Depriving the Survival Space of Adversaries Against Poisoned Gradients in Federated Learning

Jianrong Lu, Shengshan Hu, Wei Wan, Minghui Li, Leo Yu Zhang, Lulu Xue, Haohan Wang, and Hai Jin, *Fellow, IEEE*

***Abstract*—Federated learning (FL) allows clients at the edge to learn a shared global model without disclosing their private data. However, FL is susceptible to poisoning attacks, wherein an adversary injects tainted local models that ultimately corrupt the global model. Despite various defensive mechanisms having been developed to combat poisoning attacks, they all fall short of securing practical FL scenarios with heterogeneous and unbalanced data distribution. Moreover, the cutting-edge defenses currently at our disposal demand access to a proprietary dataset that closely mirrors the distribution of clients' data, which runs counter to the fundamental principle of privacy protection in FL. It is still challenging to devise an effective defense approach that applies to practical FL.**

In this work, we strive to narrow the divide between FL defense and its practical use. We first present a general framework to comprehend the effect of poisoning attacks in FL when the training data is not independent and identically distributed (non-IID). We then HeteroFL, a novel FL scheme that incorporates four complementary defensive strategies. These tactics are implemented in succession to refine the aggregated model toward approaching the global optimum. Ultimately, we devise an adaptive attack specifically for HeteroFL, aimed at offering a more thorough evaluation of its robustness. Our extensive experiments over heterogeneous datasets and models show that HeteroFL surpasses all state-of-the-art defenses in thwarting various poisoning attacks, *i.e.*, HeteroFL achieves global model accuracies comparable to the baseline, whereas other defenses suffer a significant accuracy reduction ranging from 34% to 79%.

***Keywords*—Federated learning, poisoning attacks, defenses.**

I. Introduction

Federated Learning (FL) [25] has gained considerable momentum as a distributed learning paradigm with promising potential. It offers a viable solution for collaborative learning among resource-constrained clients, *e.g.*, IoT devices, without compromising their privacy-sensitive data. In FL, numerous clients engage in local model training over their private datasets and upload local updates (or models) to a central server such as Google or Apple. The central server then employs aggregation strategies to publish an aggregated global model, which incorporates the contributions from the local models. So far, FL has gained widespread adoption in a variety of real-world applications such as finance [4], medical care [40], and smart city [17], *etc.*

FL is known to be vulnerable to poisoning attacks, wherein the adversary (*e.g.*, compromised clients) can taint the local training data (referred to as data poisoning attack [1], [34], [36]) or the local models (known as model poisoning attack [2], [13], [21], [33], [37]) to compromise the global model. For instance, the adversary may manipulate the labels of data to trigger erroneous predictions from the global model or generate a random local model to conserve computing resources while still benefiting from the global model.

To counteract poisoning attacks on FL, numerous research endeavors have been dedicated to crafting defensive FL schemes [3], [5], [26], [33], [35], [38], [41]. These schemes strive to devise a robust aggregation strategy on the server-side, with the goal of minimizing the influence of updates from potentially malicious clients on the global model. However, existing FL defenses suffer from two key limitations: 1) As will be explicitly discussed in Section IV, existing works have not taken full consideration of heterogeneous data distribution, which is one of the most important and basic characteristics of FL. Most existing defenses are only valid when clients' datasets are independent and identically distributed (IID) [3], [26], [33], [35], [41]. While recently proposed defenses claim to be Byzantine-robust in the non-IID setting [5], [38], we have experimentally found that they account for a very limited degree of heterogeneity, which is merely a special case of the practical FL scenarios; 2) To enhance the identification of harmful updates, certain defense mechanisms rely on a validation dataset [5], [38], whose distribution should be close or even identical to the distribution of data from clients. However, in FL scenarios where the original intention is to safeguard the privacy of diverse clients, this assumption seems to be overly restrictive and invalid. *How to protect FL against poisoning attacks while not breaching privacy remains a challenging problem, particularly when dealing with highly heterogeneous data.*

Our contributions. In this study, we thoroughly examine the failure of current approaches and then introduce HeteroFL, an innovative defensive FL scheme. Importantly, we provide an in-depth analysis of the impact of poisoning attacks on FL, as well as effective defense strategies specifically tailored for the non-IID setting.

A deep understanding of poisoning attacks on FL. We begin by demonstrating that all existing FL methods are inadequate when dealing with heterogeneous data. While recent studies [5], [29], [38], [43] have assessed their effectiveness on heterogeneous datasets, they have only explored a much milder form of heterogeneity: clients possess data samples with the same label types but only differing quantities per label. To provide a precise characterization, we put forward two metrics

Manuscript received June 28, 2023

that gauge the extent of heterogeneity in a given context: *load diversity* and *label diversity*. These two metrics facilitate a meticulous assessment of the disparities in the local training data among various clients. Following that, we give a detailed analysis of prevalent defense schemes [3], [5], [33], [38], [41] along with their failures. Drawing from our observations, we present a general framework to illustrate how the poisoned models injure the global model when operating on non-IID data. Our findings suggest that the non-IID setting substantially widens the legitimate direction space of local gradients, which is the primary factor that makes it exceedingly challenging to detect byzantine attacks.

A novel FL defense scheme. Our proposal is called HeteroFL, a squeezing-then-rectifying FL defense scheme to thwart data/model poisoning attacks. HeteroFL is the first scheme capable of protecting against the most challenging model poisoning attacks, *Min-Max*, and *Min-Sum*, outlined in [33], in highly heterogeneous environments. Our approach revolves around the idea of narrowing down the legitimate gradient direction space as much as possible and applying corrective aggregation to steer the aggregated global model toward the the optimal state attained in the absence of attacks. To achieve this goal, we suggest a four-stage procedure for progressively compressing the expansive legitimate gradient direction space. This will coerce attackers into crafting adversarial gradients that closely resemble the benign ones, ultimately curbing the effectiveness of the attacks. Specifically, we initially closely monitor the proximate gradients in the legitimate direction space, which prevents attackers from using sybil attacks to increase the stealthiness and potency of their attacks. We then employ a *clustering-then-grouping* strategy to identify strongly poisoned gradients in the space. This approach makes the local gradients less heterogeneous, allowing for the use of similarity detection techniques to filter out the poisoned gradients. Additionally, to further deprive the space available for the attackers to generate stealthier poisoned gradients, we embed a watermark into each local gradient using randomly generated trigger samples. This watermark allows us to keenly detect any minor adversarial manipulations on the local gradients. As a consequence, attackers are confined to a narrow legitimate direction space, which only permits the generation of weakly poisoned gradients. Finally, our corrective aggregation further mitigates the impact of these weakly poisoned gradients.

Defending against adaptive attacks. It is essential to assess the efficacy of HeteroFL in countering adaptive attacks, where the attacker is familiar with the defense mechanisms and can customize their attacks accordingly. To accomplish this, we leverage the state-of-the-art *AGR-tailored attack* [33] framework as a foundation to design a formidable adaptive attack specifically for HeteroFL. We explore the effectiveness-detectability trade-off for the adaptive attack and observe that the potent adaptive attack learns to circumvent each detection step. Nevertheless, the poisoned updates are already weakened by HeteroFL, rendering them useless.

Extensive evaluations. We conduct evaluations over four real-world heterogeneous datasets from different fields, including three image classification datasets and a Shakespeare dataset for a next-character prediction task. We evaluate multiple poisoning attacks, including label-flipping attack (a data poisoning attack), sign-flipping attack, adaptive attack, Min-Max attack, and Min-Sum attack [33] (model poisoning attacks). In a variety of heterogeneous scenarios, the results show that HeteroFL significantly outperforms state-of-the-art defense schemes. For instance, for CIFAR-10 with Resnet20, Multi-Krum (MKrum), Zeno, FLTrust, and DnC have $20\%, 21\%, 42\%$, and 62% reduction in global accuracy, respectively, while HeteroFL can achieve similar global model accuracy to the baseline with no attack. We also conduct ablation studies for the proposed four steps in HeteroFL. The results show that each step plays an indispensable role in detecting poisoned updates.

II. Background

A. Federated Learning

We consider a federated learning (FL) system with a central server and n clients. Each client i has a local training dataset D_i. Specifically, the FL system iteratively performs the following steps.

Step I: In the t-th iteration, the server sends the global model $\boldsymbol{w}_t$ to all the clients or a subset of them.

Step II: Each client i trains a new local model $\boldsymbol{w}_i^{t+1}$ over D_i by solving the optimization problem:

$$\arg\min_{\boldsymbol{w}_i^{t+1}} \ell(D_i, \boldsymbol{w}_i^{t+1}), \tag{1}$$

where $\ell(D_i, \boldsymbol{w}_i^{t+1})$ is the loss with $\boldsymbol{w}_t$ set as the initialization of $\boldsymbol{w}_i^{t+1}$. In particular, the optimization problem is usually solved by $\boldsymbol{w}_i^{t+1} \leftarrow \boldsymbol{w}_i^{t+1} - \alpha_i \frac{\partial \ell(D_i, \boldsymbol{w}_i^{t+1})}{\partial \boldsymbol{w}_i^{t+1}}$, where α_i is the learning rate of client i. After E_i iterations, the local update $\boldsymbol{g}_i^{t+1} = \boldsymbol{w}_i^{t+1} - \boldsymbol{w}^t$ will be uploaded to the server.

Step III: The server aggregates all the updates (usually using FedAvg [24]), and obtains a new global model update $\boldsymbol{g}^{t+1}$, Finally, the server updates the global model as $\boldsymbol{w}^{t+1} = \boldsymbol{w}^t + \boldsymbol{g}^{t+1}$.

B. Poisoning Attacks on Federated Learning

Poisoning attacks can break many machine learning systems, for instance, retrieval systems [20] and face recognition [32]. FL is also threatened by poisoning attacks [1], [2], [13], [21], [33], [34], [36], [37].

Data poisoning attacks. The adversary constructs poisoned model updates by contaminating the local training data [1], [34], [36]. Data poisoning attacks can be untargeted or targeted. The goal of untargeted attacks is to make the aggregated global model have low accuracy or converge slowly. For example, the label flipping attack changes the labels of training examples while keeping the features unchanged [34]. In targeted data poisoning attacks, the goal is to minimize the accuracy of the global model on attacker-chosen samples while keeping other non-target samples unaffected. For example, the backdoor attack injects backdoor triggers into a part of training samples, then the global model would falsely predict the specific samples to the target label while predicting other normal samples correctly [1], [36].

Model poisoning attacks. Unlike data poisoning attacks, in model poisoning attacks the adversary can manipulate local

model updates directly [2], [13], [21], [33], [37]. Conventional model poisoning attacks include sign-flipping attack [21], bit-flip attack [37], Gaussian attack [37], *etc*. However, these attacks result in significant differences between malicious and benign updates, making malicious updates easy to be detected. Baruch *et al.* [2] demonstrated that small but well-crafted perturbation is enough to circumvent defenses for distributed learning. To construct more powerful attacks, recently Fang *et al.* [13] formulated the local model poisoning attack as an optimization problem to find the optimal poisoned update. Based on this, Shejwalkar *et al.* [33] presented three state-of-the-art model poisoning attacks: *AGR-tailored attack* that aims to maximize the noise added to a guiding normal update, *Min-Max* and *Min-Sum* attacks that aim to minimize the maximum distance or the sum of distances between malicious update and benign updates.

III. Problem Formulation

Adversary's objective: We consider both data poisoning and model poisoning attacks including extensive attack methods. The adversary aims to reduce the accuracy of the final global model or prevent it from converging. We consider such an objective since most of the attacks will result in that FL cannot provide any model service for the clients. We admit that such an objective may not include backdoor attack, however, in the literature, backdoor attack is generally investigated independently of the other byzantine attacks, in which many backdoor defenses are complementary to our method.

Adversary's capability: Following existing works [3], [5], [15], [21], [26], [29], [35], [41], we assume that there are less than 50% attackers, *i.e.*, $f/n < 0.5$; here f stands for the number of attackers. The adversary can arbitrarily modify the labels of the training samples or tamper with the local model updates directly. We also let the adversary have full knowledge of the FL system, including its own local training data, the aggregation strategy, and the updates of benign clients.

Defender's objectives: HeteroFL aims to achieve the following goals:

- **Accuracy of the global model.** The method should achieve similar accuracy to the global model with the baseline where there is no attack.
- **Robustness in non-IID settings.** The method should prevent malicious updates from degrading the accuracy or the convergence rate of the global model in broad non-IID settings.
- **Efficiency for resource-constrained clients.** The method does not incur additional computation and communication overheads for the resource-constrained clients.
- **Privacy protection of clients.** The method should not incur any privacy concerns for clients.

Defender's knowledge and capability: We assume that the central server can only observe the local model updates received from clients in each iteration.

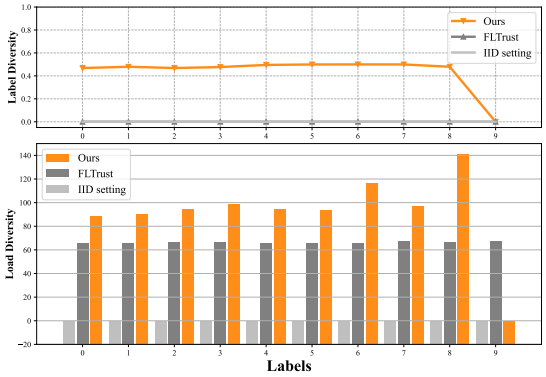


Fig. 1: Load diversity v.s. Label diversity

IV. A Deep Investigation on Poisoning Attacks and Existing Defenses

A. Existing Defenses Fail in Heterogeneous Scenarios

Heterogeneous local training data among different clients is one of the uppermost characteristic of FL. Although recently proposed schemes have performed experiments to demonstrate the effectiveness over heterogeneous datasets [5], [38], [42], their defense ability gets deteriorated rapidly when the heterogeneous degree becomes fairly higher, as evidenced through our extensive experiments shown in Section VII.

To give a precise illustration, we first propose two metrics to quantify the degree of a heterogeneous setting as follows.

- *Load diversity* evaluates the difference of the size of local training data set among clients. It is obtained by computing the standard deviation of the number of the training data samples for a given label as:

$$\boldsymbol{\varphi} = \sigma(\boldsymbol{v}_{\{i\in[n]\}}), \tag{2}$$

  where $\boldsymbol{v}_i$ is an L-dimensional statistical vector for client i, each element of which denotes the amount of training data with label l ($l \in [L-1]$), *i.e.*, $\boldsymbol{v}_i = (|D_i^0|, |D_i^1|, |D_i^2|, ..., |D_i^{L-1}|)$, and $\sigma(\cdot)$ denotes the dimension-wise standard deviation of $\boldsymbol{v}$, *i.e.*, φ is a vector.
- *Label diversity* evaluates the difference of training data labels among clients. It is obtained by computing the standard deviation of $\mathrm{sign}(\boldsymbol{v}_i)$ as:

$$\boldsymbol{\phi} = \sigma(\mathrm{sign}(\boldsymbol{v}_{\{i\in[n]\}})), \tag{3}$$

  where ϕ is a vector, sign denotes element-wise sign function of $\boldsymbol{v}$, and we set $\mathrm{sign}(0) = 0$.

In practical applications, it's common to encounter clients' training datasets that exhibit both high load diversity and high label diversity, especially when the datasets are sourced from different geographical regions. **Nevertheless, the experiments in existing works only consider load diversity while paying much less attention to label diversity.** Fig. 1 compares the experiment settings of FLTrust [5] and our work with standard IID distribution over the CIFAR-10 dataset in terms of load diversity and label diversity. We can see that although FLTrust has a much larger load diversity than the standard IID distribution, they have the same label diversity for all the

labels. In other words, the label types of the data samples between clients are the same, only the amount of samples for each label is different. On the contrary, our experiments for evaluating HeteroFL are set with high load and label diversities. Note that an exception occurs with label 9 since the samples with label 9 are used as the same trigger dataset for all clients.

B. Detailed Analysis of Existing Defenses

To figure out the reason of the failure of existing defenses, we provide a theoretical analysis *w.r.t.* the impact of load and label diversities on local gradients.

Theorem 1. *The difference of gradients between any two clients i and j is bounded by a number that is a function of data distributions and the local models of the previous round. Formally, we have:*

$$\begin{aligned}\|\boldsymbol{g}_i^{t+1}-\boldsymbol{g}_j^{t+1}\|\leq &\sum_{l=0}^{L-1}\tau_{\boldsymbol{x}_i|y_j=l}\left\|\boldsymbol{w}_i^t-\boldsymbol{w}_j^t\right\|+\\ &G\sum_{l=0}^{L-1}\left|\mathcal{X}_i[y_i=l]-\mathcal{X}_j[y_j=l]\right|,\end{aligned} \tag{4}$$

where $\|\cdot\|$ denotes the ℓ_2-norm; τ and G represent Lipschitz constant and the upper bound of gradient respectively.

Proof: see supplementary.

Remark 1. From Eq. (4), we know that the differences between local model updates are mainly determined by two parts: local data distributions $\mathcal{X}_i$ and $\mathcal{X}_j$, and the previous round local models $\boldsymbol{w}_i^t$ and $\boldsymbol{w}_j^t$, which is consistent with [44]. In the centralized FL, all the clients will receive the same global model at the beginning of round $t+1$, *i.e.*, $\boldsymbol{w}_i^t=\boldsymbol{w}_j^t$, so local models (and the updates) of clients i and j are similar to each other for the IID setting, *i.e.*, $\mathcal{X}_i=\mathcal{X}_j$. When the load diversity and label diversity increase, the difference between $\mathcal{X}_i$ and $\mathcal{X}_j$ becomes larger, such that the gradients and local model updates gradually differ from each other with the iterations. Next, we will make use of this observation to analyze five state-of-the-art defenses in the non-IID setting. Please refer to the supplementary for more details of these defenses. In addition, a detailed introduction for existing works are moved to the supplementary.

Multi-Krum [3]: Multi-Krum does not work in the non-IID setting since the difference between benign updates, evaluated by Euclidean distance, may be larger than that of a malicious update and benign updates, thus leading to the excluding of benign updates.

Median [41]: In the non-IID setting, due to the large legitimate direction space of local gradient caused by heterogeneous data distribution, the median value in each dimension could be significantly different from the result of FedAvg, resulting in poor accuracy of the resultant global model.

Zeno [38]: Zeno fails to work in the non-IID setting for two reasons. First, using an IID dataset to evaluate a local gradient that is trained over a non-IID dataset will cause a large estimated error, because there exist data samples in the IID dataset that local models have not learned. Second, the heterogeneity property in local hyper-parameters (*e.g.*, the amount of batches) determines the magnitude of local gradients. As a result, the magnitudes of benign gradients may become much larger than that of malicious gradients.

FLTrust [5]: As suggested in Theorem 1, the non-IID setting enlarges the legitimate direction space. Therefore it is possible that the cosine similarity between the guiding gradient and the benign local gradient is evaluated as negative, making the scores assigned to benign gradients become zero after clipping. As a result, they will be discarded while poisoned ones get preserved.

DnC [33]: As the authors indicated, protecting FL from adaptive attacks in non-IID settings is the shortcoming of DnC. This limitation derives from the assumption that the common features of poisoned and benign gradients should be sufficiently separated, which apparently violates Theorem 1.

In summary, when dealing with heterogeneous data distribution in FL, it is difficult to distinguish elaborately crafted poisoned models from benign models by directly using similarity detection over the raw local models, which is the underpinning of most existing Byzantine-robust aggregation algorithms.

C. A General Framework for Understanding Poisoning Attack

To design a well-working defense method in heterogeneous settings, we must first figure out the fundamental mechanism of how poisoned local gradients impact the global model. Existing study [44] shows that the centralized training over D ($D=\bigcup_{i=1}^{n}D_i$) following a distribution $\mathcal{X}$ can achieve higher accuracy than the aggregated global model from FL. For drawing a general conclusion, we take the centralized training as a baseline to analyze the model accuracy of FL under IID, non-IID, and adversarial settings, respectively. We define the model from the centralized training as the optimal one.

Suppose the centralized training performs T iterations with B batches and a batch size of S, then we denote $\boldsymbol{d}_b^{t,l}$ as the optimal gradient *w.r.t.* data sample $(\boldsymbol{x},l)$ in the b-th batch at iteration t, where $l\in[L-1], b\in[B], t\in[T]$. For simplicity, we assume FL also performs T global iterations and all the clients perform E local epochs over B batches with a batch size of S. We denote $\boldsymbol{g}_{i,b,e}^{t,l}$ as the local update (gradient) *w.r.t.* data sample $(\boldsymbol{x}_i,l)$ in the b-th batch at local epoch e at iteration t, where $l\in[L-1], b\in[B], e\in[E], t\in[T]$. When considering the IID setting, we refer to the IID setting where $D_i=D$ $(i\in[n])$ and all the clients perform training totally in the same way with the training. Therefore, all the local models are updated toward the optimal point via the same updating path, *i.e.*, $\boldsymbol{g}_{i,b,e}^{t,l}=\boldsymbol{d}_b^{t,l}$, $(i\in[n])$. When considering the non-IID setting, since each client follows a distribution $\mathcal{X}_i$ differing from $\mathcal{X}$, the local gradient of client i is thus different from the optimal gradient $\boldsymbol{d}_b^{t,l}$. Therefore, there will be a deviation over $\boldsymbol{d}_b^{t,l}$, denoted as $\boldsymbol{\Delta}_{i,b,e}^{t,l}$. Then after one epoch of the local training in FL, the deviation of client i is accumulated as $\sum_{b=1}^{B}\sum_{l=0}^{L-1}\frac{1}{S}\boldsymbol{\Delta}_{i,b,e}^{t,l}$. When considering data poisoning attack, an attacker has two ways to poison the dataset: changing the training data $\boldsymbol{x}_i$ into $\widetilde{\boldsymbol{x}}_i$ or its label l into $\widetilde{l}$. Hence the attacker will get a malicious gradient

$\widetilde{\boldsymbol{g}}_{i,b,e}^{t,l}$ trained over poisoned data sample $(\widetilde{\boldsymbol{x}}_i, \widetilde{l})$. Compared with the local gradient $\boldsymbol{g}_{i,b,e}^{t,l} = \boldsymbol{d}_b^{t,l} + \boldsymbol{\Delta}_{i,b,e}^{t,l}$ on the non-IID setting, the data poisoning attack in fact only causes a deviation $\widetilde{\boldsymbol{\Delta}}_{i,b,e}^{t,l} = \widetilde{\boldsymbol{g}}_{i,b,e}^{t,l} - \boldsymbol{g}_{i,b,e}^{t,l}$. Then in one epoch, the deviation is accumulated as $\sum_{b=1}^{B}\sum_{l=0}^{L-1}\frac{1}{S}\widetilde{\boldsymbol{\Delta}}_{i,b,e}^{t,l}$. For the model poisoning attack, the most simple and effective approach for an attacker is replacing the benign gradient with an arbitrary one, which leads to a deviation on the local gradient $\boldsymbol{g}_i^t$ after one round. We denote such deviation as $\widetilde{\boldsymbol{\Delta}}_i^t$. Then, in the t-th iteration, any client i would derive the gradient:

$$\boldsymbol{g}_i^t = \sum_{e=1}^{E}\sum_{b=1}^{B}\sum_{l=0}^{L-1}\frac{1}{S}(\boldsymbol{d}_b^{t,l} + \boldsymbol{\Delta}_{i,b,e}^{t,l} + \widetilde{\boldsymbol{\Delta}}_{i,b,e}^{t,l}) + \widetilde{\boldsymbol{\Delta}}_i^t, \quad (5)$$

and the deviations have the following properties:

$$\boldsymbol{\Delta}_{i,b,e}^{t,l} = \begin{cases} \mathbf{0}, & D_i \text{ is IID}, \\ *, & D_i \text{ is non-IID}, \end{cases}$$

$$\widetilde{\boldsymbol{\Delta}}_{i,b,e}^{t,l} = \begin{cases} \mathbf{0}, & \text{client } i \text{ is honest}, \\ *, & \text{client } i \text{ conducts data poisoning}, \end{cases}$$

$$\widetilde{\boldsymbol{\Delta}}_i^t = \begin{cases} \mathbf{0}, & \text{client } i \text{ is honest}, \\ *, & \text{client } i \text{ conducts model poisoning}, \end{cases}$$

where "$*$" represents an arbitrary deviation direction. Let $\boldsymbol{\Delta}_{def}^t$ denote the rectified gradient generated by any defense method, then, after receiving all the updates, the server performs the aggregation as:

$$\begin{aligned} \boldsymbol{g}^t &= \sum_{i=1}^{n}\frac{1}{n}\boldsymbol{g}_i^t + \boldsymbol{\Delta}_{def}^t \\ &= \underbrace{\sum_{i=1}^{n}\sum_{e=1}^{E}\sum_{b=1}^{B}\sum_{l=0}^{L-1}\frac{1}{nS}\boldsymbol{d}_b^{t,l}}_{\boldsymbol{g}_{iid}^t} + \underbrace{\sum_{i=1}^{n}\sum_{e=1}^{E}\sum_{b=1}^{B}\sum_{l=0}^{L-1}\frac{1}{nS}\boldsymbol{\Delta}_{i,b,e}^{t,l}}_{\boldsymbol{\Delta}_{niid}^t} + \\ &\underbrace{\sum_{i=1}^{n}\sum_{e=1}^{E}\sum_{b=1}^{B}\sum_{l=0}^{L-1}\frac{1}{nS}\widetilde{\boldsymbol{\Delta}}_{i,b,e}^{t,l}}_{\widetilde{\boldsymbol{\Delta}}_{dp}^t} + \underbrace{\sum_{i=1}^{n}\frac{1}{n}\widetilde{\boldsymbol{\Delta}}_i^t}_{\widetilde{\boldsymbol{\Delta}}_{mp}^t} + \boldsymbol{\Delta}_{def}^t \\ &= \boldsymbol{g}_{iid}^t + \boldsymbol{\Delta}_{niid}^t + \widetilde{\boldsymbol{\Delta}}_{dp}^t + \widetilde{\boldsymbol{\Delta}}_{mp}^t + \boldsymbol{\Delta}_{def}^t, \end{aligned} \quad (6)$$

where $\boldsymbol{g}_{iid}^t$ represents the optimal global gradient obtained from the IID setting, $\boldsymbol{\Delta}_{niid}^t$, $\widetilde{\boldsymbol{\Delta}}_{dp}^t$ and $\widetilde{\boldsymbol{\Delta}}_{mp}^t$ represent the global gradient deviations generated by non-IID setting, data poisoning attack and model poisoning attack, respectively.

Remark 2: From Eq. (5) and (6), we can draw the following conclusions: **1) An effective poisoning attack will definitely cause the direction deviation of the global model.** Generally, the attackers can tamper with either the magnitude or the direction of the benign aggregated gradient $\boldsymbol{g}_{iid}^t + \boldsymbol{\Delta}_{niid}^t$ to destroy the global model accuracy. In practice, however, it cannot change the magnitude of $\boldsymbol{g}_{iid}^t + \boldsymbol{\Delta}_{niid}^t$ independently without affecting its direction, since the attackers are not able to obtain the aggregated direction $\boldsymbol{g}_{iid}^t + \boldsymbol{\Delta}_{niid}^t$ from benign clients in the t-th iteration. **2) A larger legitimate direction space is the key reason why non-IID settings make it more difficult to detect poisoned updates.** In IID settings, the legitimate direction of benign gradients is limited to be $\boldsymbol{g}_{iid}^t$, thus the attacker only has a quite small direction space to manipulate the benign gradients, which makes it much easier to be detected. But for the case of non-IID scenario, the legitimate direction space of benign gradients ranges from $\boldsymbol{g}_{iid}^t$ to $\boldsymbol{g}_{iid}^t + \boldsymbol{\Delta}_{niid}^t$, it thus becomes much more difficult to detect abnormality since the server cannot figure out whether such dissimilarity comes from the attack deviation $\widetilde{\boldsymbol{\Delta}}_{dp}^t + \widetilde{\boldsymbol{\Delta}}_{mp}^t$ or from the non-IID deviation $\boldsymbol{\Delta}_{niid}^t$.**3) An effective defense approach should eliminate malicious deviations as much as possible.** The primary approach for an effective defense lies in rectifying global gradient direction by mitigating the impact of $\widetilde{\boldsymbol{\Delta}}_{dp}^t + \widetilde{\boldsymbol{\Delta}}_{mp}^t$, such that the aggregated global model moves towards the optimal model that was trained under no poisoning. Fig. 2 gives a brief illustration for this observation. We can see that a defensive algorithm performs best iff $\boldsymbol{\Delta}_{def}^t = -(\widetilde{\boldsymbol{\Delta}}_{dp}^t + \widetilde{\boldsymbol{\Delta}}_{mp}^t)$.

V. HeteroFL: Our Defensive Scheme

A. Intuition and Overview

The intuition behind HeteroFL is to squeeze the legitimate direction space as much as possible, such that the available direction space for the attacker to manipulate is quite small, *i.e.*, resulting in a small $||\widetilde{\Delta}_{dp} + \widetilde{\Delta}_{mp}||$. Then, for the malicious updates from the squeezed space, we generate Δ_{def} to further rectify the deviation $\widetilde{\Delta}_{dp} + \widetilde{\Delta}_{mp}$. Specifically, we make use of four complementary defensive steps to sequentially squeeze the legitimate direction space and construct Δ_{def} by corrective aggregation. Fig. 3(a) shows the entire legitimate direction space with different kinds of poisoned updates. We divide the legitimate direction space into two parts: the strong effect area where a single poisoned update can significantly damage the global model accuracy, and the weak effect area where poisoned updates can only slightly change the direction of the global model after aggregation. As shown in Fig. 3(b), HeteroFL first restricts the legitimate direction space by discarding updates whose directions are excessively similar with each other, in order to prevent attackers from constructing multiple updates through reinforcing their similarity (*e.g.*, sybil attack [14]), which will finally make a significant impact on the global model. We call this sybil direction removal. Based on this, HeteroFL then aims to push the legitimate direction space into the weak effect area through removing all the updates that lie in the strong effect area. As shown in Fig. 3(c), in order to mitigate the impact of poisoned updates as much as possible, we will find a direction space (called detectable area) that is larger than the strong effect area by using our proposed similarity detection and watermark verification steps. Finally, HeteroFL rectifies the magnitudes of all the remaining updates, such that the malicious updates will be more similar to the benign local updates in terms of the magnitude (Fig. 3(d)). For the remaining negative impact from $\widetilde{\Delta}_{dp} + \widetilde{\Delta}_{mp}$, they will be

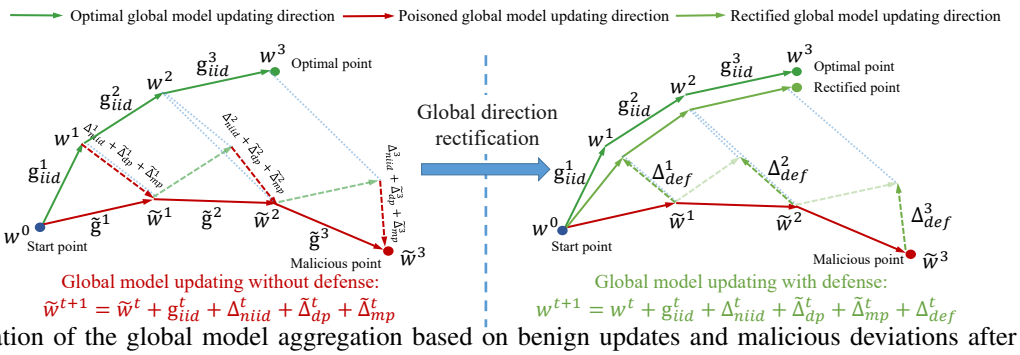
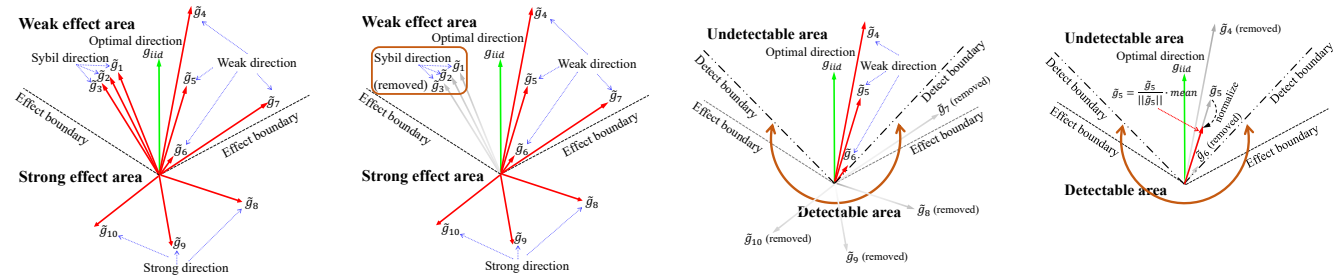


Fig. 2: Illustration of the global model aggregation based on benign updates and malicious deviations after three iterations



(a) The legitimate direction space with various poisoned updates.

(b) Sybil direction removal.

(c) Similarity detection & watermark verification based direction removal.

(d) Magnitude rectification.

Fig. 3: Illustration of our proposed four defensive steps

rectified with Δ_{def} generated from our corrective aggregation strategy.

B. A Complete Description of HeteroFL

Algorithm 1 shows the complete HeteroFL algorithm. After preprocessing an trigger dataset D_{trg} (Lines 1-3), T iterations are conducted. In each iteration, HeteroFL sequentially executes four steps, *i.e.*, *sybil direction removal* (Lines 9-10), *similarity detection* (Lines 11-12), *watermark verification* (Lines 13-14), *magnitude rectification* (Lines 15). Finally, with the help of magnitude rectification, corrective aggregation is achieved (Line 17-18).

C. Details of Our Defense Steps

Sybil direction removal. It is indicated in our experiment in supplementary that even in homogeneous data settings, updates that are extremely close with each other (measured by cosine similarity) have a higher possibility of being malicious. Removing such sybil gradient directions is beneficial to compelling attackers to construct poisoned updates with more abnormality. To this end, HeteroFL adaptively discards updates with high relative similarities beyond the clipping upper bound $2(\check{s} + \check{r})$ (Alg. 2). Since we assume $f/n < 0.5$, $\check{s} + \check{r}$ will approach the half of the maximal similarity of the benign updates. Hence, we can estimate a small upper bound of the benign similarities by $2(\check{s} + \check{r})$.

Similarity detection. Our intuition is that the gradients (including the aggregated ones) that are trained over the datasets following similar distribution can have a nearly consistent objective if they all are obtained from the same initial model. Hence, any (aggregated) gradient whose training dataset is poisoned will make the update move towards a biased objective. In view of this, HeteroFL adopts a *clustering-then-grouping* strategy to construct many aggregated gradients with similar objectives such that we can make use of similarity between the aggregated gradients to filter out malicious gradients. As shown in Alg. 4, HeteroFL first utilizes the popular mean shift clustering method [12] that can adaptively decide the number of the clusters to divide the update set (Lines 2-3) into several clusters, each of which consists of similar gradients which are trained over possibly similar data distributions. Then HeteroFL randomly selects a gradient from each cluster to constitute a new group (Lines 4-11) and computes the average as its centroid (Lines 13-16). Each centroid can be regarded as a gradient that is trained over a dataset that covers samples as diverse as possible. Therefore all the centroids can be viewed as "new" updates trained in an IID-like setting, namely, they have a more consistent objective than before. Since FL is used for large-scale distributed learning and usually includes a massive number of clients (*e.g.*, millions of users), we can construct sufficient groups that have similar centroids. Thus if any group contains malicious updates, the corresponding centroid will be far from the benign ones. In light of this, similarity can be used to identify malicious centroids/groups and the malicious updates in those groups (Lines 17-24). Note that existing schemes like APFed [9] simply remove the outliers in the clusters, while our similarity detection step designs a brand new grouping strategy to put gradients from different clusters into the same group. Furthermore, SmartFL [39] adjusts the aggregated weights, while FedEqual [8] adjusts the model weights of each layer. These aspects represent orthogonal designs that can be seamlessly integrated into our scheme.

In Fig. 4(a), we perform an ablation study to intuitively

Algorithm 1 Complete Description of HeteroFL

Input: A trigger dataset D_{trg}; number of classification tasks L; number of global iterations T; number of clients n; number of sampled clients m; filtering fraction β; loss reduction threshold λ.

Output: Final global model: $\boldsymbol{w}^T$.

Label the samples in D_{trg} with L.
The server sends D_{trg} and a randomly initialized model $\boldsymbol{w}^0$ with $L+1$ classifications to all the clients.
The clients append D_{trg} to their training datasets.
for $t = 0, 1, 2, \cdots, T-1$ **do**
  The server broadcasts $\boldsymbol{w}^t$ to m sampled clients.
  Each client i uploads update $\boldsymbol{g}_i^{t+1}$ using Eq. (1).
  The server receives the update set $C = \{\boldsymbol{g}_1^{t+1}, \boldsymbol{g}_2^{t+1}, \cdots, \boldsymbol{g}_m^{t+1}\}$.
  Initialize empty sets C_1, C_2, C_3, C_4 and $\widetilde{C}_1, \widetilde{C}_2, \widetilde{C}_2$.
  $\widetilde{C}_1 =$*SybilDirectionRemoval*(C)
  $C_1 \leftarrow C - \widetilde{C}_1$.
  $\widetilde{C}_2 =$*SimilarityDetection*(C_1, β).
  $C_2 \leftarrow C_1 - \widetilde{C}_2$.
  $\widetilde{C}_3 =$*WatermarkVerification*$(C_2, D_{trg}, \boldsymbol{w}^t, \lambda)$.
  $C_3 \leftarrow C_2 - \widetilde{C}_3$.
  $C_4 =$*MagnitudeRectification*(C_3).
  //Corrective aggregation.
  $\bar{\boldsymbol{g}} \leftarrow$ the average of all updates in C_4.
  $\boldsymbol{w}^{t+1} \leftarrow \boldsymbol{w}^t + \bar{\boldsymbol{g}}$.
end for
return $\boldsymbol{w}^T$.

Algorithm 2 SybilDirectionRemoval(C)

Output: A malicious updates set C_{mal}.

Initialize an empty set C_{mal}.
$\{s_1, s_2, \cdots, s_{|C|}\} =$*RelativeCosineSimilarity*(C, C)
$\check{s} \leftarrow$ the median of $\{s_1, s_2, \cdots, s_{|C|}\}$
$\check{r} \leftarrow$ the median of $\{|s_1 - \check{s}|, |s_2 - \check{s}|, \cdots, |s_{|C|} - \check{s}|\}$
Append all updates in C with $s_i > 2(\check{s} + \check{r})$ to C_{mal}
return C_{mal}.

Algorithm 3 RelativeCosineSimilarity(C_1, C_2)

Output: Relative similarity set.

for $i = 1, 2, \cdots, |C_2|$ **do**
  //Compute a score s_i for each update in C_2.
  $s_i = \sum_{C_1[j] \in \Gamma_{i, \frac{|C_1|}{2}}} \frac{\langle C_1[j], C_2[i] \rangle}{\|C_1[j]\| \cdot \|C_1[i]\|}$, where $\Gamma_{i, \frac{|C_1|}{2}}$ is the set of the half of updates from C_1 that have the highest cosine similarity with $C_2[i]$.
end for
return $\{s_1, s_2, \cdots, s_{|C_2|}\}$.

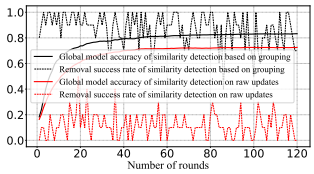
Algorithm 4 SimilarityDetection(C, β)

Output: A set of malicious updates C_{mal}.

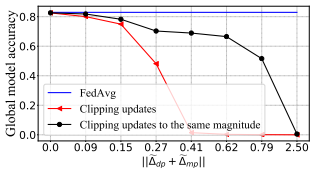
//Divide the update set C into c clusters by mean shift clustering that can dynamically determine the clusters.
$\{s_1, s_2, \cdots, s_{|C|}\} =$*RelativeCosineSimilarity*(C, C)
$Cluster_{\{i \in [c]\}} =$ *MeanShiftClustering*$(\{s_1, s_2, \cdots, s_{|C|}\})$
//Rearrange clusters into r groups such that each group contains the most diverse updates.
$r \leftarrow$ the size of the cluster that has the maximum number of updates.
for $i = 1, 2, \cdots, r$ **do**
  Initialize an empty set $Group_i$.
  for $j = 1, 2, \cdots, c$ **do**
    $Group_i[j] \leftarrow Cluster_j[i]$ **if** $Cluster_j[i] != \emptyset$.
  end for
end for
//Identify the groups that contain malicious updates.
Initialize three empty sets C_1, C_2, C_{mal}.
for $i = 1, 2, \cdots, r$ **do**
  $C_1[i] \leftarrow$ the average of all updates in $Group_i$.
end for
$\{s_1, s_2, \cdots, s_{|C_1|}\} =$*RelativeCosineSimilarity*(C_1, C_1)
$Cluster \leftarrow$ the cluster with the maximum number of updates by *MeanShiftClustering*$(\{s_1, s_2, \cdots, s_{|C_1|}\})$
Append updates in C_1 but not in $Cluster$ to C_{mal}.
//Select malicious updates from identified groups.
$C_2 \leftarrow$ set of updates of $Group_{\{i \in [r]\}}$ whose average lies in C_{mal}.
$\{s_1, s_2, \cdots, s_{|C_2|}\} =$*RelativeCosineSimilarity*(C, C_2)
$C_{mal} \leftarrow \emptyset$
Append $\beta \cdot |C_2|$ updates from C_2 with the lowest s_i to C_{mal}.
return C_{mal}.

show the usefulness of the clustering-then-grouping strategy. We can see that our similarity detection with the clustering-then-grouping strategy can remove much high percentages of the attackers while directly conducting similarity detection on the raw local updates barely spots the attackers. This indicates that our strategy can significantly improve the similarity of the benign updates such that the malicious ones are more easily detected by the similarity detection. We further prove this in supplementary.

Watermark verification. To prevent the adversary from compromising the model performance, the server can embed a watermark into each local model and then verify its effectiveness to detect poisoned models. To do this, our core idea is to send a watermark trigger dataset (D_{trg}), which is entirely different from the clients' datasets, to all clients for local training. The server can then verify the loss trend of each local model to identify whether the watermark has been broken. To this end, the server first extends the classification task number of the global model from L to $L+1$ and randomly generates samples with the new label L to serve as the watermark trigger dataset. By extending the model, the server can ensure that the watermark trigger dataset is entirely different from the clients' datasets. Then, the server will issue the trigger dataset and the extended global model to the clients, who will append D_{trg} to the local training



(a) The impact of our clustering-then-grouping strategy on similarity detection under label flipping attack.



(b) The impact of our clipping strategy on FedAvg when $||\widetilde{\Delta}_{dp}+\widetilde{\Delta}_{mp}||$ is increased.

Fig. 4: Ablation study on similarity detection and magnitude rectification over MNIST. We set $n = 100$ and $f = 20$. We define removal success rate as the ratio of malicious updates in the removed updates to the total number of attackers.

Algorithm 5 WatermarkVerification($C, D_{trg}, \boldsymbol{w}^t, \lambda$)

Output: a set of malicious updates C_{mal}.

1: Initialize an empty set C_{mal}.
2: **for** $i = 1, 2, \cdots, |C|$ **do**
3: //Evaluate the loss with an estimated model $\hat{\boldsymbol{w}}_i^{t+1}$.
4: $\hat{\boldsymbol{w}}_i^{t+1} = \boldsymbol{w}^t + C[i]$.
5: $loss\_reduction_i = \ell(D_{trg}, \boldsymbol{w}^t) - \ell(D_{trg}, \hat{\boldsymbol{w}}^{t+1})$.
6: **if** $loss\_reduction_i \leq \lambda$ **then**
7: Append $C[i]$ to C_{mal}.
8: **end if**
9: **end for**

return C_{mal}.

Algorithm 6 MagnitudeRectification(C)

Output: The final set of selected updates: C.

1: $median \leftarrow$ the median of ℓ_2 norms of updates in C.
2: Initialize empty set D.
3: **for** $i = 1, 2, \cdots, |C|$ **do**
4: **if** $\|C[i]\| < median$ **then**
5: Append $\frac{median}{\|C[i]\|}$ to D.
6: **else**
7: Append $\frac{\|C[i]\|}{median}$ to D.
8: **end if**
9: **end for**
10: //Remove updates with too small/large magnitudes by an adaptive clipping bound $8(\check{m} + \check{s})$.
11: $\check{m}, \check{s} \leftarrow$ the mean and standard deviation of the first $\frac{|D|}{2}$ smallest values in D.
12: **for** $i = 1, 2, \cdots, |D|$ **do**
13: **if** $D[i] > 8(\check{m} + \check{s})$ **then**
14: Remove $C[i]$ from C.
15: **end if**
16: **end for**
17: $mean \leftarrow$ the mean of ℓ_2 norms of updates in C.
18: **for** $i = 1, 2, \cdots, |C|$ **do**
19: $C[i] \leftarrow \frac{C[i]}{\|C[i]\|} \cdot mean$.
20: **end for**
21: **return** C.

dataset and initiate the local model with the extended global model (Lines 1-3 in Alg. 1). After receiving local updates trained over such a mixed dataset, the server will conduct watermark verification with D_{trg} (Line 13 in Alg. 1). The local updates whose loss reductions are smaller than a threshold λ are regarded as malicious and will be removed (Lines 4-7 in Alg. 5). Note that Zeno [38] also uses loss reduction to estimate the updates, however, it needs the dataset to follow the same distribution as that of clients, which infringes privacy. We make a comprehensive comparison between HeteroFL and Zeno (see the supplementary).

Magnitude rectification. For constructing an effective Δ_{def}, we assume $f/n < 0.5$. As shown in Alg. 6, HeteroFL adaptively removes the updates with too large or small magnitudes (Lines 10-17) and rescales updates to have the same magnitude that equals to the average of the remaining updates (Lines 18-20). In this way, Δ_{def} will be generated spontaneously when aggregating. Note that since $\check{m} + \check{s}$ approaches half of the maximal normalized magnitude of the benign updates. Hence, we can empirically estimate a bigger upper bound of the benign normalized magnitudes by $8(\check{m}+\check{s})$. Although clipping updates are considered in many previous defenses, none of them take all the updates into consideration to have the same magnitude for aggregation, which provides critical corrective aggregation, *i.e.*, generating Δ_{def} (theoretical analysis in supplementary). Fig. 4b shows that FedAvg with magnitude rectification can effectively eliminate the impact of attacks with a small $||\widetilde{\Delta}_{dp}+\widetilde{\Delta}_{mp}||$, while only clipping updates suffers from significant drops in accuracy, which means Δ_{def} can be obtained by clipping the updates to have the same magnitude.

VI. Adaptive Attacks on HeteroFL

A recently proposed adaptive attack [13], [33] is a new kind of attacks where the attacker knows the defensive method in advance and then adapts his attack strategy to circumvent the defense. In particular, AGR-tailored attack proposed in [33] is a state-of-the-art framework for constructing a strong adaptive attack. In this section, we first briefly introduce the AGR-tailored attack framework and then show the design of adaptive attack on HeteroFL based on the framework.

A. Adaptive Attack Framework

To ease the expression, we assume the first f updates are poisoned. In order to construct poisoned local updates, the adversary with AGR-tailored attack performs the following optimization problem:

$$\begin{aligned} &\underset{\gamma}{\text{argmax}} \ \|\boldsymbol{g} - \mathscr{A}(\widetilde{\boldsymbol{g}}_{\{i\in[f]\}} \cup \boldsymbol{g}_{\{i\in[f+1,n]\}})\| \\ &\widetilde{\boldsymbol{g}}_{\{i\in[f]\}} = \boldsymbol{g} + \gamma\widetilde{\boldsymbol{\Delta}}; \quad \boldsymbol{g} = \mathit{FedAvg}(\boldsymbol{g}_{\{i\in[n]\}}), \end{aligned} \tag{7}$$

where $\mathscr{A}$ is the known defense method, $\boldsymbol{g}_{\{i\in[n]\}}$ are the benign updates that the adversary knows, $\boldsymbol{g}$ is a reference benign aggregation obtained by FedAvg [24] that averages all the benign updates. $\widetilde{\boldsymbol{\Delta}}$ is a malicious perturbation and γ is a scaling coefficient. In [33], three types of $\widetilde{\boldsymbol{\Delta}}$ are given as follows:

- *Inverse unit vector:* $\widetilde{\boldsymbol{\Delta}}_{uv} = -\frac{\boldsymbol{g}}{\|\boldsymbol{g}\|}$.
- *Inverse standard deviation:* $\widetilde{\boldsymbol{\Delta}}_{std} = -\sigma(\boldsymbol{g}_{\{i\in[n]\}})$.
- *Inverse sign:* $\widetilde{\boldsymbol{\Delta}}_{sgn} = -\operatorname{sign}(\boldsymbol{g})$.

Algorithm 7 Optimization algorithm of scale factor γ

Input: γ_{init}, $\boldsymbol{g}_{\{i\in[n]\}}$, γ_{succ}, τ

$step \leftarrow \gamma_{init}/2$, $\gamma \leftarrow \gamma_{init}$
while $|\gamma_{succ} - \gamma| > \tau$ **do**
  if $|\{\boldsymbol{d} \in \widetilde{\boldsymbol{g}}_{\{i\in[f]\}} | \boldsymbol{d} \in C_4\}| = f$ **then**
    $\gamma_{succ} \leftarrow \gamma$
    $\gamma \leftarrow (\gamma + step/2)$
  else
    $\gamma \leftarrow (\gamma - step/2)$
  end if
  $step = step/2$
end while
return γ_{succ}

B. Instantiating Adaptive Attacks

We utilize the state-of-the-art framework to design and solve the optimization objective.

Optimization objective: Recall that HeteroFL performs four defensive steps to construct a benign selection set C_4 and calculates an average of the updates in C_4 as its aggregate. Thus, following [33], the adversary should maximize the number of poisoned updates in C_4 to the optimal value f, while maximizing the perturbation $\gamma\widetilde{\boldsymbol{\Delta}}$ added to the reference benign update $\boldsymbol{g}$ to boost the attack impact on the final aggregate. Formally, we have the following optimization objective to attack HeteroFL:

$$\begin{aligned} \underset{\gamma}{\text{argmax}}\ & f = |\{\boldsymbol{d} \in \widetilde{\boldsymbol{g}}_{\{i\in[f]\}} | \boldsymbol{d} \in C_4\}| \\ & \widetilde{\boldsymbol{g}}_{\{i\in[f]\}} = \boldsymbol{g} + \gamma\widetilde{\boldsymbol{\Delta}}_{\{i\in[f]\}}; \quad \boldsymbol{g} = \mathit{FedAvg}(\boldsymbol{g}_{\{i\in[n]\}}), \end{aligned} \tag{8}$$

where all $\widetilde{\boldsymbol{g}}_{\{i\in[f]\}}$ do not loss performance on D_{trg}; $\widetilde{\boldsymbol{\Delta}}_1$ is one of the three types of perturbation defined in Section VI-A; $\widetilde{\boldsymbol{\Delta}}_i$ $(2 \le i \le f)$ are generated by randomly flipping the signs of $\lceil \mu \cdot r \rceil$ elements in $\widetilde{\boldsymbol{\Delta}}_1$ to the opposite; here μ and r are the percentage of flips and the number of parameters in an update, respectively.

Solving the optimization objective: We strictly follow [33] to find the most efficient scale factor γ by Alg. 7. Given malicious directions $\widetilde{\boldsymbol{\Delta}}_{\{i\in[f]\}}$ and the reference update $\boldsymbol{g}$, Alg. 7 begins with a large initial γ to construct the poisoned updates.

VII. Experiments

We use FedML [16], a popular research library and benchmark for FL, to evaluate the effectiveness of HeteroFL against poisoning attacks, and compare with the baseline (*i.e.*, FedAvg [24] under no attack), FedAvg [24] under attacks (abbreviated as FedAvg-A), and the state-of-the-art defenses, including Multi-Krum (abbreviated as MKrum) [3], Median [41], Zeno [38], FLTrust [5], and DnC [33]. *In addition, we notice some recent related works [6], [10], [27]. As [10] requires linear regression analysis for the local model parameters of each dimension, which incurs significant computing overhead and exceeds our memory upper bound, we had to exclude it from our analysis. FedRecover [6] and FRL [27] are orthogonal to HeteroFL and potentially can be combined with ours. For a detailed analysis, please refer to the supplementary.*

A. Experimental Setup

1) Datasets and models: Please refer to supplementary.

2) Data partition method: Following FedML [16], we use the popular Latent Dirichlet Allocation (denoted as $Dir_n(q)$, q is the concentration parameter) to partition the CIFAR-10 dataset. Specifically, we simulate a heterogeneous partition for n clients by sampling $\boldsymbol{p}_l \sim Dir_n(q)$ and allocating a $\boldsymbol{p}_{l,i}$ proportion of the training instances of class l to local client i, where $\boldsymbol{p}_l$ is a n-dimensional vector and $\boldsymbol{p}_{l,i}$ is the i-th value in $\boldsymbol{p}_l$. Therefore, a smaller q leads to a higher heterogeneous degree. In our experiments we set $q = 0.2$.

3) Evaluated poisoning attacks: We evaluate both data poisoning attacks and model poisoning attacks.

Label flipping (LF) attack: LF is a data poisoning attack. Specifically, for each training example $(\boldsymbol{x_i}, y_i) \in D_i$, the data poisoning attackers flip its label y_i to $L - y_i - 1$.

Sign flipping (SF) attack: SF is a local model poisoning attack. An attacker would flip the direction of the local update g_i to $-zg_i$; here z is a scale factor.

Min-Max attack: Min-Max attack [33] is an optimization-based approach to make the poisoned updates lie close to the clique of the benign updates by minimizing the maximum distance between malicious gradient and other gradients.

Min-Sum attack: Min-Sum attack [33] aims to minimize the sum of distances between malicious gradient and all the benign gradients.

Adaptive attack: We evaluate the adaptive attack proposed in Section VI-B, which is the strongest attack on HeteroFL.

4) Evaluation metrics: We define *global model accuracy* as the proportion of correctly predicted testing samples to the total testing samples. Besides, we define *removal success rate* as the ratio of malicious updates in the removed updates to the total number of attackers.

5) Parameter settings: Please refer to the supplementary.

B. Comprehensive Comparison with State-of-the-art Defenses

HeteroFL achieves the four defense goals: We aim to design a defense method satisfying four goals presented in Section III. Table I exhibits the experimental results using diverse defenses under five attacks.

Accuracy: HeteroFL achieves nearly identical accuracy to the baseline [24]. Nevertheless, some of the existing defenses yield much lower accuracy even under no attacks. For example, on the CIFAR-10 dataset, the global model accuracy of HeteroFL is 72.39%, while that of FLTrust and Median are 58.47% and 68.64%, respectively. Note that there are counter-intuitive cases where the adaptive attack on HeteroFL is weaker than the other attacks (at most 0.0014 worse) because it uses a coarse-grained optimization way (solving for the most effective γ via updated step sizes). The original paper [33] also shows similar counterintuitive cases in its Table II (Purchase), *e.g.*, in Purchase (Table II), adaptive attack on Bulyan ($I_\theta = 28.7$) is 1.6 weaker than Min-Sum attack ($I_\theta = 30.3$).

Robustness: The accuracy of HeteroFL under all attacks is comparable to that of baseline. However, the accuracy of the existing defenses decreases significantly, especially under the adaptive attack. For example, on the CIFAR-10 and EMNIST

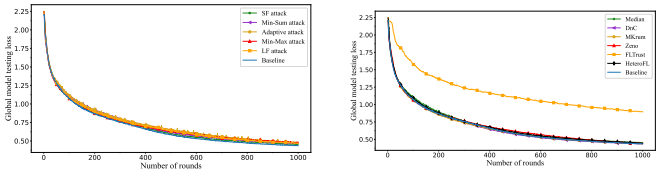
TABLE I: A comparison with state-of-the-art defense methods over heterogeneous datasets in terms of the global model accuracy.

Dataset (Model)	Defense	Without attack	Adaptive attack	Min-Max attack	Min-Sum attack	LF attack	SF attack
CIFAR-10 (ResNet20)	MKrum	**0.7323**	0.5331	0.6125	0.5789	0.7093	0.7017
	Median	0.6864	0.5070	0.5466	0.5017	0.6521	0.6506
	Zeno	0.7322	0.5163	0.6872	0.6309	**0.7299**	0.7119
	FLTrust	0.5847	0.1642	0.6073	0.5907	0.5618	0.5912
	DnC	0.7306	0.1111	0.2318	0.2240	0.7164	0.7109
	HeteroFL	0.7239	**0.7123**	**0.7201**	**0.7133**	0.7180	**0.7120**
EMNIST (CNN)	MKrum	0.8021	0.0513	0.0557	0.0519	0.6983	0.7871
	Median	0.7216	0.5345	0.6021	0.5802	0.6490	0.6914
	Zeno	0.8014	0.0557	0.0557	0.0557	**0.7910**	0.7952
	FLTrust	0.7986	0.0513	0.0513	0.0513	0.7909	0.7812
	DnC	**0.8031**	0.0524	0.0524	0.0524	0.7899	0.0031
	HeteroFL	0.8024	**0.7914**	**0.8001**	**0.7928**	0.7908	**0.7965**
MNIST (LR)	MKrum	0.8201	0.2215	0.4501	0.4769	0.6851	**0.8200**
	Median	0.8007	0.7106	0.7328	0.7521	0.4982	0.7296
	Zeno	0.8211	0.6052	0.6344	0.6590	0.7570	0.8193
	FLTrust	0.8203	0.8109	0.8105	**0.8183**	**0.8145**	0.8192
	DnC	0.8210	0.7823	0.7967	0.7934	0.5957	0.0221
	HeteroFL	**0.8216**	**0.8110**	**0.8169**	0.8132	0.8143	**0.8200**
Shakespeare (RNN)	MKrum	0.5234	0.2345	0.2175	0.2009	0.5019	0.5187
	Median	0.5124	0.4931	0.4899	0.4787	0.5144	0.4897
	Zeno	**0.5235**	0.2115	0.1705	0.2516	**0.5231**	0.5199
	FLTrust	0.5214	0.5108	**0.5191**	0.5162	0.4690	0.4365
	DnC	0.5229	0.5118	0.5156	0.5177	0.5186	0.5011
	HeteroFL	0.5218	**0.5128**	0.5114	**0.5186**	0.5151	**0.5200**

datasets, the accuracy of the other defenses is considerably reduced after performing the adaptive attack, while the accuracy of HeteroFL is barely reduced. This is because HeteroFL can effectively remove most malicious updates. Then a few malicious updates escaped from our detection only have little impact on the global model and are further rectified by the corrective aggregation strategy.

Efficiency: Our method involves simple computation operations that can be easily handled by a FL server typically with powerful computational resources. Initially, the server provides clients with a trigger dataset of 4 samples, incurring minimal communication costs before FL training without overhead during training. Next, the first step involves computing pair similarities between updates. This operation is computationally efficient and widely accepted in the literature (e.g., Multi-Krum, FoolsGold, Bulyan). The second step utilizes a clustering-then-grouping approach on scalar pair similarities. Scalar computations are typically executed at a very high speed, often measured in nanoseconds or even picoseconds. The third step only requires computing the loss values for the 4 samples. Calculating the loss for a few samples can be completed in milliseconds or less. Finally, in the fourth step, computing the norm of updates is a standard operation in the literature and machine learning systems, and it is not time-consuming. To gain an intuitive insight into the efficiency, Fig. 5 compares the testing loss of the global model for HeteroFL over the CIFAR-10 dataset under attacks and no attacks, HeteroFL can converge as fast as the baseline, which indicates that HeteroFL does not incur heavy communication costs to clients. Table II shows that our method does not impose heavy computational overhead compared to the baseline FedAvg and achieves comparable computation time with existing methods.

Privacy: HeteroFL utilizes a randomly generated dataset to construct the trigger dataset, which does not require any private information about clients' local training datasets. Hence, HeteroFL does not undermine the principle of privacy protection.



(a) The global model testing loss when HeteroFL is under all attacks

(b) The global model testing loss of different defenses under no attacks

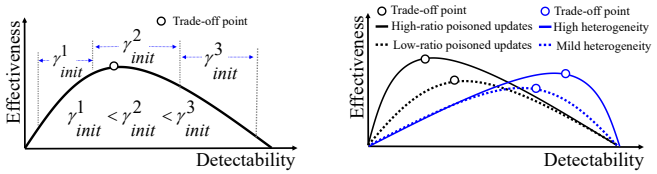
Fig. 5: The global model testing loss on CIFAR-10

TABLE II: Comparison of time-to-accuracy under different settings. We report the time-to-accuracy, which is the time for a system to train to a target accuracy (TA). We measure the time-to-accuracy in hours (h)

Method	MNIST, LR	CIFAR-10, ResNet20	Shakespeare, RNN	FEMNIST, CNN
	TA (80%)	(70%)	TA (50%)	TA (70%)
FedAvg	0.0883h	55.0419h	7.6178h	0.3353h
Ours	0.1869h	71.1811h	17.8122h	0.8872h
Median	0.1422h	58.6653h	17.1472h	0.6100h
Zeno	0.1683h	68.8683h	17.5297h	0.6714h
FLTrust	0.2011h	98.2744h	19.7017h	0.9719h
DnC	0.2378h	75.4025h	19.2150h	0.9369h
Multi-krum	0.2422h	71.8450h	18.0239h	0.9044h

C. Trade-off Between Effectiveness and Detectability of Adaptive Attack

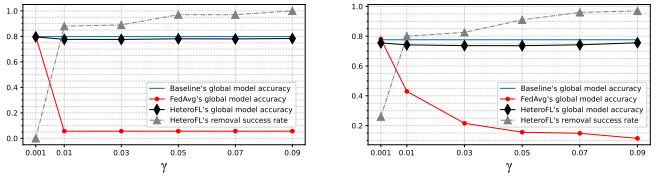
There exists a trade-off between attack effectiveness and detectability given a defense method. The stealthier the poisoned models are, the less effective they will be. Hence, it is necessary to demonstrate where the trade-off point is and how it impacts the final global model. Next, we will show the



(a) The impact of γ_{init} on the trade-off.

(b) The impact of heterogeneity and poisoned ratio on the trade-off.

Fig. 6: The attack effectiveness-detectability continuum. A trade-off must be made between the effectiveness of the attacks and how easy they are to detect.



(a) EMNIST

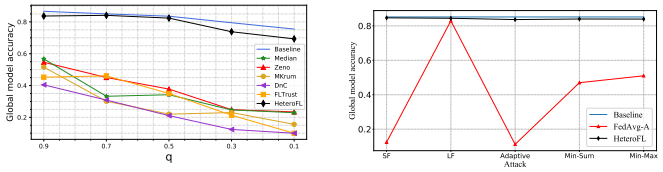
(b) CIFAR-10

Fig. 7: Adaptive attack using perturbation $\widetilde{\boldsymbol{\Delta}}_{std}$. The removal success rate is the average of removal success rates in all rounds.

adaptive attack with the following three factors that mainly affect the trade-off:

Impact of γ_{init} on the trade-off: From Algorithm 7, it is easy to know that the most effective γ should be in the range $[0.5\gamma_{init}, 1.5\gamma_{init}]$, where a large initial γ leads to a large solution γ and makes the adaptive attack more detectable. We use Fig. 6(a) to simply illustrate the situation that different γ_{init} can bring a different range of effectiveness and detectability. The experimental results are shown in Fig. 7, when the detectability decreases (the initial γ reduced from 0.09 to 0.001), the adaptive attack can gradually break through HeteroFL, and its trade-off points on HeteroFL occur at $\gamma_{init} = 0.03$. However, the best effectiveness only leads to a negligible impact on HeteroFL. Besides, even if the adversary changes the initial $\widetilde{\Delta}$ to be $\widetilde{\Delta}_{uv}$ or $\widetilde{\Delta}_{sgn}$ over four heterogeneous datasets, HeteroFL can also not remove all the poisoned updates. However, they are still insufficient to poison our HeteroFL in the trade-off point. Note that we leave the experimental results on all datasets to the supplementary.

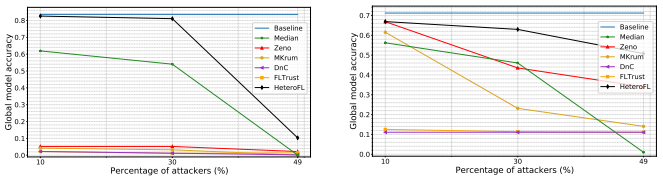
Impact of data heterogeneity (including IID setting) on the trade-off: As discussed in Remark 2, a larger legitimate direction space resulting from the higher heterogeneous degree could make the adaptive attack generate more threatening poisoned models. Fig. 6(b) roughly depicts the trade-off situations under different heterogeneous degrees. It would be interesting to show how much legitimate direction region HeteroFL can restrict when it is enlarged by the heterogeneous data distribution. We evaluate the impact of heterogeneous degree on the CIFAR-10 dataset, considering 25% attackers under adaptive attack with $\gamma_{init} = 0.03$. The experimental results in Fig. 8(a) show that when the heterogeneous degree is increased (q decreases from 0.9 to 0.1), all the defenses suffer from increasing drops in accuracy, which indicates higher heterogeneity makes the attack more effective at the



(a) Impact of heterogeneous degree. A smaller q has a higher heterogeneous degree.

(b) The global model accuracy of HeteroFL in homogeneous setting under all attacks.

Fig. 8: Impact of homogeneous and heterogeneous setting on CIFAR-10



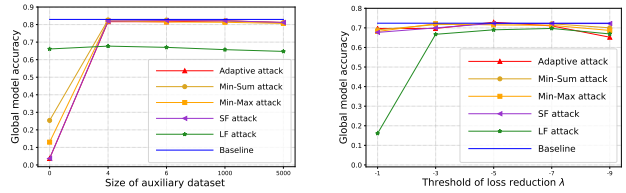
(a) EMNIST

(b) CIFAR-10

Fig. 9: Impact of the percentage of attackers on the accuracy of global model

trade-off point. HeteroFL has the smallest drop in accuracy, being acceptable, which shows HeteroFL largely restricts the legitimate direction space to a benign region. In addition, HeteroFL works also well in the homogeneous setting when facing various attacks, as shown in Fig. 8(b).

Impact of the percentage of attackers on the trade-off: It is obvious that using more poisoned updates, the adaptive attack can construct poisoned updates that slightly deviate from the benign ones, thus being less detectable. As shown in Fig. 6(b), more poisoned updates move the trade-off point to a more stealthy position. To evaluate this scenario, we set $q = 0.2$ and use the most effective γ_{init} for all datasets. Fig. 9 illustrates the accuracy of the existing defenses against adaptive attack when the percentage of attackers varies from 10% to 49%. We can see that high-ratio attackers significantly improve the attack effectiveness on all defenses, *i.e.*, the trade-off points roughly occur in 10% percentage for the other defenses, and 30% percentage for our HeteroFL.



(a) Impact of the trigger dataset size $|D_{trg}|$ in watermark verification

(b) Impact of the loss reduction threshold λ in watermark verification

Fig. 10: Impact of different parameters on HeteroFL under different attacks

D. Ablation Studies on HeteroFL

HeteroFL consists of four core defense steps: sybil direction removal (step 1), similarity detection (step 2), watermark verification (step 3), and magnitude rectification (step 4). We will keep stacking each step onto FedAvg and evaluate the

TABLE III: Ablation study of HeteroFL over CIFAR-10 dataset. Note that we report the global model testing accuracy (%), denoted as *ACC*, and the average removal success rate on all rounds (%), denoted as *ARSR*.

Settings	Adaptive attack		Min-Max attack		Min-Sum attack		LF attack		SF attack	
	ACC	*ARSR*	*ACC*	*ARSR*	*ACC*	*ARSR*	*ACC*	*ARSR*	*ACC*	*ARSR*
FedAvg	0.00	0.00	17.75	0.00	19.30	0.00	67.27	0.00	0.00	0.00
Step 1	0.00	0.05	19.19	0.72	23.67	0.98	67.48	0.97	0.00	1.45
Steps 1-2	57.31	0.12	60.28	73.51	64.50	79.92	70.76	83.52	71.03	100.00
Steps 1-3	69.97	0.14	71.14	88.42	70.82	87.64	70.85	84.80	71.18	100.00
Steps 1-4	71.23	0.16	72.01	89.53	71.33	88.47	71.80	85.06	71.20	100.00
Baseline	72.75	-	72.75	-	72.75	-	72.75	-	72.75	-

resulting defense variants (*i.e.*, step 1, steps 1-2, and steps 1-3) against the same attacks.

Table III compares FedAvg, the three variants *w.r.t.* the global model accuracy and the removal success rates under different attacks. Firstly, step 1 is primarily effective in preventing the adversary from utilizing similar poisoned updates to enhance the attacks, but is not effective against dissimilar poisoned updates. When FedAvg absorbs step 1, the Min-Max and Min-Sum attacks require uploading dissimilar updates, resulting in step 1 having higher ACC even under similar ARSRs compared to FedAvg. Secondly, step 2 can easily detect simple attacks (*e.g.*, LF and SF attacks), but its ability to mitigate more sophisticated attacks (*e.g.*, adaptive attacks) is moderate. Thirdly, step 3 can spot more sophisticated attacks but is ineffective against the simple LF attack. Finally, step 4 can rectify the remaining weakly poisoned gradients from all attacks. Steps 1-4 (*i.e.*, HeteroFL) show no ARSR improvement over steps 1-3 while achieving similar ACC compared to the Baseline.

Overall, HeteroFL combines the strengths of the four complementary steps and thus keeps the ACC high for all levels of attack complexity. This shows the necessity and benefits of integrating the four steps, as they effectively overcome the limitation of one-step defense in existing methods like Multi-Krum, Median, and FLTrust, which cannot fully monitor the entire survival space for the adversaries to construct poisoned gradients. In contrast, our four steps provide complementary monitoring across the entire space, effectively addressing this limitation and enhancing the overall defense mechanism.

E. Evaluating Key Parameters in HeteroFL

HeteroFL only contains two empirical parameters β and λ. β can be set as a small value to select the minorities like existing works did. Next, we will investigate how other key parameters λ, and $|D_{trg}|$ impact each step in HeteroFL.

Impact of the trigger dataset size $|D_{trg}|$: Our watermark verification relies on an trigger dataset that is completely irrelevant with client private datasets. Fig. 10(b) shows the impact of size of trigger dataset on the global model accuracy under five attacks on MNIST. We observe that our watermark verification is sensitive to the model poisoning attacks. With only 4 synthetic samples, our watermark verification can achieve similar accuracy with the baseline. Using only 4 samples can achieve the high sensitivity against attacks since the samples are directly learned by the local models.

Impact of the loss reduction threshold λ: As shown in Fig. 10(c), when λ decreases from -1 to -3, the LF attack can result in a sharp reduction on the global model accuracy. This is because the watermark verification is ineffective against data poisoning attack such that a too large λ can remove many benign updates, which increases the proportion of malicious updates in an aggregation. We show that when $\lambda = -5$, the watermark verification can achieve the similar accuracy with the baseline.

VIII. LIMITATION

Defending poisoning attacks over ciphertext gradients. Despite our substantial strides in mitigating poisoning attacks in challenging highly non-IID scenarios, we acknowledge the presence of more complex situations that require attention. One such situation is defending against poisoning attacks over ciphertext gradients. Currently, our approach primarily centers around traditional research topics [2], [3], [5], [28], [33] that address poisoning attacks over plaintext gradients, which may not be inherently applicable to encrypted data. Effectively adapting these techniques to handle encrypted gradients would necessitate special design considerations. In our ongoing research, we are dedicated to investigating and developing techniques that can effectively and securely handle encrypted gradients within highly non-IID scenarios.

IX. CONCLUSIONS

We analyzed the failure of existing methods and the effect of poisoning attacks. We then proposed HeteroFL, a new defense that is suitable for heterogeneous settings. Our evaluations on four datasets demonstrated that HeteroFL outperforms state-of-the-art defense methods and can defeat all poisoning attacks.

REFERENCES

[1] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS'20)*, 2020, pp. 2938–2948.

[2] G. Baruch, M. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," in *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems (NIPS'19)*, 2019, pp. 8632–8642.

[3] P. Blanchard, E. M. E. Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proceedings of the 30th Annual Conference on Neural Information Processing Systems (NIPS'17)*, 2017, pp. 119–129.

[4] D. Byrd and A. Polychroniadou, "Differentially private secure multi-party computation for federated learning in financial applications," *CoRR*, vol. abs/2010.05867, 2020. [Online]. Available: https://arxiv.org/abs/2010.05867

[5] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," in *Proceedings of the 28th Annual Network and Distributed System Security Symposium (NDSS'21)*, 2021.

[6] X. Cao, J. Jia, Z. Zhang, and N. Z. Gong, "Fedrecover: Recovering from poisoning attacks in federated learning using historical information," in *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2022, pp. 326–343.

[7] X. Cao and L. Lai, "Distributed gradient descent algorithm robust to an arbitrary number of byzantine attackers," *IEEE Trans. Signal Process.*, vol. 67, no. 22, pp. 5850–5864, 2019.

[8] L.-Y. Chen, T.-C. Chiu, A.-C. Pang, and L.-C. Cheng, "Fedequal: Defending model poisoning attacks in heterogeneous federated learning," in *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2021, pp. 1–6.

[9] X. Chen, H. Yu, X. Jia, and X. Yu, "Apfed: Anti-poisoning attacks in privacy-preserving heterogeneous federated learning," *IEEE Transactions on Information Forensics and Security*, 2023.

[10] T. Chu, Á. García-Recuero, C. Iordanou, G. Smaragdakis, N. Laoutaris *et al.*, "Securing federated sensitive topic classification against poisoning attacks," in *Usenix Network and Distributed System Security Symposium*, 2023.

[11] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, "EMNIST: an extension of MNIST to handwritten letters," *CoRR*, vol. abs/1702.05373, 2017. [Online]. Available: http://arxiv.org/abs/1702.05373

[12] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[13] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to byzantine-robust federated learning," in *Proceedings of the 29th USENIX Security Symposium (USENIX Security'20)*, 2020, pp. 1605–1622.

[14] C. Fung, C. J. Yoon, and I. Beschastnikh, "The limitations of federated learning in sybil settings," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, 2020, pp. 301–316.

[15] L. Haoyang, Q. Ye, H. Hu, J. Li, L. Wang, C. Fang, and J. Shi, "3dfed: Adaptive and extensible framework for covert backdoor attack in federated learning," in *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2023, pp. 1893–1907.

[16] C. He, S. Li, J. So, X. Zeng, M. Zhang, H. Wang, X. Wang, P. Vepakomma, A. Singh, H. Qiu *et al.*, "Fedml: A research library and benchmark for federated machine learning," *arXiv preprint arXiv:2007.13518*, 2020.

[17] J. Jiang, B. Kantarci, S. F. Oktug, and T. Soyata, "Federated learning in smart city sensing: Challenges and opportunities," *Sensors*, vol. 20, no. 21, p. 6230, 2020.

[18] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[20] J. Li, R. Ji, H. Liu, X. Hong, Y. Gao, and Q. Tian, "Universal perturbation attack against image retrieval," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4899–4908.

[21] 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 *Proceedings of the 23rd AAAI Conference on Artificial Intelligence, (AAAI'19)*, 2019, pp. 1544–1551.

[22] S. Li, Y. Cheng, Y. Liu, W. Wang, and T. Chen, "Abnormal client behavior detection in federated learning," *CoRR*, vol. abs/1910.09933, 2019.

[23] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *arXiv preprint arXiv:1812.06127*, 2018.

[24] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS'17)*, 2017, pp. 1273–1282.

[25] H. B. McMahan and D. Ramage, "Federated learning: Collaborative machine learning without centralized training data," 2017.

[26] E. M. E. Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in byzantium," in *Proceedings of the 35th International Conference on Machine Learning (ICML'18)*, 2018, pp. 3518–3527.

[27] H. Mozaffari, V. Shejwalkar, and A. Houmansadr, "Every vote counts: Ranking-based training of federated learning to resist poisoning attacks," in *USENIX Security Symposium*, 2023.

[28] L. Muñoz-González, K. T. Co, and E. C. Lupu, "Byzantine-robust federated machine learning through adaptive model averaging," *CoRR*, vol. abs/1909.05125, 2019.

[29] T. D. Nguyen, P. Rieger, H. Chen, H. Yalame, H. Möllering, H. Fereidooni, S. Marchal, M. Miettinen, A. Mirhoseini, S. Zeitouni *et al.*, "{FLAME}: Taming backdoors in federated learning," in *31st USENIX Security Symposium (USENIX Security 22)*, 2022, pp. 1415–1432.

[30] V. K. Pillutla, S. M. Kakade, and Z. Harchaoui, "Robust aggregation for federated learning," *CoRR*, vol. abs/1912.13445, 2019.

[31] S. Prakash and A. S. Avestimehr, "Mitigating byzantine attacks in federated learning," *CoRR*, vol. abs/2010.07541, 2020.

[32] R. Ramachandra and C. Busch, "Presentation attack detection methods for face recognition systems: A comprehensive survey," *ACM Computing Surveys (CSUR)*, vol. 50, no. 1, pp. 1–37, 2017.

[33] V. Shejwalkar and A. Houmansadr, "Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning," in *Proceedings of the 28th Annual Network and Distributed System Security Symposium (NDSS'21)*, 2021.

[34] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *Proceedings of the 25th European Symposium on Research in Computer Security (ESORICS'20)*, 2020, pp. 480–501.

[35] Q. Xia, Z. Tao, Z. Hao, and Q. Li, "FABA: an algorithm for fast aggregation against byzantine attacks in distributed neural networks," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19)*, 2019, pp. 4824–4830.

[36] C. Xie, K. Huang, P. Chen, and B. Li, "Dba: Distributed backdoor attacks against federated learning," in *Proceedings of the 8th International Conference on Learning Representations (ICLR'20)*, 2020.

[37] C. Xie, O. Koyejo, and I. Gupta, "Generalized byzantine-tolerant SGD," *CoRR*, vol. abs/1802.10116, 2018.

[38] C. Xie, S. Koyejo, and I. Gupta, "Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance," in *Proceedings of the 36th International Conference on Machine Learning (ICML'19)*, 2019, pp. 6893–6901.

[39] Y. Xie, W. Zhang, R. Pi, F. Wu, Q. Chen, X. Xie, and S. Kim, "Robust federated learning against both data heterogeneity and poisoning attack via aggregation optimization," *arXiv preprint*, 2022.

[40] J. Xu, B. S. Glicksberg, C. Su, P. B. Walker, J. Bian, and F. Wang, "Federated learning for healthcare informatics," *J. Heal. Informatics Res.*, vol. 5, no. 1, pp. 1–19, 2021.

[41] D. Yin, Y. Chen, K. Ramchandran, and P. L. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proceedings of the 35th International Conference on Machine Learning (ICML'18)*, 2018, pp. 5636–5645.

[42] Z. Zhang, Y. Zhang, D. Guo, S. Zhao, and X. Zhu, "Communication-efficient federated continual learning for distributed learning system with non-iid data," *Sci. China Inf. Sci.*, vol. 66, no. 2, 2023.

[43] B. Zhao, P. Sun, T. Wang, and K. Jiang, "Fedinv: Byzantine-robust federated learning by inversing local model updates," 2022.

[44] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

Jianrong Lu received the M.S degree in cyberspace security from Huazhong University of Science and Technology, Wuhan, China, in 2023. He is currently a Research Assistant at City University of Hong Kong. His research interests include federated learning, optimization, and AI for science.

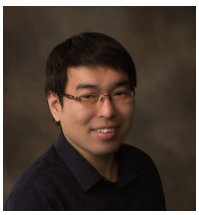
Shengshan Hu received the B.E. and Ph.D. degrees in computer science and technology from Wuhan University in 2014 and 2019, respectively. He is currently an Associate Professor in the School of Cyber Science and Engineering, Huazhong University of Science and Technology. His research interests include the security and privacy of intelligent systems such as automatic driving, smart medicine, and federated learning, *etc.* He has published more than 30 papers in international conferences and journals such as ACM CCS, CVPR, IJCAI, and IEEE TIFS.

Wei Wan received the M.S degree from the School of Software Engineering at Huazhong University of Science and Technology, Wuhan, China, in 2021. He is currently pursuing the Ph.D. degree at the School of Cyber Science and Engineering at Huazhong University of Science and Technology, Wuhan, China. His research interests include the security mechanisms of federated learning.

Minghui Li received her Ph.D. degree from the School of Cyber Science and Engineering at Wuhan University in 2021. She is currently an Assistant Professor in the School of Software Engineering at Huazhong University of Science and Technology. Her research interests focus on the security and privacy of artificial intelligent systems. She has published over 10 papers in international conferences and journals, including AAAI, CVPR, ACM MM, INFOCOM, TDSC, and TIFS, *etc.*

Leo Yu Zhang (M'17) is currently a Senior Lecturer with the School of Information and Communication Technology, Griffith University, QLD, Australia. And he used to be a Lecturer at the School of Information Technology, Deakin University from 2018-2022. He received the bachelor's and master's degrees in computational mathematics from Xiangtan University, Xiangtan, China, in 2009 and 2012, respectively, and the Ph.D. degree from the City University of Hong Kong, Hong Kong, in 2016. He held various research positions with the City University of Hong Kong, the University of Macau, Macau, China, the University of Ferrara, Ferrara, Italy, and the University of Bologna, Bologna, Italy. His current research interests include trustworthy AI and applied cryptography, and he has published more than 90 refereed journal and conference articles in these fields.

Lulu Xue is currently pursuing the M.S. degree in cyberspace security in Huazhong University of Science and Technology. Her research interests include federal learning and privacy security.



Haohan Wang is an assistant professor in the School of Information Sciences at the University of Illinois Urbana-Champaign. His research focuses on the development of trustworthy machine learning methods for computational biology and healthcare applications, such as decoding the genomic language of Alzheimer's disease. In his work, he uses statistical analysis and deep learning methods, with an emphasis on data analysis using methods least influenced by spurious signals. Wang earned his PhD in computer science through the Language Technologies Institute of Carnegie Mellon University where he works with Professor Eric Xing. In 2019, Wang was recognized as the Next Generation in Biomedicine by the Broad Institute of MIT and Harvard because of his contributions in dealing with confounding factors with deep learning.

Hai Jin is a Chair Professor of computer science and engineering at Huazhong University of Science and Technology (HUST) in China. Jin received his PhD in computer engineering from HUST in 1994. In 1996, he was awarded a German Academic Exchange Service fellowship to visit the Technical University of Chemnitz in Germany. Jin worked at The University of Hong Kong between 1998 and 2000, and as a visiting scholar at the University of Southern California between 1999 and 2000. He was awarded Excellent Youth Award from the National Science Foundation of China in 2001. Jin is a Fellow of IEEE, Fellow of CCF, and a life member of the ACM. He has co-authored more than 20 books and published over 900 research papers. His research interests include computer architecture, parallel and distributed computing, big data processing, data storage, and system security.