorithms. Then Section III introduces the threat models evaluated in this paper. Subsequently, representative robust aggregation schemes are presented in Section IV. Section V presents an adaptive attack to the proposed aggregation scheme, `ClippedClustering`. Section VI presents the experiments for robust aggregation schemes, from which some notable findings are uncovered. Finally, we review related work in Section VII and make some conclusions in Section VIII.

II. Federated Learning

In this section, we first formulate the optimization problem of FL. Then we introduce two popular algorithms for solving the FL problem, one is the classic distributed SGD optimization algorithm *FedSGD* and the other is the famous communication-efficient algorithm *FedAvg*.

A. Problem Formulation

In FL, multiple clients collaboratively learn a shared global model using their private datasets in a distributed way, assisted by the coordination of a central server. The goal is to find a parameter vector $\boldsymbol{w}$ that minimizes the following distributed optimization model

$$\min_{\boldsymbol{w}} F(\boldsymbol{w}) = \min_{\boldsymbol{w}} \frac{1}{K} \sum_{k\in[K]} F_k(\boldsymbol{w}), \tag{1}$$

where K is the total number of clients, the local objective $F_k(\cdot)$ can be defined as an empirical risk over local data, i.e., $F_k(w) = \frac{1}{n_k}\sum_{j\in[n_k]} \ell(\boldsymbol{w}; x_{k,j})$, where $\ell(\cdot;\cdot)$ is a user-specified loss function, $x_{k,j}$ is a training sample and n_k is the size of training dataset owned by client k.

A common assumption in FL is that local training datasets can be unbalanced, i.e., clients can have different numbers of training samples [1]. However, in this paper, we assume that data are balanced, i.e., $n_1 = n_2 = \cdots = n_K$ to align with most studies that specifically focus on Byzantine robustness [8], [18], [20]. We note that one can get rid of this assumption using the re-scaling trick proposed by Li et al. [25].

Algorithm 1: Optimization of Federated Learning.

```
    Input: $K, T, \eta_t, \boldsymbol{w}^0$
1:  for each global round $t \in [T]$ do
2:    for each client $k \in [K]$ in parallel do
3:      $\boldsymbol{w}_k^t \leftarrow \boldsymbol{w}^t$
4:      Option I (FedSGD):
5:        Sample mini-batch $\xi$ from local dataset
6:        $\boldsymbol{\Delta}_k^t \leftarrow \nabla F_k(\boldsymbol{w}_k^t, \xi)$
7:      Option II (FedAvg):
8:        for $E_l$ local rounds, do
9:          Sample mini-batch $\xi$ from local dataset
10:         $\boldsymbol{w}_k^t \leftarrow \boldsymbol{w}_k^t - \eta_t \nabla F_k(\boldsymbol{w}_k^t, \xi)$
11:       end for
12:       $\boldsymbol{\Delta}_k^t \leftarrow \boldsymbol{w}_k^t - \boldsymbol{w}^t$
13:     Sends $\Delta_k^t$ back to the server
14:   end for
15:   $\Delta^{t+1} \leftarrow AGG(\{\Delta_k^t\}_{k\in[K]})$
16:   Option I (FedSGD):
17:     $\boldsymbol{w}^{t+1} \leftarrow \boldsymbol{w}^t - \eta_t \Delta^{t+1}$
18:   Option II (FedAvg):
19:     $\boldsymbol{w}^{t+1} \leftarrow \boldsymbol{w}^t + \Delta^{t+1}$
20: end for
21: return $\boldsymbol{w}^T$
```

B. Optimizations of Federated Learning

We adopt the two most popular algorithms in Byzantine robust optimization literature to solve Problem (1), i.e., *FedSGD* and *FedAvg*.

1) FedSGD: Stochastic gradient descent (SGD) can be applied naively to the federated optimization problem (1) [1]. As summarized in Algorithm 1 with option I, each client calculates a single mini-batch gradient and uploads it to the server in parallel at each round of training. The server then aggregates the received gradients and updates the model parameters according to the aggregated gradients. Benefiting from mini-batch of stochastic gradient calculation, this approach is computationally efficient, but it still requires a very large number of communication rounds to produce good models [1], [26]. In this paper, we refer to this algorithm as *FedSGD*, also known as *sync-SGD* in some related work [23], [24].

2) FedAvg: A more communication-efficient framework for FL is *FedAvg* [1]. As summarized in Algorithm 1 with Option II, at each round of training, the server broadcasts its global model to each client. In parallel, the clients run multiple steps of SGD on their own loss functions and send the resulting model to the server. The server then updates its global model according to its aggregation rule and broadcasts the resulting global model to each client to enable the next round of training. Multiple rounds of interactions between the server and clients are required to obtain an accurate shared global model.

As one may see, general *FedAvg*-based algorithms usually randomly select a subset of clients to perform local training while the algorithm we adopt involves full participation of all clients at each round. This is because all of the aggregation schemes considered in this paper are based on an assumption that less than half of the updates for aggregation are malicious on each round. Selecting subsets at random violates this assumption with some probability, as it may select more malicious clients than benign ones by chance. Therefore, full participation is used in this paper.

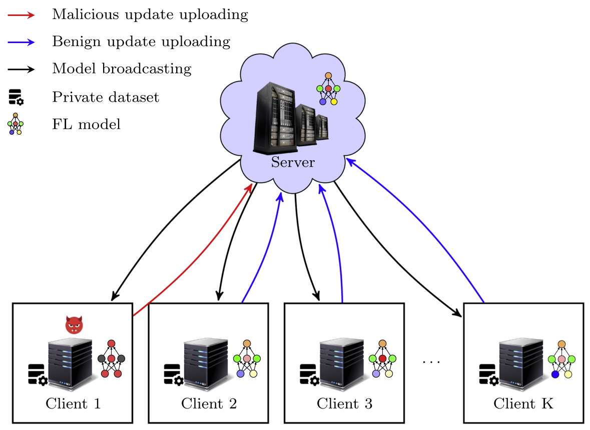

Fig. 1. Illustration of an FL system with Byzantine clients, where K clients collaboratively train a machine learning model using their local datasets. The coordinate server aggregates local updates and updates the model, then broadcasts the new model to the clients in each communication round. The attackers control some of the clients (e.g., Client 1) and send malicious updates to the server, while the honest clients compute and upload benign updates.

3) Update Aggregation: In both *FedSGD* and *FedAvg*, the server aggregates the received updates and uses the result of the aggregation to update the global model. A widely-used aggregation scheme is calculating the sample `Mean` of the uploaded updates, i.e.,

$$\Delta^{t+1} \leftarrow \frac{1}{K} \sum_{k\in[K]} \Delta_k^t. \tag{2}$$

However, `Mean` is vulnerable to malicious local updates. As the breakdown point of `Mean` is $1/K$ [27], which means that even if only one of the clients is malicious, the resulting global model can significantly deviate from the original `Mean`. In Section IV we will cover robust aggregations that aim to defend against malicious updates.

III. Threat Models

In this section, we describe the five threat models of Byzantine attacks we evaluate in this paper.

In terms of Byzantine attacks, most existing literature for distributed learning and federated learning focuses on convergence prevention [7], [19], [21], [24]. As illustrated in Fig. 1, the attackers (known as Byzantine clients) may upload arbitrary parameters to the server in order to degrade the performance of the global model. Thus, Line 6 and 12 of Algorithm 1 are replaced by the following

$$\boldsymbol{\Delta}_k^t \leftarrow \begin{cases} \star & \text{if } k\text{-th client is Byzantine,} \\ \nabla F_k(\boldsymbol{w}_k^t, \xi) & \text{if } k\text{-th client is benign (for } FedSGD\text{),} \\ \boldsymbol{w}_k^t - \boldsymbol{w}^t & \text{if } k\text{-th client is benign (for } FedAvg\text{),} \end{cases} \tag{3}$$

where $\star$ represents arbitrary values.

In this paper, we follow the assumption that the majority of the clients are benign [19], [20], which means we have $\frac{M}{K} < 0.5$, where M is the number of Byzantine clients. We examine five typical attacks in our threat models.

A. Noise

A straightforward attack is to sample some random noise from a distribution (e.g., Gaussian distribution) and add it to the updates before uploading [19], [28]. For simplicity sake, the mean and variance of the noise are both 0.1 in our experiment.

B. A Little is Enough (ALIE)

In contrary to the random Noise attack, the attackers may modify the noise carefully to pretend to be benign and fool the aggregation rules. A Little is Enough (ALIE) [23] assumes that the benign updates are expressed by a normal distribution. The attackers therefore immediately take advantage of the high empirical variance between the updates of clients and upload a noise in a range without being detected.

For each coordinate $i \in [d]$, the attackers calculate mean (μ_i) and std (δ_i) over benign updates, and set corrupted updates Δ_i to values in the range $(\mu_i - z^{\max}\delta_i, \mu_i + z^{\max}\delta_i)$, where $z^{\max}$ ranges from 0 to 1, and is typically obtained from the Cumulative Standard Normal Function [23].

C. Inner Product Manipulation (IPM)

The Inner Product Manipulation (IPM) attack [24] seeks for the negative inner product between the true mean of the updates and the output of the aggregation schemes so that at least the loss will not descend. Assuming that the attackers know the mean of benign updates, a specific way to perform an IPM attack is

$$\boldsymbol{\Delta}_1^t = \cdots = \Delta_M^t = -\frac{\epsilon}{K-M}\sum_{i=M+1}^{K}\Delta_i^t, \tag{4}$$

where we assume that the first M clients are malicious, ϵ is a positive coefficient controlling the magnitude of malicious updates. Then the `Mean` becomes

$$\frac{1}{K}\sum_{k\in[K]}\Delta_k^t = \frac{K-M(1+\epsilon)}{K(K-M)}\sum_{i=M+1}^{K}\Delta_i^t. \tag{5}$$

Note that when $\epsilon < \frac{K}{M} - 1$, IPM does not change the direction of the average over benign updates but only decreases its magnitude, because we have

$$\frac{K-M(1+\epsilon)}{K(K-M)} > 0,$$

the optimization thus can still converge using `Mean` as an aggregation scheme. However, as we will show in Section VI, such an attack can circumvent the defense of several aggregation schemes and inverse the direction of updates, which heavily damages the global model. On the contrary, when $\epsilon > \frac{K}{M} - 1$, the sign of `Mean` is reversed, indicating that the loss will increase if the model is updated using the `Mean`. In our experiment, we examine both cases by letting $\epsilon = 0.5$ and $\epsilon = 100$, respectively.

D. Sign Flipping (SF)

Different from IPM, the Sign Flipping (SF) attackers do not need to know the updates from other clients and simply flip the signs of the gradient [13], [20], which means that the attackers strive to maximize the loss via gradient ascent instead of gradient descent. Specifically, in *FedSGD*, the clients upload the negative gradients; in *FedAvg*, the flipping is applied at every local updating step.

E. Label Flipping (LF)

The aforementioned attacks assume that the attackers have full access to the training process so that they can modify the updates immediately. However, full access may be limited as the training APIs are not always open. Correspondingly, the attackers can also change the training dataset instead of the update parameters [29]. The Label Flipping (LF) attack simply flips the label of each training sample [7]. Specifically, a label l is flipped as $L - l - 1$, where L is the number of classes in the classification problem and $l = 0, 1, \ldots, L-1$.

IV. Aggregation Schemes for Evaluation

In this section, we survey existing robust aggregation schemes, which represent state-of-the-art methods in the literature. Then, we propose a new scheme ClippedClustering, which addresses the weakness of the clustering-based scheme. Other than that, we provide a taxonomy of the eight aggregation schemes evaluated in our experiments.

All aggregation schemes considered in this paper are working on each round separately. For the sake of readability, we will omit the notation of the round ^{t} in the following sections.

A. Krum

`Krum` [17] strives to find one of the local model updates that is closest to another $K - M - 2$ ones with respect to squared euclidean distance, which can be expressed by

$$Krum := \{\boldsymbol{\Delta}_i | i = \arg\min_{i\in[K]} \sum_{i\to j} \|\boldsymbol{\Delta}_i - \boldsymbol{\Delta}_j\|^2\},$$

where $i \to j$ is the indices of the $K - M - 2$ nearest neighbours of $\boldsymbol{\Delta}_i$ measured by squared euclidean distance, recall that K is the number of clients in total, and M is the number of malicious clients.

Under the *FedSGD* framework, `Krum` was proven to converge with an important assumption that $c_1\sigma < \|g\|$, where c_1 is a constant factor depending on the number of malicious clients and the dimension of model parameters, σ is the maximal variance of the updates and $\|g\|$ is the expectation of updates.

B. Geomed

The Geometric Median (`GeoMed`) [18], [30] scheme aims to find a vector that minimizes the sum of its euclidean distances to all the update vectors

$$GeoMed := \arg\min_{\boldsymbol{z}} \sum_{k\in[K]} \|\boldsymbol{z} - \boldsymbol{\Delta}_k\|. \tag{6}$$

Although there is no closed-form solution to the GeoMed problem, a $(1+\epsilon)$-approximate solution can be computed in nearly linear time [31].

Similar to Krum, GeoMed was also proven to converge under the *FedSGD* framework, with the assumption that $c_2\sigma < \|g\|$, where c_2 is a another constant factor that differs from c_1.

C. Autogm

Auto-weighted Geometric Median (AutoGM) [19] is a generalized version of GeoMed. AutoGM aggregates the updates by solving the following problem

$$AutoGM := \arg\min_{\boldsymbol{z}} \sum_{k\in[K]} \alpha_k \|\boldsymbol{z} - \boldsymbol{\Delta}_k\| + \frac{\lambda}{2}\|\boldsymbol{\alpha}\|^2,$$
$$\text{s.t.} \quad \boldsymbol{\alpha} \in \mathbb{R}_+^K, \mathbf{1}^\top \boldsymbol{\alpha} = 1, \tag{7}$$

where λ is a user-specified hyper-parameter that controls the smoothness of $\boldsymbol{\alpha}$.

The key idea of optimizing AutoGM is to divide the problem into two parts, i.e., one subproblem for estimating the weighted GeoMed, and the other subproblem for weighting the importance of each point. Then, we can minimize the objective iteratively with respect to one variable each time while fixing the other one [19].

D. Median

Median [8] is defined as the coordinate-wise median of the given set of updates, i.e.,

$$med := Median(\{\boldsymbol{\Delta}_k : k \in [K]\}),$$

where the i-th coordinate $med_i = median(\{\boldsymbol{\Delta}_k^i : k \in [K]\})$, and $median$ is the usual (one-dimensional) median.

When using the *FedSGD* framework, the robustness of the Median scheme is based on the assumptions that the gradient of the loss function has bounded variance, and each coordinate of the gradient has coordinate-wise bounded absolute skewness [8].

E. Trimmedmean

The TrimmedMean [8] aggregation scheme computes the coordinate-wise trimmed average of the model updates, which can be expressed by

$$trmean := TrimmedMean(\{\boldsymbol{\Delta}_k : k \in [K]\}),$$

where the i-th coordinate $trmean_i = \frac{1}{(1-2\beta)m}\sum_{x\in U_k} x$, and U_k is a subset obtained by removing the largest and smallest β fraction of its elements.

In addition to the aforementioned assumptions for Median, the robustness of TrimmedMean relies on one stronger assumption that all the moments of the derivatives of the loss function are bounded [8].

F. Centered Clipping (CC)

Centered Clipping (CC) [20] iteratively clips the updates around the center while updating the center accordingly. For $l \geq 0$, CC computes

$$\Delta_{l+1} \leftarrow \Delta_l + \frac{1}{K}\sum_{k\in[K]} (\Delta_i - \Delta_l) \min\left(1, \frac{\tau_1}{\|\Delta_i - \Delta_l\|}\right), \tag{8}$$

where Δ_0 is assigned with the aggregated updates in the previous round.

Karimireddy et al. [20] proofed the robustness of the CC scheme when the variance of updates is bounded and $\frac{K}{M} \leq 0.15$.

G. Clustering

This Clustering aggregation scheme [21], [32] first calculates the pairwise cosine distances between their parameter updates, i.e.,

$$\alpha_{i,j} := 1 - \frac{\langle \boldsymbol{\Delta}_i, \boldsymbol{\Delta}_j \rangle}{\|\boldsymbol{\Delta}_i\|\|\boldsymbol{\Delta}_j\|}, \tag{9}$$

then it separates the client population into two groups based on the cosine similarities using agglomerative clustering with average linkage. Finally, it aggregates the updates in the largest group using Mean.

Despite the lack of theoretical guarantee of robustness, Clustering achieves superior robustness in some cases, as we will show in Section VI. However, an obvious drawback of clustering using cosine similarities is that it only considers the relative directions, ignoring the magnitude of each vector. The attackers thus can fool the clustering scheme by simply amplifying the updates without changing their directions. As a consequence, the resulting updates added to the parameters will make the model jump over the minima and prevent the convergence of the optimization without being detected.

H. Clippedclustering

We enhance the robustness of the aforementioned Clustering aggregation scheme by performing a clipping on all the updates before clustering, i.e.,

$$\Delta_k \leftarrow \Delta_k \min(1, \frac{\tau}{\|\Delta_k\|}). \tag{10}$$

Here, τ is a clipping value hyper-parameter that is determined by the server. Note that this clipping scheme is the so-called clip by norm, not clip by value, where individual values of the update vectors are clipped if they go beyond a pre-set value. In clip by norm, the entire update is scaled if the norm of the update exceeds the threshold τ. Thus we place a maximum on the magnitude of each vector that can be taken during training, preventing the attackers from amplifying the updates in the same direction. If the norm of update is below the threshold τ, the update is unaffected.

Inspired by [33], we design an automatic clipping strategy to defend against potential amplified malicious updates that the naive cosine similarity-based clustering scheme cannot handle well. Specifically, we set the clipping value hyper-parameter based on the statistics of the historical norms of the updates uploaded during training, i.e., we save the update norms up to current iteration and automatically set τ using the 50-th percentile value (i.e., the median) of the history.

TABLE I
SUMMARY OF THE ROBUST AGGREGATION SCHEMES AND THEIR MAIN DEFENSE MECHANISMS

Defense	*Euclidean distance*	*Mean*	*Median*	*Cosine similarity*	*Clipping*
`Krum` [17]	✓				
`GeoMed` [18]	✓		✓		
`AutoGM` [19]	✓		✓		
`Median` [8]			✓		
`TrimmedMean` [8]		✓			
`CC` [20]		✓			✓
`Clustering` [21], [32]		✓		✓	
`ClippedClustering` (ours)		✓		✓	✓

There are two reasons of choosing the median update norm as the clipping threshold: First, prior work [34] has demonstrated that adaptive clipping to median norm works well across a range of general federated learning tasks without the need to tune any clipping hyper-parameter. Second, the median itself is a robust statistical measure of central tendency. As we assume that the majority of the clients are benign, malicious clients are unable to control the median norm even if they have full knowledge of all the updates.

I. Taxonomy

Table I shows a taxonomy of the eight aggregation schemes, where `Krum`, `GeoMed`, and `AutoGM` are typical euclidean distance-based schemes, i.e., they are all designed to find a vector closest to the updates measured by euclidean distance. Among them, `GeoMed` and `AutoGM` are both based on geometric median. The `Median` scheme simply computes the coordinate-wise median instead of geometric median. `TrimmedMean`, `CC`, `Clustering` and `ClippedClustering` are all categorized as mean-based schemes as they eventually compute the mean, although they also utilize other mechanisms. `Clustering` and `ClippedClustering` both perform clustering based on cosine similarity while `ClippedClustering` clips the updates before clustering.

V. ADAPTIVE ATTACK ON `ClippedClustering`

In this section, we design a strong adaptive attack on `ClippedClustering`. We assume that the attacker has full knowledge of the system, including the aggregation scheme and all the updates from benign clients.

Given the fact that the updates are clipped by the server if their magnitudes are beyond the threshold, state-of-the-art approach [35] that aims to maximize euclidean distance between the aggregated vector and the true update is no longer applicable. Instead, the attack would be more effective if it can change the direction of update without being excluded by the aggregation scheme.

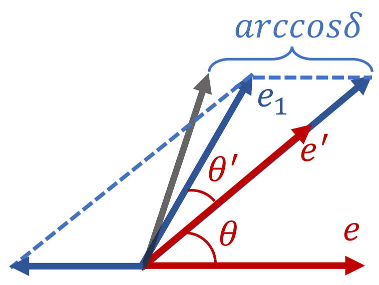

Fig. 2. Demonstration of the solution to (11). The solution of $\boldsymbol{e}_1$ can be expressed as a linear combination of $\boldsymbol{e}$ and $\boldsymbol{e}'$.

The idea of our attack is to make sure that all the malicious updates stay in the largest cluster, while deviating from the correct direction as possible. Since the clustering method and linkage function are known to us, we can perform the same clustering process with the benign updates only. Then we can carefully design malicious updates to make it close enough to the largest cluster, without breaking the existing structure. Specifically, we compute the average cosine similarity of the two benign clusters, denoted by δ, and use it as the bound to design malicious updates.

For simplification, we let $\boldsymbol{\Delta}_1 = \cdots = \boldsymbol{\Delta}_M$, and denote by $\boldsymbol{e}_i = \frac{\boldsymbol{\Delta}_i}{\|\boldsymbol{\Delta}_i\|}$ the unit vector whose direction is the same as $\boldsymbol{\Delta}_i$. The problem becomes

$$\begin{aligned} \min_{\boldsymbol{e}_1} \quad & \langle \boldsymbol{e}_1, \boldsymbol{e} \rangle, \\ \text{s.t.} \quad & \langle \boldsymbol{e}_1, \boldsymbol{e}' \rangle > \delta, \end{aligned} \tag{11}$$

where $\boldsymbol{e}$ is the unit vector of $\boldsymbol{\Delta}$, and $\boldsymbol{e}'$ is the unit vector of the center of the largest benign cluster. The constraint ensures that the malicious group is included by the largest cluster when we perform hierarchical clustering based on average linkage.

Minimizing the problem is equivalent to maximizing the angle between $\boldsymbol{e}_1$ and $\boldsymbol{e}$, with the constraint that the angle between $\boldsymbol{e}_1$ and $\boldsymbol{e}'$ should be smaller than $\arccos \delta$, which is demonstrated in Fig. 2. Now we are able to solve the problem using $\{\boldsymbol{e}, \boldsymbol{e}'\}$ a basis, i.e., let

$$\theta = \arccos \langle \boldsymbol{e}, \boldsymbol{e}' \rangle, \tag{12}$$

$$\theta' = \arccos \delta - \epsilon, \tag{13}$$

where $\epsilon > 0$ is a small enough number. The solution to (11) is expressed as

$$\boldsymbol{e}_1 = \left(\cos(\theta + \theta') - \frac{\sin(\theta + \theta')}{\tan \theta} \right) \boldsymbol{e} + \left(\cos \theta' + \frac{\sin \theta'}{\tan \theta} \right) \boldsymbol{e}'. \tag{14}$$

Once we obtain the unit vector of malicious updates, we can scale the magnitude by the clipping threshold τ, i.e.,

$$\boldsymbol{\Delta}_1 = \cdots = \boldsymbol{\Delta}_M = \tau \boldsymbol{e}_1. \tag{15}$$

We note that such an attack is applicable to agglomerative clustering with average linkage. For complete linkage, one can simply replace δ with the minimum cosine similarity of the two benign clusters and solve the problem in the same manner. Other linkage functions are omitted due to space limitations.

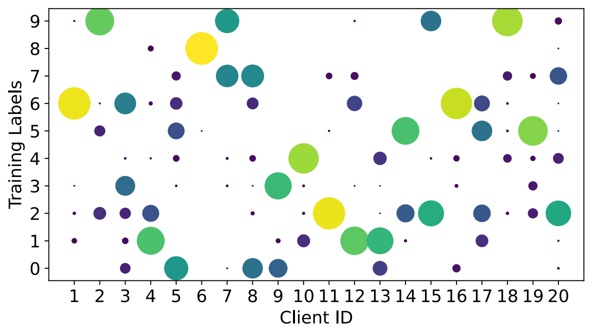

Fig. 3. Visualization of statistical heterogeneity among clients on our Non-IID partition of CIFAR-10 dataset, where the x-axis indicates client IDs, the y-axis indicates class labels, and the size of scattered points indicates the number of training samples for a label available to that client.

VI. Evaluation

In this section, we design experiments to evaluate the aforementioned robust aggregation schemes under different attacks and show the experimental results.

A. Experimental Setup

We simulate an FL system with the server and 20 clients, targeting at image classification tasks on CIFAR-10 [36] and MNIST [37] datasets with both IID and Non-IID partitions, where five of the clients are Byzantine clients by default. For the IID partitioning, we randomly split the training set into 20 subsets and allocate them to the 20 clients. For the Non-IID partition, we follow prior work [38], [39] and model the Non-IID data distributions with a Dirichlet distribution $\boldsymbol{p}_l \sim Dir_K(\alpha)$. Then we allocate a $\boldsymbol{p}_{l,k}$ proportion of the training sample of class l to client k, in which a smaller α indicates a stronger Non-IID data partition. We let $\alpha = 0.1$ for all Non-IID settings. Fig. 3 visualizes the resulting statistical heterogeneity of labels on CIFAR-10. Such a partition is strongly Non-IID as one can see that some of the classes are completely missing for each client.

For CIFAR-10, we choose a lightweight Compact Convolutional Transformers (CCT) network [40], as such a small yet effective model has more potential to overcome the on-board resource limitation of FL devices [41]. For MNIST, we train a simple two-layer perceptron using ReLu as activation function. We train the model for 6000 and 600 communication rounds for *FedSGD* and *FedAvg*, respectively. By default, we set the batch size to 128 and 64 for MNIST and CIFAR-10, respectively. As suggested by Karimireddy et al. [20], we decay the learning rate during training to improve convergence, i.e., for *FedSGD*, we let

$$\eta_t \leftarrow \begin{cases} 0.1 & \text{if } t \leq 2000, \\ 0.05 & \text{if } 2000 < t \leq 5000, \\ 0.025 & \text{else;} \end{cases} \tag{16}$$

for *FedAvg*, we let

$$\eta_t \leftarrow \begin{cases} 0.1 & \text{if } t \leq 200, \\ 0.05 & \text{if } 200 < t \leq 500, \\ 0.025 & \text{else.} \end{cases} \tag{17}$$

For *FedAvg*, we apply 50 SGD steps before uploading the updates to the server (i.e., $E_l = 50$).

B. Impact on the `Mean` Scheme

We first demonstrate the impact of attacks on conventional *FedSGD* and *FedAvg* using the `Mean` scheme to aggregate updates by plotting test accuracy versus the number of communication rounds in Fig. 4. At first glance, when the datasets are IID, *FedAvg* takes fewer communication rounds than *FedSGD* to converge when there is no attack, benefiting from multiple steps of local training. However, when the datasets are Non-IID, the performance of *FedAvg* significantly decreases while *FedSGD* still maintains relatively high accuracy. We note that this is a well-known drawback of *FedAvg* [42], [43]. Recall that some of the attacks (e.g., IPM with small ϵ [24]) are particularly designed to break the line of robust defense, which means that they may not bring much damage to the `Mean` scheme. For instance, Fig. 4 shows that IPM ($\epsilon = 0.5$) produces less damage to `Mean` compared to the other attacks, as the malicious updates do not change the direction of the average update but only decrease its magnitude (see Section III-C). Furthermore, Noise and IPM ($\epsilon = 100$) eventually damage the models under the four settings by decreasing the test accuracy to around 10% (no better than random guess). This is because they both make large changes to the updates, and `Mean` could be easily biased by large changes [19].

Unsurprisingly, ALIE attack is ineffective to MNIST dataset, while it makes a large impact on CIFAR-10 with *FedSGD*. This is because the amount of noise for ALIE is determined by the empirical variance of benign updates. As it is already known that MLP on MNIST usually has a lower variance than CNN on CIFAR-10 [44].

C. Impact on Robust Aggregation Schemes

Table II shows the overall comparison of the robust aggregation schemes in *FedSGD* and *FedAvg* with respect to test accuracy on both IID and Non-IID partitioned datasets.

First of all, with the absence of malicious clients, `Krum`, `GeoMed`, `AutoGM`, and `Median` achieve relatively lower accuracy compared to other schemes. Especially, such degradation is more significant with Non-IID data. For instance, the accuracy almost reduces to random guessing, 10%, for CIFAR-10 with *FedSGD*, while `Mean`, `TrimmedMean`, `Clustering`, and `ClippedClustering` sustain the accuracy at around 80%. This suggests that `Krum`, `GeoMed`, `AutoGM`, and `Median` should be used with caution for highly Non-IID data.

Euclidean-based schemes (i.e., `Krum`, `GeoMed`, and `AutoGM`) reach lower accuracy compared to the other schemes in the complete absence of attackers, especially when the datasets are Non-IID. Surprisingly, their performance of *FedSGD* with Non-IID CIFAR-10 data is not much better than random guessing. This might be because they all tend to select a single update that is closest to all or part of the others measured by euclidean distance, which however is a poor estimation of the overall tendency in data heterogeneity situations. Furthermore, they all show similar robustness in both *FedSGD* and *FedAvg*. E.g., with

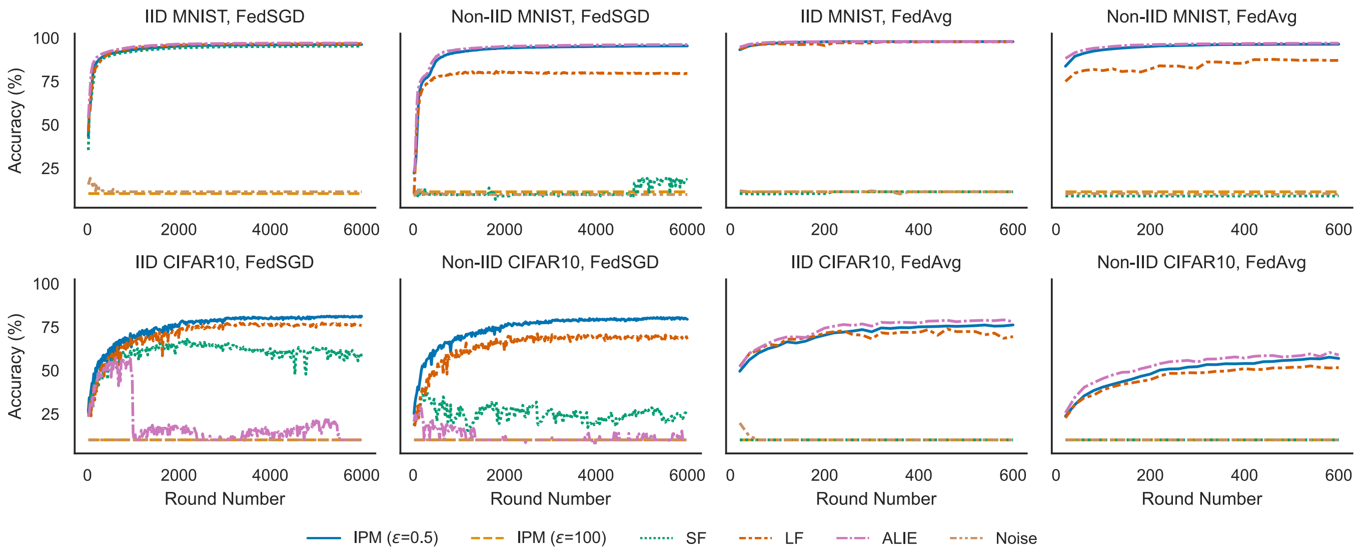

Fig. 4. Impact of malicious attacks on the `Mean` aggregation scheme. Overall, compared with using IID datasets, training with Non-IID datasets is more vulnerable to attacks.

IID data partition, they all handle Noise and IPM ($\epsilon = 100$) well while somehow struggling with the other attacks. This is not surprising as those euclidean-based schemes are essentially designed to defend against large changes of updates. On the other hand, small-scale attacks such as IPM ($\epsilon = 0.5$) challenges them as the malicious updates are close to the benign ones when measured by euclidean distance.

Two other classic robust aggregation schemes, `Median`, and `TrimmedMean` show similar robustness in most cases. For instance, when tested with IID MNIST, they successfully defend against most attacks with insignificant accuracy degradation. However, for Non-IID MNIST, both suffer from IPM attacks. A similar phenomenon occurs with CIFAR-10.

`CC` and `Clustering` both fail in all IPM ($\epsilon = 100$) and Noise attacks, because these attacks are of large magnitude. In these cases, `ClippedClustering` significantly enhances the robustness of `Clustering`, benefiting from the adaptive clipping mechanism. In the experiments, our proposed `Clipped-Clustering` successfully defends against more types of attacks than the other schemes, and achieves the highest test accuracy. However, it is defeated by ALIE when we train the model using *FedSGD* with CIFAR-10. Similar to other schemes, `ClippedClustering` is also less robust in Non-IID data scenarios. We note that Non-IIDness is a well-known challenge to FL especially when it comes to robustness [39], as it becomes hard to induce a consensus model for the benign clients if their data distributions are significantly different. Thus the malicious clients can damage the global model more easily.

Another important observation from our experiments is that ALIE makes almost no accuracy degradation when we train models using MNIST dataset, while it successfully circumvents almost all aggregation schemes of *FedSGD* with CIFAR-10 and keeps the test accuracy below 20%. The models suffer much less damage from ALIE when trained with *FedAvg*. Moreover, even the `Mean` scheme could handle it well. We thus infer that multiple steps of SGD updates may result in lower variance compared to single-step updates when the datasets are IID.

D. Impact of Fraction of Malicious Clients

To study the impact of the number of malicious clients, we perform an experiment using *FedAvg* with IID partitioned CIFAR-10 under six types of attacks. The results are shown in Fig. 5. Overall, the attacks make varying degrees of impact on the aggregation schemes as the fraction of Byzantine clients increases. Interestingly, the tendency of `Krum`, `GeoMed`, and `AutoGM` are similar, except for the impact of SL attack. Particularly, these schemes all suffer from even a small fraction of IPM ($\epsilon = 0.5$) attackers, i.e., the accuracy is reduced by 10% with the presence of a single Byzantine client (when 5% of the clients are malicious).

`CC` and `Clustering` are both vulnerable to Noise, SF and IPM ($\epsilon = 100$) attacks even with the presence of only 5% Byzantine clients. `Median`, `TrimmedMean`, and `Clipped-Clustering` are more robust to attacks especially when the fraction is low. Surprisingly, `ClippedClustering` tends to be affected by LF attacks as the fraction increases. Note that Fig. 5 only shows the performance of *FedAvg* with IID data. When the local datasets are Non-IID, all schemes show considerably less tolerance to Byzantine attacks.

E. Impact of Batch Size

In the previous experiments, the batch size per client is relatively small (i.e., 128 and 64 for MNIST and CIFAR-10, respectively), which leads to a large variance among benign updates, thus making the attacks more challenging [20]. Taking CIFAR-10 as an example, we investigate the effect of batch size on the robustness of aggregation schemes by varying batch size in $\{64, 128, 512, 2500\}$.

TABLE II
COMPARING STATE-OF-THE-ART ROBUST AGGREGATION SCHEMES AND OUR `ClippedClustering` UNDER VARIOUS ATTACKS, WHERE 25% OF THE CLIENTS ARE MALICIOUS. A SEMI-TRANSPARENT VALUE INDICATES THAT THE CORRESPONDING ACCURACY IS LOWER THAN 15% (NOT MUCH BETTER THAN RANDOM GUESSING). WE BOLD THE NUMBERS WITH THE HIGHEST ACCURACY. WHEN INTEGRATED WITH ROBUST AGGREGATION SCHEMES, *FEDAVG* IS MORE ROBUST THAN *FEDSGD*. ON THE OTHER HAND, ALL THE ATTACKS BECOME MORE EFFECTIVE IN NON-IID SCENARIOS

Dataset	*Algorithm*	Defense	IID data							Non-IID data						
			No Attack	IPM (ϵ=0.5)	IPM (ϵ=100)	SF	LF	ALIE	Noise	No Attack	IPM (ϵ=0.5)	IPM (ϵ=100)	SF	LF	ALIE	Noise
MNIST	*FedSGD*	`Mean`	**97.13**	96.16	10.32	95.19	96.70	96.98	11.35	96.70	**95.35**	11.35	18.18	79.45	**96.14**	9.80
		`Krum`	96.04	88.64	96.15	**95.51**	96.15	95.96	96.15	78.57	52.28	81.54	66.37	82.07	93.82	81.54
		`GeoMed`	95.97	77.12	95.99	95.47	96.12	95.96	**96.19**	57.59	47.57	55.94	49.07	58.16	93.74	50.08
		`AutoGM`	95.97	77.12	96.15	95.47	96.08	95.96	96.19	58.51	46.86	55.75	48.76	59.99	93.74	50.23
		`Median`	96.07	90.65	89.10	94.78	95.27	96.11	94.68	80.10	15.62	19.52	52.39	72.12	92.90	79.96
		`TrimmedMean`	97.13	93.92	88.97	95.02	95.24	96.27	93.63	96.70	21.52	12.60	76.85	77.19	92.53	**89.43**
		`CC`	97.13	96.16	0.79	95.20	96.70	96.98	10.09	96.69	95.35	11.35	82.70	79.46	96.14	9.58
		`Clustering`	97.11	96.07	10.32	95.26	96.72	**97.00**	11.35	**96.79**	95.31	11.35	9.74	80.95	95.88	10.32
		`ClippedClustering` (ours)	97.04	**96.90**	**96.98**	95.50	**96.80**	96.93	92.71	95.91	85.19	**84.82**	**88.71**	**85.28**	95.56	10.07
MNIST	*FedAvg*	`Mean`	98.13	97.83	11.35	11.35	97.66	97.87	11.35	96.87	**96.36**	11.35	8.92	87.02	**96.91**	9.80
		`Krum`	95.23	93.61	95.35	95.35	95.35	97.86	95.35	73.06	42.03	74.81	74.81	58.06	96.44	74.81
		`GeoMed`	95.24	93.17	93.77	95.22	95.13	97.86	95.36	46.64	55.75	26.09	48.90	48.15	96.44	46.68
		`AutoGM`	95.24	85.45	95.21	95.17	95.13	97.86	95.48	46.69	55.75	48.23	48.94	49.09	96.44	46.68
		`Median`	98.08	96.72	96.09	97.78	97.80	97.82	**98.02**	89.85	58.06	32.16	86.97	91.09	96.14	92.43
		`TrimmedMean`	98.13	97.64	95.73	**97.83**	97.78	97.85	98.01	96.87	61.93	55.51	91.77	93.03	95.26	**94.98**
		`CC`	98.10	97.84	9.82	11.35	97.67	**97.89**	11.35	**96.88**	96.35	11.35	9.74	87.00	96.91	9.80
		`Clustering`	98.11	97.87	11.35	9.74	97.81	97.84	10.28	96.85	96.04	11.35	9.58	87.70	96.86	9.80
		`ClippedClustering` (ours)	**98.15**	**97.93**	**97.90**	**97.83**	**97.90**	97.86	97.99	96.48	96.22	**81.08**	**93.80**	**94.74**	96.83	93.56
CIFAR-10	*FedSGD*	`Mean`	**82.35**	**81.22**	10.00	57.62	75.86	10.00	10.00	**81.15**	79.44	10.00	26.48	**67.98**	10.00	10.00
		`Krum`	67.47	58.17	50.60	65.24	67.90	10.00	60.26	10.00	10.00	10.00	17.54	10.00	10.00	10.00
		`GeoMed`	63.25	59.40	36.79	65.29	66.52	26.21	22.55	17.09	10.00	16.21	18.98	10.00	10.00	10.00
		`AutoGM`	66.16	42.70	66.69	65.16	66.55	10.00	48.06	10.00	10.00	10.00	18.44	18.18	10.00	18.97
		`Median`	78.78	44.42	33.31	68.76	59.60	10.00	10.00	10.00	10.00	10.00	10.00	10.00	10.00	10.08
		`TrimmedMean`	82.06	51.30	34.97	68.49	54.63	10.00	16.33	80.98	10.00	10.00	10.00	10.00	10.00	10.00
		`CC`	82.21	81.20	8.76	15.76	23.08	38.81	10.00	53.44	**80.23**	10.00	10.00	67.79	10.00	10.00
		`Clustering`	81.80	79.64	10.00	61.89	76.37	**77.48**	10.00	79.37	77.47	10.00	24.77	66.35	10.00	10.00
		`ClippedClustering` (ours)	81.58	80.34	**81.32**	**77.94**	**79.18**	10.00	**73.89**	76.83	69.53	**20.06**	**52.37**	64.48	**16.84**	**20.28**
CIFAR-10	*FedAvg*	`Mean`	80.19	76.18	10.00	10.00	69.45	78.10	10.00	63.35	56.89	10.00	10.00	51.60	58.89	10.00
		`Krum`	72.04	54.53	72.32	70.41	72.22	75.55	71.92	27.09	10.00	21.47	23.26	26.98	47.36	21.78
		`GeoMed`	72.93	49.76	69.80	10.00	73.72	75.97	71.92	27.95	10.00	10.00	10.00	24.19	48.45	26.38
		`AutoGM`	73.13	44.64	69.11	71.40	72.88	75.39	71.13	30.12	10.00	21.68	26.21	20.30	48.92	29.17
		`Median`	80.74	66.39	65.01	73.86	**77.94**	76.48	75.43	57.62	20.60	16.96	**48.60**	45.22	49.49	42.58
		`TrimmedMean`	**80.76**	73.42	61.13	66.52	76.42	77.66	73.08	62.98	30.92	18.61	47.95	47.92	53.03	39.67
		`CC`	80.01	77.77	10.00	10.00	69.80	**78.53**	24.71	**63.39**	56.97	10.00	10.00	**52.41**	**59.29**	17.24
		`Clustering`	80.15	77.74	10.00	10.00	75.61	78.12	10.00	61.45	**57.37**	10.00	10.23	49.43	58.66	10.00
		`ClippedClustering` (ours)	79.71	**78.80**	**78.44**	**78.01**	63.07	77.73	**78.82**	62.52	53.31	**31.24**	24.31	50.77	56.51	**48.55**

Fig. 6 shows the performance of *FedSGD* with IID partition under ALIE attacks. `Mean`, `ClippedClustering`, `AutoGM`, `TrimmedMean` and `CC` become more robust as the batch size increases. This is because the variance tends to be lower as we increase the batch size. Particularly, when the batch size is 2500, the clients use all their training data for each step of training, stochastic gradient then becomes population gradient (the optimization becomes full-batch gradient descent). However, except `ClippedClustering`, all the other aggregation schemes still fail to achieve acceptable test accuracy. Although a large batch size is more favorable for robustness purposes as indicated by this experiment, it is not the desired solution as it brings a large computation burden to local training.

F. Impact of Adaptive Attack

We examine `ClippedClustering` with the adaptive attack described in Section V. The result is shown in Fig. 7. For the sake of visualization, we clamp the loss into the range $[0, 10^5]$. For *FedSGD* with IID data (shown in the first column), `ClippedClustering` tolerates up to 15% clients attacked with little performance degradation. When more than 25% of clients are malicious, the loss curves fluctuate or even tend to increase. For *FedSGD* with Non-IID data (shown in the second column), the models diverge as long as there are 15% clients are malicious.

As for *FedAvg* with IID data, `ClippedClustering` shows higher tolerance than *FedSGD*. Specifically, the model for IID MNIST successfully converges with all the different fractions of Byzantine clients. However, as shown by the fourth column of Fig. 7, it is still much less robust when it comes to Non-IID data, where the model does not tend to converge when more than 15% of clients are malicious.

Noticeably, we observe that preventing convergence is not always sufficient to degrade the accuracy. For example, for

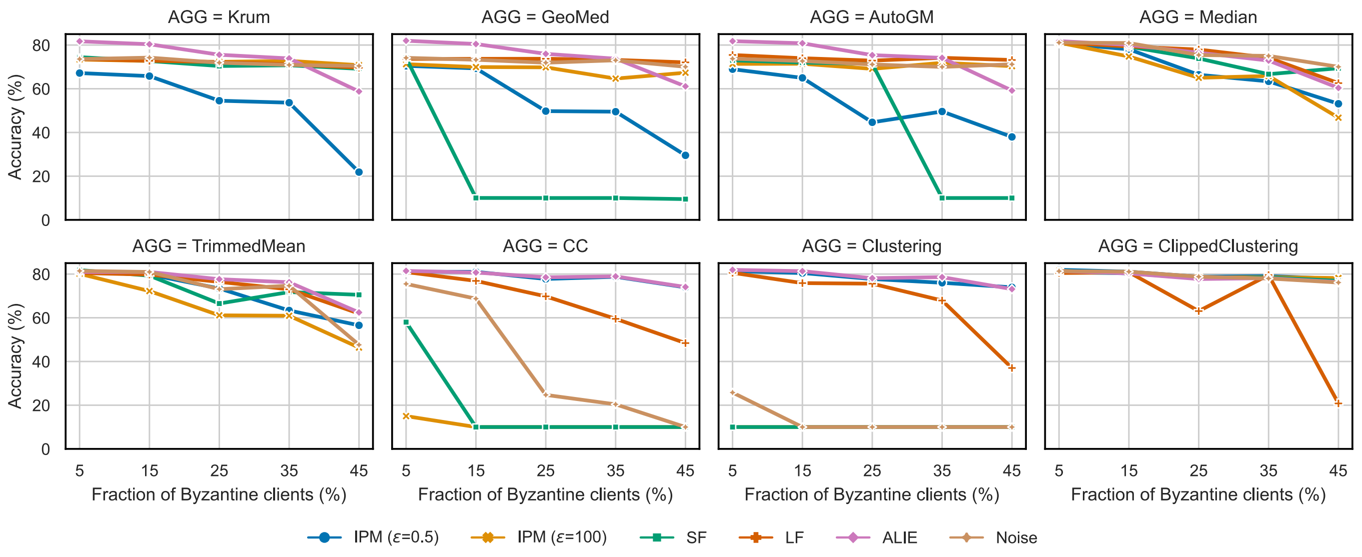

Fig. 5. Impact of the fraction of Byzantine clients on test accuracy for CIFAR-10 with *FedAvg*. `Median`, `TrimmedMean`, and `ClippedClustering` show high tolerance to most attacks.

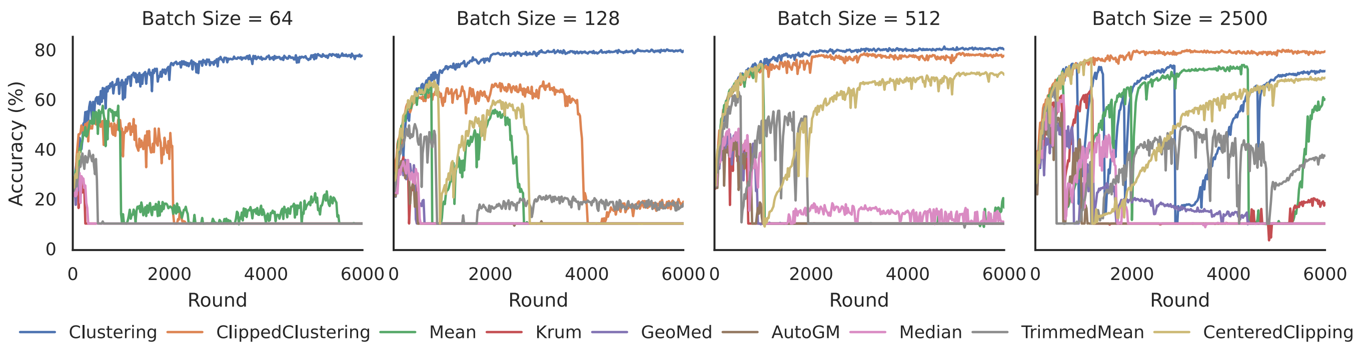

Fig. 6. Impact of batch size on the performance of *FedSGD* with IID partition under ALIE attack. The aggregation schemes (including `Mean`) become more robust as the batch size increases.

FedSGD on Non-IID MNIST with 15% clients are malicious, the loss increases to the upper bound (10^5) after 4000 rounds, the model, however, still remains 92% accuracy. This is because the top-1 accuracy (as we use in this paper) only takes into account the output with the highest probability. In this case, the attackers significantly change the output distribution but fail to change the index of the highest output.

G. Pairwise Cosine Similarities

Recall that the aggregation schemes show different robustness in *FedSGD* and *FedAvg* with respective to IID and Non-IID data partitions. To further investigate this issue, we compare the pairwise cosine similarities of all benign local updates without attacks. We note that cosine similarly reflects the angle between two vectors, i.e., a higher value indicates a smaller angle. As visualized in Fig. 8, the pairwise cosine similarities of updates from *FedSGD* vary widely, no matter if the local datasets are IID or not. Considerable pairs of clients even show negative similarities. Such a stochastic property may confuse the robust aggregation schemes and make it more challenging to detect malicious updates. The updates from *FedAvg* with IID data show the highest pairwise similarities, which means that their update directions are almost identical. Benefiting from this, clustering-based schemes can group benign updates together and exclude malicious updates. However, they become less similar when the local datasets are Non-IID.

VII. Related Work

A. Byzantine Attacks on FL

Byzantine attacks on FL are carried out by malicious clients during the distributed optimization of machine learning models, aiming to bias the global model to the desire of malicious clients [5]. Depending on the adversarial goals, Byzantine attacks in FL can be classified into two categories: targeted attacks and untargeted attacks [5], [29]. Targeted attacks, such as backdoor attacks, aim to make the global model generate attacker-desired misclassifications for some particular test samples [45], [46], [47]. While untargeted attacks aim to degrade the overall performance of the global model indiscriminately [7].

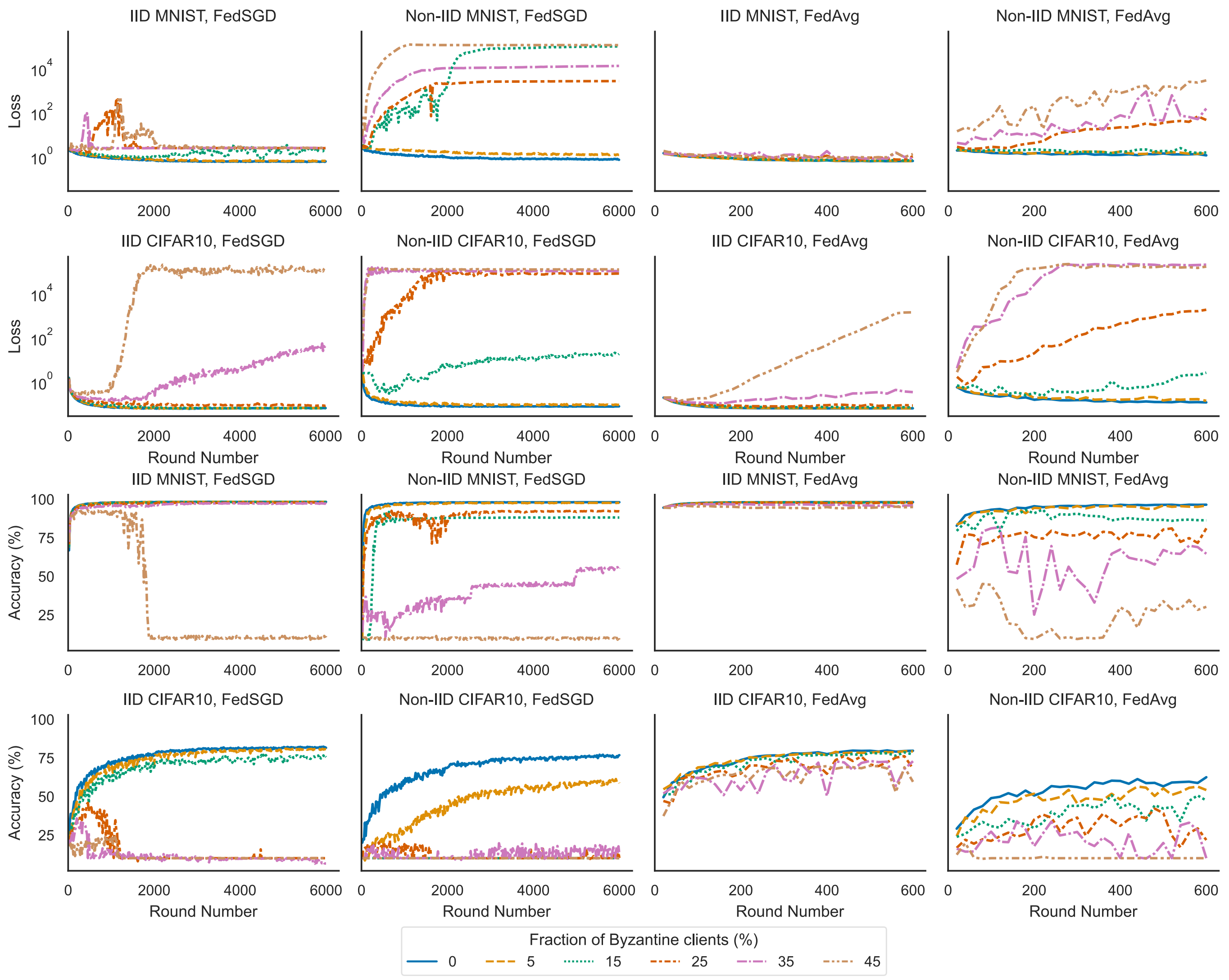

Fig. 7. Adaptive attack to `ClippedClustering` scheme. The top two rows show the test loss and the bottom rows show the test accuracy. We clamp the loss into the range $[0, 10^5]$. The attack is particularly effective when the local datasets are Non-IID.

We particularly focus on untargeted attacks as most Byzantine-robust studies do [8], [17], [18], [19], [20], [21]. Many studies considered to launch attacks by adding Gaussian noise or flipping the sign of the actual updates [13], [19]. Those attacks, however, can be detected by some euclidean distance-based aggregation schemes such as `Krum` [17], as they usually make the malicious updates far from the benign ones when measured by euclidean distance. On the other hand, Baruch et al. [23] showed that the attackers can actually circumvent robust schemes including `TrimmedMean` and `Krum` by taking advantage of empirical variance between the updates of clients if such variance is high enough. Furthermore, Xie et al. proposed an attack strategy, Inner Product Manipulation (IPM) attack that poses a significant threat to `Median` and `Krum` by manipulating the inner product between the true mean of the updates and the output of the aggregation schemes [24].

There exists a debate over Byzantine attacks in real-world FL systems. Shejwalkar et al. [48] claimed that the fraction of compromised genuine clients are usually small (e.g., 0.01%) in practice due to the high cost of hacking and manipulating multiple devices simultaneously, resulting in a very limited impact on the global model. However, Cao et al. [49] argued that the attackers may address this limitation by injecting fake clients into FL systems using zombie devices and simulators.

B. Byzantine-Robust FL

In FL settings, a number of strategies have been explored to defend against specific types of attacks or failures, including backdoor attacks [47], [50], free-rider attacks [51], [52], and gradient inversion attacks [53], [54]. On a more general level, Byzantine-robust FL solutions aim to mitigate the effect of arbitrary updates uploaded by malicious clients, instead of focusing on specific types of attacks [7]. Those Byzantine-robust solutions can be classified into three categories: redundancy-based schemes, trust-based schemes, and robust aggregation schemes.

Redundancy-based schemes assign each client redundant updates and use this redundancy to eliminate the effect of Byzantine

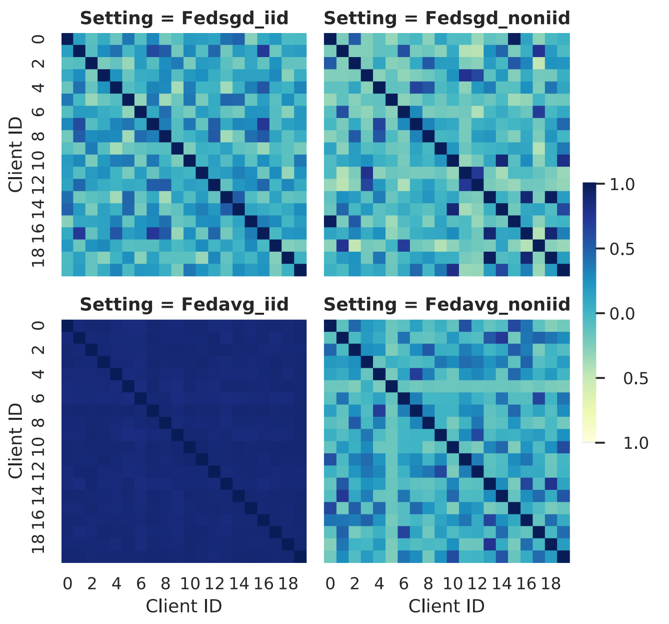

Fig. 8. Pairwise cosine similarities between local updates in different scenarios, where a similarly value 1 represents identical update directions, and -1 indicates opposite directions. *FedAvg* with IID data results in the most similar directions.

failures [10], [11], [55]. In 2018, Chen et al. [10] presented a framework, DRACO, for robust distributed training that uses ideas from coding theory. In DRACO, each client evaluates redundant gradients that are used by the server to eliminate the effects of adversarial updates. In 2019, Rajput et al. presented DETOX, a framework that combines algorithmic redundancy with robust aggregation. The defense of DETOX operates in two steps, a filtering step that uses limited redundancy to significantly reduce the effect of Byzantine nodes, and a hierarchical aggregation step that can be used in tandem with any state-of-the-art robust aggregation method. However, these redundant updates, in the worst case, require each node to compute $\Omega(M)$ times more updates, where M is the number of Byzantine clients [10]. This overhead is prohibitive in settings with a large number of Byzantine clients. In 2021, Cao et al. [56] proposed to use a randomly selected subset of clients to learn redundant global models. At inference time, it takes the majority vote among the global models when predicting the label of a testing sample. The authors showed that such an ensemble approach with any base FL algorithm is provably secure against malicious clients.

Trust-based schemes assume that some of the clients or datasets are trusted for filtering and re-weighting the local model updates [13], [14], [15], [16]. For example, in 2019, Li et al. [13] proposed to incorporate the objective function with a regularization term, which minimizes the distance between the server parameters and the client parameters. In 2021, Park et al. [15] designed an entropy-based filtering scheme to detect the outlier updates based on some trusted public data on the server side. During the training, the server computes the entropy of each update with the trusted dataset. Based on their experimental observations, they argue that the updates with higher entropy will lead to lower accuracy during the testing stage. Thus, they set a threshold for the entropy and filter out updates with entropy higher than the threshold. In 2021, Cao et al. [16] utilized cosine similarity to measure the similarity between updates submitted by the clients and the update obtained by training based on the trusted dataset owned by the server. The authors argued that an attacker can manipulate the directions of updates to perform model poisoning attacks, and the directions of the updates can, to a certain extent, indicate the honesty of the end devices. After the calculation of cosine similarity, the server calculates a trust score for each update using the ReLu function. The score is then used as the weight for the global model aggregation. In general, trust-based schemes have the potential to deal with situations where more than half of the updates are malicious according to some pre-validated information to detect malicious updates. However, trusted datasets or clients are not always available for the server, for example, due to the concern of user data privacy.

Robust aggregation schemes estimate the global update based on the local updates according to their robust aggregation rules or algorithms [8], [17], [18], [19], [20], [21]. Byzantine-robust aggregation has been explored to handle the devices sending corrupted updates to the server, including geometric median (`GeoMed`) [18], `Krum` [17], `TrimmedMean` [8], and `Median` [8]. They are commonly used to estimate the model parameters and mitigate the effect of malicious updates in global aggregation. In 2017, Chen et al. [18] proposed a `GeoMed`-based method to aggregate the gradients for distributed statistical machine learning and showed the robustness and convergence in i.i.d settings. In 2020, Wu et al. [57] showed that `GeoMed` scheme provably provides improved Byzantine robustness compared to other aggregation schemes in FL. In 2022, Pillutla et al. [30] applied `GeoMed` as a robust aggregation rule for FL and analyze the convergence of the resulting FL algorithm for least-squares objective with IID local datasets. In 2022, Li et al. [19] proposed `AutoGM`, a variant of `GeoMed` that automatically re-scales the weight of each parameter component according to a user-specified threshold of skewness. According to our empirical study in this paper, this category of schemes all show limitations in *FedSGD* and *FedAvg* algorithms in terms of Byzantine robustness.

VIII. Conclusion

In this paper, we provided an experimental study of Byzantine robust aggregation schemes for FL. In particular, we survey existing Byzantine attacks and defense strategies in the FL literature. We also propose a novel scheme, `ClippedClustering`, which enhances the robustness of the clustering-based scheme by automatically clipping the updates to mitigate the effect of amplified malicious updates. We then evaluate eight robust aggregation schemes under five representative Byzantine attack strategies. Our experimental results show that all those aggregation schemes achieve limited robustness in the presence of Byzantine attacks. In the future, it would be interesting to carry out a theoretical analysis to guarantee the robustness of `ClippedClustering`. Furthermore, we plan to improve the robustness of FL from more perspectives, e.g., low variance algorithms, and robust learning rates.

REFERENCES

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[2] Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated machine learning: Concept and applications,” *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, 2019.

[3] J. KonečnÝ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, “Federated learning: Strategies for improving communication efficiency,” 2016, *arXiv:1610.05492*.

[4] Y. Arjevani and O. Shamir, “Communication complexity of distributed convex learning and optimization,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1756–1764.

[5] L. Lyu, H. Yu, J. Zhao, and Q. Yang, “Threats to federated learning,” in *Federated Learning*. Berlin, Germany: Springer, 2020, pp. 3–16.

[6] F. Ang, L. Chen, N. Zhao, Y. Chen, W. Wang, and F. R. Yu, “Robust federated learning with noisy communication,” *IEEE Trans. Commun.*, vol. 68, no. 6, pp. 3452–3464, Jun. 2020.

[7] M. Fang, X. Cao, J. Jia, and N. Gong, “Local model poisoning attacks to byzantine-robust federated learning,” in *Proc. 29th USENIX Secur. Symp.*, 2020, pp. 1605–1622.

[8] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, “Byzantine-robust distributed learning: Towards optimal statistical rates,” in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5650–5659.

[9] S. Hu, J. Lu, W. Wan, and L. Y. Zhang, “Challenges and approaches for mitigating byzantine attacks in federated learning,” 2021, *arXiv:2112.14468*.

[10] L. Chen, H. Wang, Z. Charles, and D. Papailiopoulos, ”Draco: Byzantine-resilient distributed training via redundant gradients,” in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 903–912.

[11] S. Rajput, H. Wang, Z. Charles, and D. Papailiopoulos, “DETOX: A redundancy-based framework for faster and more robust gradient aggregation,” *Adv. Neural Inf. Process. Syst.*, vol. 32, pp. 10320–10330, 2019.

[12] J.-Y. Sohn, D.-J. Han, B. Choi, and J. Moon, “Election coding for distributed learning: Protecting signsgd against byzantine attacks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 14615–14625.

[13] L. Li, W. Xu, T. Chen, G. B. Giannakis, and Q. Ling, “Rsa: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets,” in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 1544–1551.

[14] N. Konstantinov and C. Lampert, “Robust learning from untrusted sources,” in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 3488–3498.

[15] J. W. Park, D.-J. Han, M. Choi, and J. Moon, “Sageflow: Robust federated learning against both stragglers and adversaries,” *Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 840–851, 2021.

[16] X. Cao, M. Fang, J. Liu, and N. Z. Gong, “Fltrust: Byzantine-robust federated learning via trust bootstrapping,” in *Proc. ISOC Netw. Distrib. Syst. Secur. Symp.*, 2021.

[17] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, “Machine learning with adversaries: Byzantine tolerant gradient descent,” in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 118–128.

[18] Y. Chen, L. Su, and J. Xu, “Distributed statistical machine learning in adversarial settings: Byzantine gradient descent,” *Proc. ACM Meas. Anal. Comput. Syst.*, 2017, pp. 1–25.

[19] S. Li, E. Ngai, and T. Voigt, “Byzantine-robust aggregation in federated learning empowered industrial IoT,” *IEEE Trans. Ind. Informat.*, vol. 19, no. 2, pp. 1165–1175, Feb. 2023.

[20] S. P. Karimireddy, L. He, and M. Jaggi, “Learning from history for byzantine robust optimization,” in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 5311–5319.

[21] F. Sattler, K.-R. Müller, T. Wiegand, and W. Samek, “On the byzantine robustness of clustered federated learning,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2020, pp. 8861–8865.

[22] R. Guerraoui et al., “The hidden vulnerability of distributed learning in byzantium,” in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 3521–3530.

[23] G. Baruch, M. Baruch, and Y. Goldberg, “A little is enough: Circumventing defenses for distributed learning,” *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8635–8645.

[24] C. Xie, O. Koyejo, and I. Gupta, “Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation,” in *Proc. Uncertainty Artif. Intell.*, 2020, pp. 261–270.

[25] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, “On the convergence of fedavg on non-iid data,” in *Proc. Int. Conf. Learn. Representations*, 2020.

[26] Z. Li, V. Sharma, and S. P. Mohanty, “Preserving data privacy via federated learning: Challenges and solutions,” *IEEE Consum. Electron. Mag.*, vol. 9, no. 3, pp. 8–16, May 2020.

[27] H. P. Lopuhaa and P. J. Rousseeuw, “Breakdown points of affine equivariant estimators of multivariate location and covariance matrices,” *Ann. Statist.*, vol. 19, no. 1, pp. 229–248, 1991.

[28] T. Li, S. Hu, A. Beirami, and V. Smith, “Ditto: Fair and robust federated learning through personalization,” in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 6357–6368.

[29] M. S. Jere, T. Farnan, and F. Koushanfar, “A taxonomy of attacks on federated learning,” *IEEE Secur. Privacy*, vol. 19, no. 2, pp. 20–28, Mar./Apr. 2021.

[30] K. Pillutla, S. M. Kakade, and Z. Harchaoui, “Robust aggregation for federated learning,” *IEEE Trans. Signal Process.*, vol. 70, pp. 1142–1154, 2022.

[31] M. B. Cohen, Y. T. Lee, G. Miller, J. Pachocki, and A. Sidford, “Geometric median in nearly linear time,” in *Proc. 48th Annu. ACM Symp. Theory Comput.*, 2016, pp. 9–21.

[32] Z. Li, L. Liu, J. Zhang, and J. Liu, “Byzantine-robust federated learning through spatial-temporal analysis of local model updates,” 2021, *arXiv:2107.01477*.

[33] P. Seetharaman, G. Wichern, B. Pardo, and J. Le Roux, “Autoclip: Adaptive gradient clipping for source separation networks,” in *Proc. IEEE 30th Int. Workshop Mach. Learn. Signal Process.*, 2020, pp. 1–6.

[34] G. Andrew, O. Thakkar, B. McMahan, and S. Ramaswamy, “Differentially private learning with adaptive clipping,” *Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 17455–17466, 2021.

[35] V. Shejwalkar and A. Houmansadr, “Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning,” in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2021.

[36] A. Krizhevsky, “Learning multiple layers of features from tiny images,” Univ. Toronto, Toronto, Tech. Rep. TR-2009, 2009.

[37] Y. LeCun, C. Cortes, and C. Burges, “MNIST handwritten digit database,” *ATT Labs*, vol. 2, 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist

[38] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, “Ensemble distillation for robust model fusion in federated learning,” *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 2351–2363, 2020.

[39] S. Li, E. Ngai, F. Ye, and T. Voigt, “Auto-weighted robust federated learning with corrupted data sources,” *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 5, pp. 1–20, Jun 2022.

[40] A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi, “Escaping the Big Data paradigm with compact transformers,” 2021, *arXiv:2104.05704*.

[41] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, “Federated learning: Challenges, methods, and future directions,” *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.

[42] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, “SCAFFOLD: Stochastic controlled averaging for federated learning,” in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 5132–5143.

[43] Y. Deng, M. M. Kamani, and M. Mahdavi, “Adaptive personalized federated learning,” 2020, *arXiv: 2003.13461*.

[44] F. Faghri, D. Duvenaud, D. J. Fleet, and J. Ba, “A study of gradient variance in deep learning,” 2020, *arXiv: 2007.04532*.

[45] C. Xie, K. Huang, P.-Y. Chen, and B. Li, “DBA: Distributed backdoor attacks against federated learning,” in *Proc. Int. Conf. Learn. Representations*, 2020.

[46] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, “How to backdoor federated learning,” in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2938–2948.

[47] S. Andreina, G. A. Marson, H. Möllering, and G. Karame, “BaFFLe: Backdoor detection via feedback-based federated learning,” in *Proc. IEEE 41st Int. Conf. Distrib. Comput. Syst.*, 2021, pp. 852–863.

[48] V. Shejwalkar, A. Houmansadr, P. Kairouz, and D. Ramage, “Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning,” in *Proc. IEEE Symp. Secur. Privacy*, 2022, pp. 1354–1371.

[49] X. Cao and N. Z. Gong, “Mpaf: Model poisoning attacks to federated learning based on fake clients,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 3396–3404.

[50] C. Zhao, Y. Wen, S. Li, F. Liu, and D. Meng, “Federatedreverse: A detection and defense method against backdoor attacks in federated learning,” in *Proc. ACM Workshop Inf. Hiding Multimedia Secur.*, 2021, pp. 51–62.

[51] J. Lin, M. Du, and J. Liu, “Free-riders in federated learning: Attacks and defenses,” 2019, *arXiv: 1911.12560*.

[52] Y. Fraboni, R. Vidal, and M. Lorenzi, “Free-rider attacks on model aggregation in federated learning,” in *Proc. Int. Conf. Artif. Intell. Statist.*, 2021, pp. 1846–1854.

[53] Y. Huang, S. Gupta, Z. Song, K. Li, and S. Arora, “Evaluating gradient inversion attacks and defenses in federated learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 7232–7241.

[54] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, “Inverting gradients - how easy is it to break privacy in federated learning?,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 16937–16947.

[55] D. Data, L. Song, and S. Diggavi, “Data encoding for byzantine-resilient distributed gradient descent,” in *Proc. 56th Annu. Allerton Conf. Commun., Control, Comput.*, 2018, pp. 863–870.

[56] X. Cao, J. Jia, and N. Z. Gong, “Provably secure federated learning against malicious clients,” in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 6885–6893.

[57] Z. Wu, Q. Ling, T. Chen, and G. B. Giannakis, “Federated variance-reduced stochastic gradient descent with robustness to byzantine attacks,” *IEEE Trans. Signal Process.*, vol. 68, pp. 4583–4596, 2020.

Shenghui Li (Student Member, IEEE) received the BS degree from Xidian University, Xi’an, China, in 2017, and the MS degree from Sun Yat-sen University, Guangzhou, China, in 2019. He is currently working toward the PhD degree with the Department of Information Technology, Uppsala University, Uppsala, Sweden. His current research interests include federated learning, distributed optimization, and statistical machine learning.

Edith C.-H. Ngai (Senior Member, IEEE) is currently an Associate Professor with the Department of Electrical and Electronic Engineering, The University of Hong Kong. Before joining HKU, in 2020, she was an Associate Professor with the Department of Information Technology, Uppsala University, Sweden. She was a guest researcher with Ericsson Research Sweden, in 2015-2017. Previously, she has conducted research in Imperial College London, Simon Fraser University, Tsinghua University, and UCLA. Her research interests include Internet-of-Things, machine learning, data analytics, and smart cities. She is a VINNMER fellow (2009) awarded by Swedish Governmental Research Funding Agency VINNOVA. She led the “Green IoT” project in Sweden, which was named on the IVA’s 100-list from the Royal Swedish Academy of Engineering Sciences, in 2020. She is currently an area editor of IEEE INTERNET OF THINGS JOURNAL and an associate editor of IEEE ACCESS and IEEE TRANSACTIONS OF INDUSTRIAL INFORMATICS.

Thiemo Voigt (Member, IEEE) received the PhD degree from Uppsala University, Sweden, in 2002. He is currently a professor with the Department of Electrical Engineering, Uppsala University, where he leads the Division of Networked Embedded Systems. He is also a senior researcher with RISE Computer Science. His current research interests include low-power networking, system software for embedded networked devices and the Internet of Things. His work has been cited more than 19000 times. He is a member of the editorial board for the IEEE IoT Newsletter and ACM Transactions on Sensor Networks (TOSN).