Secure Model Aggregation Against Poisoning Attacks for Cross-Silo Federated Learning With Robustness and Fairness

Yunlong Mao, *Member, IEEE*, Zhujing Ye, *Student Member, IEEE*, Xinyu Yuan, and Sheng Zhong, *Fellow, IEEE*

***Abstract*— Federated learning (FL) is a promising approach for participants' collaborative learning tasks with cross-silo data. Participants benefit from FL since heterogeneous data can contribute to the generalization of the global model while keeping private data locally. However, practical issues of FL, such as security and fairness, keep emerging, impeding its further development. One of the most threatening security issues is the poisoning attack, corrupting the global model by an adversary's will. Recent studies have demonstrated that elaborate model poisoning attacks can breach the existing Byzantine-robust FL solutions. Although various defenses have been proposed to mitigate poisoning attacks, participants will sacrifice learning performance and fairness due to strict regulations. Considering that the importance of fairness is no less than security, it is crucial to explore alternative solutions that can secure FL while ensuring both robustness and fairness. This paper introduces a robust and fair model aggregation solution, Romoa-AFL, for cross-silo FL in an agnostic data setting. Unlike a previous study named Romoa and other similarity-based solutions, Romoa-AFL ensures robustness against poisoning attacks and learning fairness in agnostic FL, which has no assumptions of participants' data distributions and the server's auxiliary dataset.**

***Index Terms*— Federated learning, poisoning attack, robustness, fairness, secure model aggregation.**

I. Introduction

The breakthrough of deep neural networks (DNNs) relies heavily on substantial training data and computing resources. As the data volume grows rapidly, gathering data from users to a central server becomes inefficient. Meanwhile, collecting users' private data may pose a number of privacy risks [2]. With its emphasis on privacy, federated learning (FL) [3], [4] has quickly gained recognition as a groundbreaking paradigm, allowing for collaborative learning while preserving the confidentiality of participants' local data. Unfortunately, collaboratively training DNNs in FL creates a new attack surface [5], [6], [7].

Of all the potential risks to consider within the realm of FL, the poisoning attack [7] emerges as a particularly severe threat. By corrupting the global model of FL, adversaries can deliberately introduce manipulations that undermine the accuracy of the classifications. Depending on whether the poisoning attack is targeted [8] or untargeted [6], the misclassified input can consist of specific or arbitrary samples. Recognizing the severity of poisoning attacks, extensive efforts have been made to counter their adversarial effects. However, the pursuit of practical and effective solutions continues to be a subject of active discussion. Several robust model aggregation methods [9], [10] have been demonstrated to be effective against model poisoning attacks. However, emerging research [6], [11] has uncovered vulnerabilities in the integration of Byzantine-robust aggregation solutions in FL, indicating that poisoning attacks can exploit improper integration to compromise the integrity of the collaborative learning process.

Meanwhile, solutions [6], [9], [10] for secure model aggregation against poisoning attacks heavily rely on member selection, which grants only a small fraction of participants (sometimes just one participant) the privilege to contribute to the global model in each iteration. This poses risks to individual fairness [12], [13], [14] as it potentially wastes the training efforts of unselected participants. Furthermore, the emergence of new strategies for model poisoning attacks presents an ongoing challenge. Given the rapid evolution of these threats, the existing defense research struggles to keep pace. Consequently, an alternative robust solution for secure FL model aggregation becomes crucial to ensure security and fairness.

However, coming up with such a defense solution is quietly challenging. A primary challenge is accurately identifying poisoning attacks from diverse learning states. FL employs randomization techniques like stochastic gradient descent (SGD) and dropout, making it difficult to differentiate attacks from normal training fluctuations. This challenge is magnified when participants' data is non-independent and identically distributed (non-IID). Additionally, the adversary could conceal the attack by reducing the degree of model manipulation or performing the attack opportunistically [11]. Hence, false alarms are inevitable when using rigorous detection solutions. Otherwise, the attacker will go undetected if lenient detecting

Manuscript received 2 November 2023; revised 31 March 2024; accepted 12 June 2024. Date of publication 17 June 2024; date of current version 25 June 2024. This work was supported in part by the National Natural Science Foundation of China under Grant NSFC-62272222, Grant NSFC-61902176, and Grant NSFC-62272215; and in part by Jiangsu Natural Science Foundation Excellent Youth Project under Grant BK20230080. An earlier version of this paper was presented at the Proceedings of the 26th European Symposium on Research in Computer Security (ESORICS) 2021 [DOI: 10.1007/978-3-030-88418-5_23]. The associate editor coordinating the review of this article and approving it for publication was Prof. Mark Manulis. *(Corresponding author: Sheng Zhong.)*

Yunlong Mao, Zhujing Ye, and Sheng Zhong are with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China (e-mail: sheng.zhong@gmail.com).

Xinyu Yuan was with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China. She is now with Alibaba, Hangzhou 310052, China.

Digital Object Identifier 10.1109/TIFS.2024.3416042

strategies are applied. Furthermore, handling suspicious participants is also a tricky problem. The adversarial impact can be avoided by aggregating data over a small group of participants. However, a significant amount of learned knowledge will be discarded. In the end, participants cannot make fair contributions to the global model.

To defend against poisoning attacks in FL, a defense solution named Romoa was introduced in previous work [1], based on similarity measurements. Although related studies [15], [16] have also employed similarity metrics to detect poisoning attacks, Romoa gives the first attempt to use a hybrid similarity measurement in a lookahead way to identify adversarial behaviors accurately and timely. Recent work [17] has highlighted that FL has a natural weakness in achieving fair learning. This weakness stems from the presence of unbalanced data distributions among FL participants. The commonly employed FL aggregation strategies tend to assign weights to users' data samples based on their frequency of occurrence. Consequently, participants with limited or rare samples may face challenges in making significant contributions to the FL process. Fair FL solutions tackle the problem by using data augmentation or reweighting methods [18], [19] to improve feature or individual fairness. However, there is a natural conflict between fairness and robustness. Fair FL methods may boost the influence of poisoning attacks since poisoning data is relatively rare compared with global training data. Poisoning attackers can leverage the chance. On the other side, robust FL solutions commonly use prior knowledge of data distributions to detect poisoning data. When data distributions are unknown or changeable, as assumed in fair FL, the detection result may be frustrated. Therefore, providing robustness and fairness simultaneously in a unified FL solution is rather challenging.

Therefore, we recommend modeling cross-silo FL with latent poisoning attacks using agnostic federated learning (AFL) [20], which assumes agnostic data distributions. In this case, data distributions' prior knowledge and auxiliary datasets are unavailable. Although fair FL studies such as [20] and robust FL studies such as [1] have provided basics of fairness and robustness separately, a unified implementation of secure and fair FL aggregation for agnostic data distributions is still an open problem. Inspired by Romoa, we introduce a hybrid similarity measurement into AFL to identify poisoning attacks. Nevertheless, similarity-based solutions are incompatible with heterogeneous data (non-IID data), as they often result in a high false alarm rate. To overcome this problem, we design Romoa-AFL, a robust model aggregation solution for AFL, using an adaptive similarity measurement. We also introduce a model complexity constraint using the measurement result to restrict the learning process for the fairness concern. Overall, we make the following contributions:

- We remove the assumption of data distributions' prior knowledge in FL and implement Romoa-AFL as a countermeasure against poisoning attacks (targeted, untargeted, and fake client attacks), resolving the conflict between robustness and fairness.
- We use a game-based analysis mothed to show the robustness of Romoa-AFL can be ensured by the existence of a Nash equilibrium. We give the fairness analysis by showing that Romoa-AFL can provide the required learning guarantees such that the theoretical fairness result given in [20] still holds.
- We comprehensively evaluate Romoa-AFL using standard image classification datasets under different poisoning attacks. In terms of robustness and fairness, we compare Romoa-AFL with abundant defenses. Experimental results confirm that Romoa-AFL can mitigate single-point and collusive poisoning attacks. Furthermore, Romoa-AFL exhibits better performance when dealing with non-IID data and large-scale FL.

II. Related Work

A. Attacks Against Federated Learning

Poisoning attacks manipulate the learning process by exploiting compromised clients. Baruch et al. [36] proposed a powerful attack strategy leveraging empirical variance in gradients to evade defense strategies. Fang et al. [6] conducted a systematic study on local model poisoning attacks, formulating it as an optimization problem. Sun et al. [37] extended it to federated multi-task learning scenarios. Shejwalkar and Houmansadr [7] developed the AGG-MM optimization framework for different settings, considering the knowledge possessed by the adversary at the client and server levels. Cao and Gong [38] explored a realistic model poisoning attack by injecting fake clients into federated learning. Li et al. [39] introduced a reinforcement learning attack framework that adapts to scenarios without prior knowledge of aggregated data distribution, using model updates to approximate the data distribution and learn an adaptive attack strategy.

The backdoor attack [40], [41] in FL is a special kind of poisoning attack. Xie et al. [42] proposed a distributed backdoor attack to enhance persistence. Then, Ning et al. [43] transformed the backdoor trigger into a noise trigger, successfully deceiving detection schemes. Lyu et al. [44] extended the divergence of poisoned local models through distributed backdoor attacks, effectively providing covert backdoor attacks and evading defense strategies in FL.

There are also attacks mounted by FL servers. Fowl et al. [45] demonstrated that a semi-honest server can reverse participant gradients. Lam et al. [46] showed that an untrusted server can recover private training data through gradient inference. Wang et al. [47] proposed an invisible server-side framework combining GANs with multi-task discriminators, allowing the recovery of user-specified data. Pasquini et al. [48] highlighted that malicious servers can bypass secure aggregation algorithms and infer private training data.

B. Defenses Against Attacks in Federated Learning

Having realized the threat of poisoning attacks, researchers propose many insightful defense solutions. A common way to protect the global model from poisoning attacks is to find robust estimators of gradients. Yin et al. [10] pioneered the use of the coordinate-wise median and trimmed-mean as robust estimators within an iteration. Similarly, Alistarch et al. [49]

TABLE I

COMPARISON OF RELATED SOLUTIONS FOR SECURE FL MODEL AGGREGATION

	robustness	fairness	malicious parties	collusion ratio	TPA defense	UPA defense	agnostic FL	no auxiliary data
Median [10]	✓	✗	client	0.5	✗	✓	✗	✓
RFA [21]	✓	✗	client	0.25	✓	✓	✓	✓
Krum [22]	✓	✗	client	$<$0.5	✗	✓	✗	✓
Contra [16]	✓	✗	client	0.5	✓	✓	✓	✓
P2Brof [23]	✓	✗	client/server	0.5	✗	✓	✗	✓
Zeno [24]	✓	✗	client	0.12	✗	✓	✗	✓
Zeno++ [25]	✓	✗	client	0.8	✗	✓	✗	✗
FLTrust [26]	✓	✗	client	0.1	✓	✓	✗	✗
SageFlow [27]	✓	✗	client	0.2	✓	✓	✗	✗
Bulyan [28]	✓	✗	client	0.25	✗	✓	✗	✓
FedDefender [29]	✓	✗	client	0.2	✓	✓	✓	✓
FLDetector [30]	✓	✗	client	0.28	✓	✓	✓	✓
DnC [7]	✓	✗	client	0.25	✗	✓	✓	✓
BREA [31]	✓	✗	client	0.1	✗	✓	✓	✓
FLGuard [32]	✓	✗	client	0.1	✓	✓	✗	✓
RoFL [33]	✓	✗	client	0.05	✓	✗	✗	✓
pMPL [34]	✓	✗	client	$<$0.5	✗	✓	✗	✓
DP-Byt [35]	✓	✗	client/server	0.6	✗	✓	✓	✗
FairAvg [17]	✓	✓	client	-	✗	✗	✓	✓
AFL [20]	✗	✓	client	-	✗	✗	✓	✓
Romoa [1]	✗	✓	client	0.5	✓	✓	✗	✓
Romoa-AFL	✔	✔	**client**	**0.5**	✔	✔	✔	✔

designed an optimally robust algorithm that utilizes geometric median to average stochastic gradients across iterations. Yin et al. [50] subsequently developed ByzantinePGD using median, trimmed mean, and iterative filtering estimators to resist the saddle points and fake local minima attacks. For the case where the compromised level is low, Pillutla et al. [21] proposed a robust aggregation algorithm based on a geometric median estimator, which is competitive with the average algorithm under a small fraction of corrupted devices.

In the field of similarity-based defense, the Krum approach [22] is widely used for selecting the global model, minimizing the impact of poisoning from compromised devices. A cosine similarity-based measurement [16] dynamically reduces the influence of potentially malicious clients by distinguishing between benign and malicious model updates. Dong et al. [23] adopt a similar cosine similarity approach using a three-party computation protocol. Another effective approach, Guerraoui et al. [28] strategically selects non-Byzantine gradient samples to create aggregated gradients. Shejwalkar and Houmansadr [7] argue that distance-based filters alone are insufficient, proposing the DnC method that projects the centered gradient set and removes gradients with high outlier scores. While FLDetector [30] utilizes the Euclidean distance to distinguish between consistent model updates from benign clients and inconsistent updates from malicious clients. It employs the Cauchy mean value theorem to predict updates, assigns dynamic suspicious scores, and utilizes k-means with Gap statistics to effectively detect malicious clients.

Besides secure aggregation algorithms on the server side, Park et al. propose FedDefender [29] to enhance client-side robustness. It incorporates synthetic noise to identify noise-tolerant model parameters and employs intermediate layer distillation to extract valuable knowledge from potentially corrupted global models, improving the local training process.

Assumptions of trusted clients or auxiliary datasets can aid in identifying outliers. Xie et al. [24] use the server's public dataset to detect adversaries, tolerating a large number of poisoning gradients. Zeno++ [25] improves on this approach, addressing worker and communication limitations while still relying on the IID data assumption. For the non-IID scenario, FLTrust [26] introduces trust scores assigned by the parameter server based on cosine similarity, offering security against adaptive attacks. Sageflow [27] utilizes entropy-based filtering and loss-weighted averaging to handle malicious adversaries and address the straggler's problem. Although these methods effectively counter collusion attacks, they all require a secure reference to identify malicious participants.

A recent study [51] provides a novel FL paradigm in an ensemble manner. Multiple global models will be yielded by selecting different groups of updates. Then, a certified security level can be estimated using the ensemble model performance. Since this novel FL paradigm is orthogonal to specific model aggregation strategies (FedAvg is used in [51]), any FL design can benefit from it, deriving a provable security guarantee.

We have summarized a collection of highly pertinent studies on secure model aggregation against poisoning attacks in Table I. The table provides insights into whether these solutions were investigated under specific assumptions, such as collusion ratio[1] and auxiliary data. Based on the information presented in the table, it can be inferred that our Romoa-AFL offers more guarantees under more practical assumptions.

III. PROBLEM STATEMENT

A. Federated Learning

In the original FL [3], [52], a central parameter server (PS) is responsible for coordinating n participants who join the same FL task. For simplicity, we assume that each participant denoted as P_i, where i ranges from 1 to n, holds private training data $\boldsymbol{x}^i$. All participants in the same task use an identical deep neural network (DNN) architecture and learning hyper-parameters. Generally, a mini-batch SGD optimizer is used by P_i to minimize a loss function $\mathcal{L}(\boldsymbol{\theta}^i)$ for model parameters $\boldsymbol{\theta}^i$. To update the local model, the gradient $\nabla_{\boldsymbol{\theta}^i}$

[1] The collusion ratio represents the proportion of Byzantine clients to the total number of clients, which is the breakdown point of the scheme or the maximum value indicated in the published paper.

regarding a batch size m should be estimated as

$$\boldsymbol{g}^i(\boldsymbol{\theta}^i) = \frac{1}{m}\sum\nolimits_{j=1}^{m} \nabla_{\boldsymbol{\theta}^i}\mathcal{L}(\boldsymbol{\theta}^i, x_j),\ \ x_j \in \boldsymbol{x}^i. \tag{1}$$

A global counter $t \in [1, T]$ is maintained by the PS. Given P_i's local gradient $\boldsymbol{g}^i_{t-1}$, model parameter $\boldsymbol{\theta}^i$ in the next iteration should be updated by $\boldsymbol{\theta}^i_t = \boldsymbol{\theta}^i_{t-1} - \eta\boldsymbol{g}^i$, where t indicates training iteration and η is a predefined learning rate. After participants' local training, the PS will perform model aggregation in accordance with a predefined strategy, such as averaging. In this way, the PS gives the model aggregation result as $\bar{\boldsymbol{\theta}}_t = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{\theta}^i_t$. At the $(t+1)$-th iteration, all participants retrieve the updated global model, denoted as $\bar{\boldsymbol{\theta}}_t$, from the PS and proceed to local training. This procedure will be repeated until either the global model has attained the desired level of usability or the maximum training iteration limit has been reached.

B. Model Poisoning Attack

1) Adversarial Capability: Generally, any legal participant in FL can be an adversary poisoning the global model. It has been proved that collusive attacks can promote poisoning attacks significantly [6], [15], [53]. For collusive poisoning, it is commonly assumed that the total amount of adversarial participants should be less than $\lceil n/2 \rceil$. Moreover, the adversary has stealth capability to avoid detection schemes. This stealth capability can be characterized by a stealth factor [11] for model poisoning attacks. Specifically, the adversary can weaken the poisoning impact by weighting the poison with a stealth factor α to avoid detection.

2) Adversarial Goal: The adversarial goal is to compromise the global model so that it behaves abnormally without sabotaging the FL framework. There are two common adversarial goals, targeted poisoning attack [11] and untargeted poisoning attack [6]. An untargeted poisoning attacker seeks to cause misclassification of any input samples indiscriminately. In contrast, a targeted poisoning attacker aims to specifically cause misclassification for targeted input sample(s). Two adversarial goals are frequently discussed separately in the existing defense studies [6], [40], but we will take them into account at the same time and provide a unified defense strategy.

3) Untargeted Poisoning Attack (UPA): In the UPA attack, the adversary has complete knowledge of the compromised client-local training data and model. The UPA attack can evade Byzantine-robust solutions by replacing the local model with a compromised one. In the threat model, the attacker employs the powerful UPA attack method proposed by Fang et al. [6]. This method involves solving an optimization problem to determine the opposite direction of model updates. To illustrate, we consider one adversary, denoted as P_a, and define the objective function of UPA[2] is

$$\begin{aligned}\mathcal{O}^{\mathrm{UPA}} &= \arg\max_{\boldsymbol{\theta}^a} \boldsymbol{s}^T(\bar{\boldsymbol{\theta}} - \bar{\boldsymbol{\theta}}^a),\\ \text{subject to } \bar{\boldsymbol{\theta}} &= \sum\nolimits_{i=1}^{n}\boldsymbol{\theta}^i, \bar{\boldsymbol{\theta}}^a = \boldsymbol{\theta}^a + \sum\nolimits_{i=1, i\neq a}^{n}\boldsymbol{\theta}^i,\end{aligned} \tag{2}$$

where $\boldsymbol{s}^T$ is a changing direction of the global model from the before-attack state $\bar{\boldsymbol{\theta}}$ to the after-attack state $\bar{\boldsymbol{\theta}}^a$.

a) Targeted poisoning attack (TPA): Contrary to UPA, TPA exhibits specific interests in certain data samples [11], [40], [54]. Specifically, the adversary in this context adopts the scale attack method proposed by Bagdasaryan et al. [40]. In this attack, the adversary implants a trigger on a compromised client and assigns a specific target class label chosen by the attacker. Additionally, the local model is multiplied by an amplification factor before transmitting the model updates to the server. This factor balances the influence of the model with the adversary's probability of being detected. We remark that an amplification factor in TPA[3] [11], [40] is omitted here, which will be integrated into the stealth factor later. Assume these samples all in one set $\boldsymbol{x}^{\mathrm{TPA}} = \{x_1, x_2, \ldots, x_r\}$. Given the corresponding labels $\boldsymbol{y}^{\mathrm{TPA}} = \{y_1, y_2, \ldots, y_r\}$, the adversary aims to have each sample $x_i^{\mathrm{TPA}} \in \boldsymbol{x}_{\mathrm{TPA}}$ misclassified as label y_i' by the global model, $y_i' \neq y_i$. Then, the TPA objective function for the adversary P_a is

$$\begin{aligned}\mathcal{O}^{\mathrm{TPA}} &= \arg\min_{\boldsymbol{\theta}^a} \mathcal{L}(\{x_i, y_i'\}_{i=1}^{r}, \bar{\boldsymbol{\theta}}^{\mathrm{a}}),\\ \text{subject to } \bar{\boldsymbol{\theta}}^{\mathrm{a}} &= \boldsymbol{\theta}^a + \sum\nolimits_{i=1, i\neq a}^{n}\boldsymbol{\theta}^i,\end{aligned} \tag{3}$$

where $\mathcal{L}(\cdot)$ is the loss function used in P_a's local training.

b) Fake client attack: The fake client attack refers to a scenario in which a malicious attacker fabricates fake clients to participate in the training process, creating a more realistic situation where the attacker has limited capabilities and no access to real training data. Specifically, we consider an implementation of the fake client attack named MPAF [38], which is a typical technique for poisoning attacks with fake clients, in which the attacker only learns the global model of the FL system without any additional information. In each round of an FL task, fake clients generate malicious local model updates corresponding to a base model and amplify their influence before transmitting them to the server.

c) Adaptive attack: In adaptive attacks, adversaries have a certain knowledge of the target and may utilize this knowledge to devise more effective attack strategies. Specifically, adversaries have a prior knowledge of the target FL system, a thorough grasp of compromised participants' data distribution and model training process, and knowledge of critical information such as aggregation algorithms on the server side. Therefore, we consider a fully capable adversary with a comprehensive understanding of the similarity-based reputation scoring and historical behavior analysis mechanism of Romoa-AFL. It conceals its poisoning behavior during local training of participants, striving to gain a higher trust score from the server, and executes model poisoning during synchronization rounds after multiple training rounds, thereby effectively amplifying the impact of the attack.

d) Stealth factor: Generally, we define the adversary's goal in the t-th iteration as $\mathcal{A}_t = \boldsymbol{\theta}^a_{t-1} - \alpha_t(\boldsymbol{\theta}^a_{t-1} - \mathcal{O}^b_t)$, $b \in \{\mathrm{UPA}, \mathrm{TPA}\}$, $t \in [1, T]$, $\alpha_t \in [0, 1]$, subjecting to the corresponding constraint. When the stealth factor α_t takes

[2] For more details of this approach, please refer to [6]. We use UPA as a general notation to refer to the attack proposed by Fang et al. [6].

[3] We use TPA as a general notation to refer to the scaling attack proposed by Bagdasaryan et al. [40].

a value of 1, the adversary's objective $\mathcal{A}_t$ is to replace the local model with a poisoning model fully. Conversely, when α_t is 0, the adversary chooses not to execute an attack at that particular time. In all other cases, $\mathcal{A}_t$ can be perceived as a combination of the global model and the poison model, denoted by $(1-\alpha_t)\boldsymbol{\theta}^a_{t-1} + \alpha_t \mathcal{O}^b_t$.

C. Data Assumptions for Cross-Silo Federated Learning

Training data from different silos may follow different distributions. Our work aims to secure FL model aggregation under arbitrary data assumptions. To align with prior research, a probability $q \in [0, 1]$ was introduced to regulate the degree of independence of data distribution. Suppose a dataset consists of M classes, and a number of clients are randomly divided into M groups. When distributing the data, the training samples with label $c \in [1, M]$ are allocated to group c with a probability q. It is assumed that the data belonging to clients within the same group are IID. If q equals $\frac{1}{M}$, the training data of each client is expected to be IID. If q equals 1, training data for clients in different groups is clearly non-IID. As q varies within $[0, 1]$, various levels of non-IID data assumptions can be represented.

D. Robustness in Federated Learning

FL is vulnerable to adversarial behaviors since the adversary can introduce fake clients [38] or hijack benign clients [26] and inject elaborately crafted poisons into the global model, eventually compromising the global model. To tackle this issue, Byzantine-robust FL has been widely studied [10], [55], employing robust model aggregation rules to supervise clients' local model updates and eliminate outliers before global aggregation. In this paper, we focus on the robustness of FL, measured by the model's ability to resist multiple adversarial clients' collusion. Moreover, we assume that adversarial clients can mount TPA or UPA separately or cooperatively. We evaluate the robustness of FL using the test error rate of the target label of a poisoning attack. Generally, we say that model $\boldsymbol{\theta}_1$ is more robust than model $\boldsymbol{\theta}_2$ if the former has a lower average test error rate over benign clients, i.e.,

$$\frac{1}{|K|}\sum_{k\in K} e_k(\boldsymbol{\theta}_1) < \frac{1}{|K|}\sum_{k\in K} e_k(\boldsymbol{\theta}_2), \tag{4}$$

where K is the collection of benign client, $e_k(\cdot)$ is the test error rate of the client with index k.

E. Fairness in Federated Learning

Algorithmic bias has been extensively studied in the field of machine learning, where individual fairness and group fairness are two common fairness notions [56]. Individual fairness requires that different individuals be treated similarly, whereas group fairness requires that disadvantaged groups be treated the same as advantaged groups. Due to the divergence of training data and user privacy concerns, the concept of fairness in the context of FL focuses more on collaborative fairness [57]. In the existing FL frameworks [3], [52], regardless of contributions, each client is given the same global model in every round of communication. This is unfair since different clients contribute with data of varying quality. As a result, data from benign clients may lead to positive updates, whereas updates from other malicious clients may degrade the model's performance. For the concern of clients' fairness, a well-defined fairness metric [58] is used in the paper to measure fairness, which considers both model variance and model utility. Generally, model $\boldsymbol{\theta}_1$ is fairer than model $\boldsymbol{\theta}_2$ if the distribution of test performance across all benign clients is more uniform, i.e.,

$$std\left\{\mathcal{L}_k(\boldsymbol{\theta}_1)\right\}_{k\in K} < std\left\{\mathcal{L}_k(\boldsymbol{\theta}_2)\right\}_{k\in K}, \tag{5}$$

where K is the collection of benign clients, $\mathcal{L}_k(\cdot)$ is the test loss of the client with index k, and $std\{\cdot\}$ denotes the standard deviation. This definition of fairness is reasonable since it promotes models to produce more consistent outcomes across various clients while maintaining acceptable accuracy.

IV. Robust Model Aggregation

The design of Romoa [1] is inspired by some observations of model poisoning attacks. In a compromised FL task, the adversary will either stealthily or overtly tamper with the learning process. The interference can be observed from two aspects: notably extra training iterations for the global convergence or more unexpected fluctuations on the global learning curve. Given these abnormal appearances, it is still challenging to identify the adversary from normal FL participants, especially when randomness and non-IID data are considered.

To address this problem, Romoa introduces an innovative approach combining a hybrid similarity measurement with a lookahead strategy. With the help of the lookahead similarity measurement, Romoa can precisely capture divergences among participants. By quantifying the divergence, Romoa assigns a sanitizing factor to each participant, which is determined by considering both the temporal similarity measurement outcomes and the historical behaviors exhibited by each participant. Then, participants' local model parameters will be sanitized by their own factors during the model aggregation. Unlike existing solutions, Romoa uses a lookahead strategy to capture potential threats, and no participant labor will be dropped hastily. For convenience, we give essential notations used in the rest in Table II for quick reference.

A. Lookahead Similarity Measurement

Previous studies have explored the potential of employing Euclidean distance or cosine similarity as metrics to quantify the dissimilarities among DNN models between FL participants [15], [22]. We have discovered that calculating distance or similarity with a single metric is inadequate. Therefore, we combine different similarity measurements to detect attacks. More importantly, we have developed a novel method for measuring similarity in a lookahead way. The original lookahead strategy proposed in [59] is an alternative optimizer for improving learning stability. In asynchronous updating, all participants are allowed to explore locally for τ iterations between two adjacent syncing points. Thus, we let

TABLE II
NOTATIONS FOR QUICK REFERENCE

Notation	Description
m	the batch size
η	a predefined learning rate
β	the residual rate
t	$t \in [1, T]$, the iteration counter
γ	the moving rate
τ	the syncing period
T	the maximal iteration
P_i	$i \in [1, n]$, the i-th participant of n the participants
$\boldsymbol{x}^i$	the private training data held by P_i
$\boldsymbol{\theta}^i$	model parameters of the i-th participant
$\boldsymbol{\theta}^i_t$	model parameters $\boldsymbol{\theta}^i$ at t iteration
$\boldsymbol{\theta}^{j,w}_{t'}$	the w-th parameter of $\boldsymbol{\theta}^j_{t'}$
$\boldsymbol{\theta}^{i[l]}_t$	the parameters in the l-th layer at t iteration
$\bar{\boldsymbol{\theta}}$	the globally learnt model
$\boldsymbol{g}^i(\boldsymbol{\theta}^i)$	the gradient of local parameters $\boldsymbol{\theta}^i$
$\boldsymbol{\delta}_t$	the participants' local model gradients
$\boldsymbol{F}^i_t$	the sanitizing factor for $\boldsymbol{\theta}^i$ at t iteration
c	the centroid of the clustered parameters
C_t	the number of clusters in the t-th iteration
$S^{i,w}_{cosine}$	cosine similarity for w-th parameter of P_i
$L^{i[l]}_{cosine}$	cosine similarity for l-th layer parameters of P_i
$L^{i[l]}_{pearson}$	pearson correlation measurement for l-th layer parameters of P_i

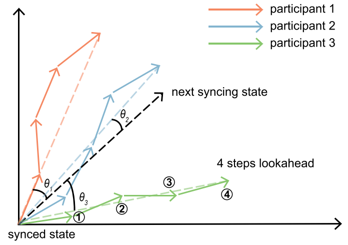


Fig. 1. Cosine similarity using a lookahead strategy ($\tau = 4$).

the PS monitor the exploration phase so that poisons generated during local exploration can be detected before aggregation.

Assuming the whole asynchronous updating process can be divided into numerous periods, say $T = K\tau, K \in \mathbb{N}$. All participants are required to upload local models during exploration. If t' counts continuously from the last syncing point t, P_i will perform τ local training iterations and upload $\boldsymbol{\theta}^i_{t'}$ to the PS for lookahead similarity measurement before the next syncing point, $t' \in [k\tau + 1, (k + 1)\tau]$, $k \in [1, K]$. After collecting all local models, the PS should perform parameter selection first. Specifically, parameters with high absolute values will be selected at the ratio of r (generally assuming $r = \frac{1}{n}$ if no further explanation is given). The selection result of $\boldsymbol{\theta}^i_{t'}$ is denoted by $\tilde{\boldsymbol{\theta}}^i_{t'}$, $|\tilde{\boldsymbol{\theta}}^i_{t'}| = r|\boldsymbol{\theta}^i_{t'}|$. Let $[\boldsymbol{\theta}^{j,w}_{t'}]$ denote the index of w-th parameter of $\boldsymbol{\theta}^j_{t'}$ and $\left\{[\boldsymbol{\theta}^{j,w}_{t'}]\right\}$ as the corresponding index set. Finally, PS merges all participants' parameter selection results:

$$\hat{\boldsymbol{\theta}}^i_{t'} = \tilde{\boldsymbol{\theta}}^i_{t'} \cup \left\{\boldsymbol{\theta}^j_{t'} | [\boldsymbol{\theta}^{j,w}_{t'}] \in \left\{[\tilde{\boldsymbol{\theta}}^{j,w}_{t'}]\right\}, j \in [1, n], i \neq j\right\}. \quad (6)$$

In the t'-th iteration, the PS calculates a lookahead aggregation of the selected parameters, $\bar{\boldsymbol{\theta}}_{t'} = \frac{1}{n}\sum_{i=1}^{n} \hat{\boldsymbol{\theta}}^i_{t'}$. If the state of an expanded parameter selection can be treated as a planar point, then the calculation of lookahead similarity can be illustrated in Figure 1. The angle to be calculated is formed by two edges. One is from the last syncing state (e.g., syncing point t) to the last lookahead state of a participant. The other one derives from the last syncing state to the lookahead aggregation of all participants. Given two updating paths both started with the last synced state $\bar{\boldsymbol{\theta}}_t$, one ended with the next syncing state $\bar{\boldsymbol{\theta}}_{t'}$, the other ended with participant P_i's selected parameters $\hat{\boldsymbol{\theta}}^i_{t'}$, we can define two non-zero vectors $[\bar{\boldsymbol{\theta}}^w_t, \bar{\boldsymbol{\theta}}^w_{t'}]$ and $[\bar{\boldsymbol{\theta}}^w_t, \hat{\boldsymbol{\theta}}^{i,w}_{t'}]$ (w denotes the index of parameters). Then, element-wise cosine similarity measurement for any participant P_i is

$$S^{i,w}_{cosine} = \frac{(\hat{\boldsymbol{\theta}}^{i,w}_{t'} - \bar{\boldsymbol{\theta}}^w_t)(\bar{\boldsymbol{\theta}}^w_{t'} - \bar{\boldsymbol{\theta}}^w_t)^T}{(\sum_{\theta \in \{\hat{\boldsymbol{\theta}}^{i,w}_{t'} - \bar{\boldsymbol{\theta}}^w_t\}} \theta^2)^{(\frac{1}{2}} \times \sum_{\theta \in \{\bar{\boldsymbol{\theta}}^w_{t'} - \bar{\boldsymbol{\theta}}^w_t\}} \theta^2)^{\frac{1}{2}}}. \quad (7)$$

The above definition gives similarity measurement for parameters $\hat{\boldsymbol{\theta}}^i_{t'}$, which are selected according to absolute values. However, it's also important to take accurate measurements of the remaining unselected parameters. Cosine similarity and Pearson correlation are utilized layer-wise to capture divergences of the unselected parameters. If all parameters in the l-th layer of a DNN model are denoted by $\boldsymbol{\theta}^{i[l]}_{t'} \in \mathbb{R}^{M_l}$ (M_l is the total number of parameters in the l-th layer) and function $std(\cdot)$ yields standard deviation.

$$L^{i[l]}_{cosine} = \frac{(\boldsymbol{\theta}^{i[l]}_{t'} - \bar{\boldsymbol{\theta}}^{[l]}_t)(\bar{\boldsymbol{\theta}}^{[l]}_{t'} - \bar{\boldsymbol{\theta}}^{[l]}_t)^T}{(\sum_{\theta \in \{\boldsymbol{\theta}^{i[l]}_{t'} - \bar{\boldsymbol{\theta}}^{[l]}_t\}} \theta^2)^{\frac{1}{2}} \times \sum_{\theta \in \{\bar{\boldsymbol{\theta}}^{[l]}_{t'} - \bar{\boldsymbol{\theta}}^{[l]}_t\}} \theta^2)^{\frac{1}{2}}}, \quad (8)$$

$$L^{i[l]}_{pearson} = \frac{L^{i[l]}_{cosine}}{std(\{\boldsymbol{\theta}^{i[l]}_{t'} - \bar{\boldsymbol{\theta}}^{[l]}_t\}) \times std(\{\bar{\boldsymbol{\theta}}^{[l]}_{t'} - \bar{\boldsymbol{\theta}}^{[l]}_t\})}. \quad (9)$$

B. Model Aggregation With Sanitizing Factor

To mitigate the adversarial impact of poisoning attacks and ensure a consistent model updating trend, we introduce a sanitizing factor $\boldsymbol{F}$. This factor is a weight vector assigned to each model and is derived from the lookahead similarity measurement results. During the model aggregation process performed by the PS, each parameter is multiplied by its corresponding sanitizing factor to undergo sanitization. To convert the similarity measurement results into sanitizing factors, we employ the mean shift algorithm [60], which is utilized to estimate the density of both the model parameters and the measurement results. By applying the mean shift algorithm to the similarity measurement results, we can identify clusters and their respective centroids, which play a crucial role in determining the sanitizing factors for each parameter. For any $\theta_w \in \boldsymbol{\theta}$, if θ_w belongs to a cluster whose centroid is denoted by c_w (the same centroid may be referred to as different identifiers), then the element-wise sanitizing factor is

$$f^i_{S_{cosine}}(\theta_w) = \begin{cases} S^i_{cosine}(\theta_w) - c_w, \text{ if } \theta_w \text{ in } \hat{\boldsymbol{\theta}}^i, \\ 0, \text{ otherwise.} \end{cases} \quad (10)$$

If θ_w belongs to the l-th layer, two layer-wise sanitizing factors can be defined as

$$f^i_{L_{cosine}}(\theta_w) = L^{i[l]}_{cosine}(\theta_w) - c_w, \quad (11)$$

$$f^i_{L_{pearson}}(\theta_w) = L^{i[l]}_{pearson}(\theta_w) - c_w. \quad (12)$$

To combine multiple factors, the minimum is selected as a representative. Then sanitizing factor $\boldsymbol{F}^i_t$ for $\boldsymbol{\theta}^i$ is

$$\boldsymbol{F}^i_t = \{\min\{f^i_{S_{cosine}}(\theta_w), f^i_{L_{cosine}}(\theta_w), f^i_{L_{pearson}}(\theta_w)\}\}_{\theta_w \in \boldsymbol{\theta}^i_t},$$
$$\boldsymbol{F}^i_t = \beta e^{\boldsymbol{F}^i_t} / \sum_{j=1}^{n} e^{\boldsymbol{F}^j_t} + (1 - \beta)\boldsymbol{F}^i_{t-1}, \quad (13)$$

Algorithm 1 robust Model Aggregation (Romoa)

```
Input  : learning rate $\eta$, amount of participants $n$,
         residual rate $\beta$, moving rate $\gamma$, syncing
         period $\tau$, maximal iteration $T$.
Output: globally learnt model $\bar{\boldsymbol{\theta}}$.
for $i \leftarrow 1$ to $n$ do
    $\boldsymbol{\theta}_0^i \leftarrow rand(0,1)$          // initialization
    $\boldsymbol{F}_0^i \leftarrow \frac{1}{n}$
end
Participant $P_i$:
for $i \leftarrow 1$ to $n$ do
    for $t \leftarrow 1$ to $T$ do
        $\boldsymbol{\theta}_t^i \leftarrow \boldsymbol{\theta}_{t-1}^i - \eta g_t^i(\boldsymbol{\theta}_{t-1}^i)$
        upload $\boldsymbol{\theta}_t^i$        // lookahead updating
        if $\tau$ divides $t$ then
            download $\bar{\boldsymbol{\theta}}_t$             // syncing
            $\boldsymbol{\theta}_t^i \leftarrow \boldsymbol{\theta}_t^i - \gamma(\boldsymbol{\theta}_t^i - \bar{\boldsymbol{\theta}}_t)$
        end
    end
end
Parameter Server:
for $t \leftarrow 1$ to $T$ do
    calculating $\boldsymbol{F}_t^i$ for $P_i$
    for $i \leftarrow 1$ to $n$ do    // sanitizing factor
        $\boldsymbol{F}_t^i \leftarrow \beta e^{\boldsymbol{F}_t^i} / \sum_{j=1}^n e^{\boldsymbol{F}_t^j} + (1-\beta)\boldsymbol{F}_{t-1}^i$
    end
    if $\tau$ divides $t$ then
        $\bar{\boldsymbol{\theta}}_t \leftarrow \sum_{i=1}^n \boldsymbol{\theta}_t^i \boldsymbol{F}_t^i$          // aggregation
    end
end
```

where β is a residual rate, accumulating $\boldsymbol{F}_t^i$ with its historical observations ($\beta = 1/2$ if no further explanation is given). For the initialization, we set $\boldsymbol{F}_0^i \leftarrow \frac{1}{n}$ since each participant is assumed to be honest from the very beginning. By integrating the lookahead similarity measurement and the sanitizing factor into FL, we get a main procedure of Romoa in Algorithm 1.

V. Robust and Fair Model Aggregation in Agnostic Federated Learning

Similarity measurement based defenses [1], [16], [26], [32] work under a fundamental assumption that participants' data distributions are known and fixed. However, poisoning attackers can manipulate their data distributions and break the assumption. Therefore, it is more reasonable to study the robust model aggregation problem in the context of agnostic federated learning (AFL) [20]. AFL assumes agnostic data distributions, making it more suitable for investigating poisoning attacks. Furthermore, AFL is designed for unbiased learning, which aligns with our goal of learning fairness. Hence, we will introduce the AFL model and propose the first robust and fair model aggregation solution for AFL, named Romoa-AFL.

A. Federated Learning With Agnostic Data Distributions

We will inherit notations of FL but refine the definitions of data distributions and loss functions for AFL. Generally, participant P_i holds IID data samples $\boldsymbol{x}^i$ of size m_i drawn from distribution $\mathcal{D}_i$, for any $i \in [1, n]$. In AFL [20], the target data distribution of an AFL learning task can be seen as an unknown mixture of $\mathcal{D}_i$, i.e., $\mathcal{D}_\lambda = \sum_{i\in[1,n]} \lambda_i \mathcal{D}_i$. The mixture weight $\lambda \in \Lambda$ of data distributions is unknown, $\Lambda = \{\lambda_1, \ldots, \lambda_n | \sum_{i\in[1,n]} \lambda_i = 1, \lambda_i \geq 0\}$. Then the loss function of AFL is $\mathcal{L}_{\mathcal{D}_\Lambda}(\boldsymbol{\theta}) = \max_{\lambda\in\Lambda} \mathcal{L}_{\mathcal{D}_\lambda}(\boldsymbol{\theta})$.

The concept of learning fairness revolves around reducing the bias that machine learning models may exhibit towards protected categories or features. Similarly, in AFL, fairness can be defined as minimizing the maximum loss on protected categories under any circumstance while avoiding overfitting the data from any particular participant's distribution. This good-intent fairness definition indicates this fairness can be achieved even under agnostic circumstances [20], so long as participants' intentions are good.

The loss of AFL can be bounded by a sum of the empirical loss $\mathcal{L}_{\bar{\mathcal{D}}_\Lambda}(\boldsymbol{\theta})$, a term controlling the complexity of the hypothesis, and a term about the skewness of Λ, where $\bar{\mathcal{D}}_\Lambda = \max_{\lambda\in\Lambda} \mathcal{L}_{\bar{\mathcal{D}}_\lambda}(\boldsymbol{\theta})$, $\bar{\mathcal{D}}_\lambda = \sum_{i\in[1,n]} \lambda_i \hat{\mathcal{D}}_i$, and $\hat{\mathcal{D}}_i$ is an empirical data distribution regarding $\boldsymbol{x}^i$. Hence, AFL minimizes the following maximum loss

$$\max_{\lambda\in conv(\Lambda)} \mathcal{L}_{\bar{\mathcal{D}}_\Lambda}(\boldsymbol{\theta}) + \gamma\|\boldsymbol{\theta}\| + \mu s(\Lambda), \tag{14}$$

where $conv(\Lambda)$ is the convex hull of Λ, γ and μ are the regularization parameters, $\int(\Lambda)$ is the skewness of Λ. We note that AFL seeks to eliminate learning bias as much as possible. Therefore, the Euclidean norm is employed to constrain the complexity of the model tightly. However, directly solving the objective function is still vulnerable to poisoning attacks. To tackle the problem, we introduce the lookahead measurement and the sanitizing factor. In this way, we can ensure the robustness and fairness of AFL while effectively mitigating the adversarial effects of poisoning attacks.

B. AFL Model Aggregation With Sanitizing Factor

Although AFL recommends reducing the model's Euclidean norm, there are other strategies to constrain the model's complexity. Romoa-AFL's primary goal is to constrain the model's complexity by minimizing the divergence of model updates. We assess the similarity of local model updates using Euclidean distances and use the similarity as a restriction rather than limiting Euclidean norms. A similarity-based method can simplify the model by updating the global model to the minimal common states of the parameters. However, this method is vulnerable to collusive poisoning attacks. Moreover, the good-intent fairness of distinct data distributions may be violated. To tackle these problems, we introduce a clustering method to the similarity constraint.

Specifically, we use an off-the-shelf hierarchical clustering method to divide participants' local model gradients $\boldsymbol{\delta}_t$ into C_t clusters in the t-th iteration (the iteration subscript omitted unless necessary). The clustering procedure is executed by the PS, aiming to obtain centroids $\boldsymbol{c}$ for $\boldsymbol{\delta}$ while minimizing the sum of the corresponding distances between $\boldsymbol{c}$ and $\boldsymbol{\delta}$. As an alternative model complexity constraint for AFL, we introduce a clustering-based similarity measurement into the original optimization problem, which can be written as

$$\arg\min \sum_{i=1}^{C} \sum_{j=1}^{m_i} \text{dist}\left(\boldsymbol{\delta}^{j[l]}, \boldsymbol{c}^{i[l]}\right), \tag{15}$$

where m_i is the number of participants in the i-th cluster, $i \in [1, C]$. We note that the clustering procedure here is run for P_j's gradient $\delta^{j[l]}$ of the l-th layer, $l \in [1, L]$. L_2 distance is used as the dist() function. It is also worth mentioning that we find the number of clusters C decreasing along the training process since the global model is getting converged. Hence, it is reasonable to control the number of clusters similarly to learning rate decay. In particular, the number of clusters C in the t-th iteration is

$$C_t = \max\left(2, \lceil C_0/(1+\gamma(t-1)) \rceil\right), \tag{16}$$

where C_0 is a constant no greater than half of the participants' number. In the default setting, we assume $C_0 = \lfloor n/2 \rfloor$.

If we denote by $\texttt{cluster}(\cdot) : \mathbb{R}^n \leftarrow \mathbb{R}^{n_l \times n}$, $n_l = |\delta^{i[l]}|$, $l \in [1, L]$, then the clustering result should be $c_1, c_2, \ldots, c_n = \texttt{cluster}(\delta^{1[l]}, \delta^{2[l]}, \ldots, \delta^{n[l]})$, $c_i \in [1, C]$ indicating the i-th participant's centroid, $i \in [1, n]$. For the cluster associated with centroid c_i, we denote its cardinality by m_i. We note that utilizing centroids to represent the gradients of participants directly is beneficial for good-intention fairness because each distribution will be associated with a good representation if we choose C correctly. However, this straightforward method cannot effectively constrain the divergence of local models. Thus, using Romoa as our foundation, we create a sanitizing factor for AFL in a different way. As for the skewness of Λ and the complexity of the global model, it is rational to shift the distribution of each model parameter towards its neighboring parameters. To this end, we define the sanitizing factor for the l-th layer of participant P_i's local model as

$$\boldsymbol{F}^{i[l]} \leftarrow \begin{cases} \dfrac{1}{K} \displaystyle\sum_{j \in [1,K], j \neq i} \dfrac{1}{\text{dist}\left(\boldsymbol{\delta}^{i[l]}, \boldsymbol{c}_j^{[l]}\right)}, & \text{if } C \geq 3, \\ \dfrac{m_j}{n}, & \text{otherwise.} \end{cases} \tag{17}$$

Considering the fairness and model complexity, we evaluate the sanitizing factor by the average similarity between $\boldsymbol{\delta}^i$ and its K neighboring centroids. The corresponding centroid of $\boldsymbol{\delta}^i$ itself is not taken into account for the reason that $\boldsymbol{\delta}^i$ is close enough to c_i. There is no need to count the distance between them. Higher $\boldsymbol{F}^i$ indicates that $\boldsymbol{\delta}^i$ is more similar to other clusters, which also implies that $\boldsymbol{\delta}^i$ is good-intent to the global model. When poisoning attackers exist, cooperating participants will be represented by the centroids, reducing the adversarial effect of collusion. Stealth attackers will be treated as outliers if they are in benign clusters.

Neighboring clusters are crucial for the calibration of sanitizing factors. Nevertheless, the number of clusters will decrease quickly when the learning process is converging. In particular, when the cluster number $C \leq 2$, the sanitizing factor $\boldsymbol{F}^i$ cannot be calculated using the neighboring clusters. In this situation, model aggregation with sanitizing factors will degenerate into a straightforward weighted sum, which indicates that $\boldsymbol{F}^i$ depends on the cluster cardinality m_i. In other words, the larger cluster gets the higher influence.

Furthermore, we will normalize sanitizing factors of AFL with a softmax function and accumulate them using historical values just like Romoa. However, unlike Romoa, we will sanitize model parameters for each AFL cluster. Overall, we give the complete construction of sanitizing factors for the l-th layer of participants' local model regarding the centroid c_i as

$$\boldsymbol{F}_{c_i}^{[l]} \leftarrow \frac{1}{m_i} \sum_{j=1}^{m_i} \boldsymbol{F}^{j[l]}, \tag{18}$$

$$\boldsymbol{F}_{c_i}^{[l]} \leftarrow \beta e^{\boldsymbol{F}_{c_i}^{[l]}} / \sum_{c_i=1}^{C} e^{\boldsymbol{F}_{c_i}^{[l]}} + (1-\beta) \boldsymbol{F}_{t-1,c_i}^{[l]}, \tag{19}$$

where β is the residual rate. If we average the sanitized model updates, the model complexity will be loosely controlled even if sanitizing factors are normalized. Hence, we will employ clipped centroids in place of local model updates for global model aggregation to constrain the model complexity of AFL rigorously. Specifically, the server bounds the centroids of clustering results before the aggregation. Denoted by $\bar{\boldsymbol{c}} \leftarrow \texttt{clip}(\boldsymbol{c}, c^+, c^-)$ the clipping procedure, the server clips each centroid in $\boldsymbol{c}$ into the range $[c^-, c^+]$, where c^+, c^- are the medians of maximum and minimum values of all clusters across the layer, respectively.

Since the local data distribution of each participant is unknown in the context of AFL, local model updates from different participants may vary arbitrarily. As a result, gradient conflict issues may appear due to the heterogeneity of agnostic data distributions. Thus, we introduce the clipping technique in Romoa-AFL, which has not been used in Romoa. Finally, the server calculates the aggregation result of each model layer using the clipped model centroids $\hat{\boldsymbol{c}}$ and the sanitizing factors $\boldsymbol{F}$. Thus, the l-th layer gradients of the global model can be given as

$$\bar{\boldsymbol{\delta}}^{[l]} \leftarrow \sum_{i=1}^{C} \frac{1}{m_i} \bar{\boldsymbol{c}}^{[l]} \boldsymbol{F}_{c_i}^{[l]}. \tag{20}$$

By summarizing all of the abovementioned steps, we give the Romoa-AFL procedure in Algorithm 2.

VI. Analysis of Robustness and Fairness

In this section, we will analyze the robustness and fairness using game theory and learning guarantees. Please note that Romoa and Romoa-AFL share an underlying robustness guarantee, while Romoa-AFL provides a fairness guarantee, which Romoa does not share.

A. Robustness Analysis

We first formalize a strategic game for the training iteration with potential adversaries in FL as an FL game (FLG). Then, we show that if no defense exists, all participants in FLG will be fully honest or adversarial. Next, we extend FLG to a finitely repeated FL game (rFLG). Finally, we will show that Romoa and Romoa-AFL are secure in rFLG if a Nash equilibrium can be achieved with all participants being honest (i.e., no attacks). FLG is a strategic game, denoted by G, containing the interactions of all participants in each iteration. We assume that all participants are rational and should take

Algorithm 2 robust and Fair Model Aggregation for Agnostic Federated Learning (Romoa-AFL)

```
Input : learning rate $\eta$, amount of participants $n$,
        batch size $m$, residual rate $\beta$, moving rate $\gamma$,
        syncing period $\tau$, amount of layers $L$,
        maximal iteration $T$.
Output: globally learnt model $\bar{\boldsymbol{\theta}}$.
for $i \leftarrow 1$ to $n$ do
    $\boldsymbol{\theta}_0^i \leftarrow rand(0, 1)$            // initialization
    $\boldsymbol{F}_0^i \leftarrow \frac{1}{n}$
end
Participant $P_i$:
for $i \leftarrow 1$ to $n$ do
    for $t \leftarrow 1$ to $T$ do
        $\boldsymbol{\theta}_t^i \leftarrow \boldsymbol{\theta}_{t-1}^i - \eta\nabla\mathcal{L}(\boldsymbol{\theta}_{t-1}^i)$        // updating
        upload $\boldsymbol{\delta}_t^i \leftarrow \boldsymbol{\theta}_t^i - \boldsymbol{\theta}_{t-1}^i$
        download $\boldsymbol{\theta}_t^i \leftarrow \bar{\boldsymbol{\theta}}_t$
    end
end
Parameter Server:
for $t \leftarrow 1$ to $T$ do
    for $l \leftarrow 1$ to $L$ do                    // clustering
        $\boldsymbol{c}_t^{[l]} \leftarrow \mathtt{cluster}(\boldsymbol{\delta}^{1[l]}, \boldsymbol{\delta}^{2[l]}, \dots, \boldsymbol{\delta}^{n[l]})$
        calculating $\boldsymbol{F}_t^{i[l]}$ for $P^{i[l]}$
        $\boldsymbol{F}_{c_i}^{[l]} \leftarrow \frac{1}{m_i}\sum_{j=1}^{m_i} \boldsymbol{F}^{j[l]}$
        $\boldsymbol{F}_{c_i}^{[l]} \leftarrow \beta e^{\boldsymbol{F}_{c_i}^{[l]}} / \sum_{c_i=1}^{C} e^{\boldsymbol{F}_{c_i}^{[l]}} + (1-\beta)\boldsymbol{F}_{t-1,c_i}^{[l]}$
        $\bar{\boldsymbol{c}}^{[l]} \leftarrow \mathtt{clip}(\boldsymbol{c}^{[l]})$                // clipping
        $\bar{\boldsymbol{\theta}}_t^{[l]} \leftarrow \bar{\boldsymbol{\theta}}_{t-1}^{[l]} + \sum_{j=1}^{C} \frac{1}{m_i}\bar{\boldsymbol{c}}_t^{[l]}\boldsymbol{F}_{t,c_j}^{[l]}$
         // sanitizing aggregation
    end
end
```

action simultaneously. By manipulating the poison dosage, the adversary can make the attack more subtle or effective. When the game comes to an end, every participant wants to have a well-trained DNN model. Honest participants win if the model functionality is above a certain threshold, whereas an adversary wins if the attack score exceeds a threshold.

Each participant in an FL task is a natural player in FLG. Participant P_i has an action set A_i, which contains all available actions, $i \in [1, n]$. A utility function mapping an action set to a real-value utility score is u_i. Particularly, $u_i(\boldsymbol{a}) \geq u_i(\boldsymbol{a}')$ if and only if P_i has preference for action set $\boldsymbol{a}$ over action set $\boldsymbol{a}'$, where $\boldsymbol{a}$ and $\boldsymbol{a}' \in \boldsymbol{A}$. Now we can define FLG as a strategic game $G =< \mathcal{P}, \{A_i\}_{i=1}^n, \{u_i\}_{i=1}^n >$. The player set is denoted by $\mathcal{P} = \{P_1, P_2, \dots, P_n\}$. For a general purpose of model poisoning attacks, we define available action set A_i as $\{q_0, q_1, q_2, \dots, q_d\}$, where d is the maximal degree of the poison dosage. Specifically, the action q_0 indicates no poison while the rest of actions $\{q_1, q_2, \dots, q_d\}$ indicate the poison dosage increasing linearly (the player can choose any action by adjusting the stealth factor α). We use $|a_i|/d \in [0.0, 1.0]$ to represent the poison percentage of action a_i.

The utility function in FLG consists of two parts. The first part is information gain from model aggregation, denoted by $\frac{1}{n}\sum_{i=1}^n g(a_i)$, where $g(a_i) = 1 - |a_i|/d$ is a set-valued mapping. The second part is the attack score, another set-valued mapping, denoted by $h(a_i) = |a_i|/d$. Then, the corresponding utility function of P_i can be defined as

$$u_i(\boldsymbol{a}) = \frac{1}{n}\sum\nolimits_{j=1}^n g(a_j) + h(a_i), \tag{21}$$

where $a_i \in A_i$. Obviously, if all participants behave normally, then the total social welfare will equal n while each player yields 1 utility. Given all possible outcomes, the adversary prefers to take the most effective action q_d if all the other participants act normally. In this case, the adversary can get $2 - \frac{1}{n}$ utility while other players get $1 - \frac{1}{n}$ utility. As all participants are rational, they will choose to take the most effective poisoning action and eventually end in 1 utility from the attack, which also yields a total social welfare n.

Now, we extend the FLG into a finitely repeated game rFLG to characterize players' interactions for the iterative learning process. Sanitizing factors in Romoa and Romoa-AFL are intended to punish undesired behavior. Given $G =< \mathcal{P}, \{A_i\}_{i=1}^n, \{u_i\}_{i=1}^n >$, rFLG can be defined as a finitely repeated game of G as $G_0 =< \mathcal{P}, H, S, \{u_i\}_{i=1}^n >$, where $\mathcal{P}$ and $\{u_i\}_{i=1}^n$ are the same player set and utility function set as G, $H = \{\Phi\} \cup \{\cup_{t=1}^T \boldsymbol{A}^t\}$ is the set of historical action profiles, Φ is the initial profile, T is a given positive integer, and $\boldsymbol{A} = \{A_i\}_{i=1}^n$. Additionally, S is the set of strategies for each player, which assigns an action in A_i to every finite sequence of action history. It should be noted that if $a, a' \in A_i$, $u_i(a) \geq u_i(a')$, we will say that P_i has a preference for action sequence $(a^1, a^2, \dots, a, \dots, a^t)$ over action sequence $(a^1, a^2, \dots, a', \dots, a^t)$. We reconstruct utility functions with sanitizing factors as

$$u_i^*(\boldsymbol{a}) = \sum\nolimits_{j=1}^n \frac{e_j}{n} g(a_j) + h(a_i), \tag{22}$$

where e_j can be seen as a predefined price of each player P_j charging for P_i's unsuitable behaviors. To have e_j worked in the same way as the sanitizing factors, we assume that e_j can be determined by the similarity between action profiles of P_i and P_j. Specifically,

$$e_j = \begin{cases} 1, & \text{if } g(a_i) \geq g(a_j), \\ g(a_i), & \text{otherwise.} \end{cases} \tag{23}$$

The adversary who takes attack action will be punished by other participants. Given e_j, we can derive another strategic game $G^* =< \mathcal{P}, \{A_i\}_{i=1}^n, \{u_i^*\}_{i=1}^n >$. Different from G, we can easily conclude that G^* has a unique Nash equilibrium where all players choose to take action q_0, which means no attacks. Then the FL task with sanitizing factors can be defined as another rFLG $G_0^* =< \mathcal{P}, H, S, \{u_i^*\}_{i=1}^n >$. Furthermore, by following the theorem about the Nash equilibrium of a finitely repeated game, we can directly conclude that the outcome of the G_0^* consists of the Nash equilibrium of G^* repeated T times, accomplishing the analysis of robustness.

Theorem 1: FL with Romoa or Romoa-AFL is secure against model poisoning attacks if the number of adversarial participants is less than $\lceil n/2 \rceil$, where n is the total number of FL participants.

B. Fairness Analysis

The fair learning guarantees of AFL have been analyzed in previous work [18], [20]. We will inherit notations and theoretical results from previous work and show that the fairness of Romoa-AFL can be ensured by the learning guarantee of AFL, even if our aggregation strategy is different. In other words, we will show that the desirable good-intent fairness can be achieved since Romoa-AFL can provide the required learning guarantees [18], [20]. Generally, if we denote by $\mathcal{L}$ the cross-entropy loss of AFL, then $\mathcal{L}$ can be bounded by $M > 0$ according to Theorem 1 presented in [20]. The following inequality holds with probability at least $1 - \delta_0$ for any $\delta_0 > 0$ and any $\boldsymbol{\delta}$:

$$\begin{aligned}&\mathcal{L}_{\mathcal{D}_\lambda}(\boldsymbol{\theta})\\&\le \mathcal{L}_{\bar{\mathcal{D}}_\lambda}(\boldsymbol{\theta}) + 2R_{\boldsymbol{m}}(\mathcal{L}, \lambda) + M\epsilon + M\sqrt{\log(|\Lambda_\epsilon|/\delta_0)s(\lambda)/2m},\end{aligned}$$

where $\boldsymbol{m} = (m_1, m_2, \ldots, m_n)$ indicating sample sizes of each participant, $m = \sum_{i\in[1,n]} m_i$, $R_{\boldsymbol{m}}(\mathcal{L}, \lambda)$ is the weighted Rademacher complexity, and Λ_ϵ is a minimum ϵ-cover of Λ in l_1 distance for $\epsilon > 0$.

Recall that AFL aims to minimize the sum of an empirical loss term, a term controlling model complexity, and a term based on skewness. Instead of constraining the complexity through parameter regularization, we adopt a clustering and clipping-based restricted parameter updating strategy. In this way, the objective will be mainly about two sets of parameters, i.e., $\boldsymbol{\theta}$ and Λ. Moreover, we use clusters instead of each participant to control the model aggregation. Hence, the set of centroids $\boldsymbol{c}$ plays a key role in the analysis.

First, we will discuss the variance of cluster-weighted gradients in AFL. If the optimization problem of AFL over $\boldsymbol{\theta}$ and λ is denoted by $\min_{\boldsymbol{\theta}} \max_{\lambda\in\Lambda} \mathcal{L}(\boldsymbol{\theta}, \lambda)$, then we can derive the empirical loss of the model under any circumstance of clustering centroids $\boldsymbol{c}$ as

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{c}) = \sum_{i=1}^{[n]} \sum_{c_i=1}^{C} \boldsymbol{F}^{c_i} \mathcal{L}(\boldsymbol{\theta}^i, \boldsymbol{x}^i), \tag{24}$$

where $\boldsymbol{x}^i$ are drawn from dataset $\bar{\mathcal{D}}^i$. Recall that $\boldsymbol{F}_t^{c_i} = \beta e^{\boldsymbol{F}_t^{c_i}} / \sum_{j\in[1,C]} e^{\boldsymbol{F}_t^{c_j}} + (1 - \beta)\boldsymbol{F}_{(t-1)}^{c_i}$ in the t-th learning iteration. In particular, we assume $\beta = 1$ to omit the historical effects of sanitizing factors. However, we note that this will not wreck the analysis result of learning guarantees. Assuming $\boldsymbol{\theta}^A = \frac{1}{T}\sum_{t\in[1,T]}\sum_{i\in[1,C]} \frac{1}{C}\bar{\boldsymbol{c}}_t \boldsymbol{F}_t^{c_i}$, then the following inequality holds for all clustering results:

$$\begin{aligned}&\mathbb{E}[\max \mathcal{L}(\boldsymbol{\theta}^A, \boldsymbol{c}) - \min\max \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{c})]\\&\le \mathbb{E}[\max \mathcal{L}(\sum_{i\in[1,C]} \frac{1}{m_i}\bar{\boldsymbol{c}}\boldsymbol{F}^i) - \min\max \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{c})]\\&\le \mathbb{E}[\max \mathcal{L}(\bar{\boldsymbol{c}}) - \min\max \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{c})].\end{aligned}$$

As suggested by AFL, we adopt essential assumptions and properties of gradient descent algorithms, including convexity, compactness, bounded gradients, and stochastic variance. Following the AFL convergence analysis, we can conclude the following corollary based on Theorem 2 in [20].

Corollary 1: When the convergence guarantee of stochastic-AFL holds for a bounded time complexity, we have the convergence guarantee of clustered stochastic-AFL by

$$\begin{aligned}&\mathbb{E}[\max \mathcal{L}(\boldsymbol{\theta}^A, \boldsymbol{c}) - \min\max \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{c})]\\&\quad\le \sigma_{\boldsymbol{c}}^2 \mathbb{E}[\max \mathcal{L}(\boldsymbol{\theta}^A, \lambda) - \min\max \mathcal{L}(\boldsymbol{\theta}, \lambda)],\end{aligned}$$

where $\sigma_{\boldsymbol{c}}^2$ is the variance of clustering results.

The above result can be proved by narrowing down $\mathbb{E}[\max \mathcal{L}(\bar{\boldsymbol{c}}) - \min\max \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{c})]$ in the same way as stochastic-AFL with clustering variance brought in. We note that the convergence guarantee of stochastic-AFL depends on essential assumptions of stochastic gradient descent algorithms. Bound of the clustering variance can be inferred directly from the constrained gradients. Hence, Romoa-AFL can be regarded as a derivation of the original AFL. According to the clustering analysis result, $\sigma_{\boldsymbol{c}}^2$ can be bounded so long as the variance of $\boldsymbol{\theta}$ is bounded, which is a direct result of the bounded stochastic gradients. Hence, the conclusion of the fairness of Romoa-AFL can be drawn as follows.

Theorem 2: Romoa-AFL provides good-intent fairness when the convergence guarantee of stochastic-AFL holds.

VII. Evaluation

We have comprehensively evaluated Romoa-AFL on the model's usability, robustness, and fairness. We use an averaging model test accuracy across all classes to assess the model usability. For robustness, we utilize an averaging model error rate over all classes for UPA and a model error rate on the target class for TPA. A standard deviation of test accuracies for all local models will be used for fairness. Furthermore, we will analyze the advantages of Romoa-AFL by comparing it with related solutions for FL robustness and fairness.

A. Evaluation Setup

Data and Models: We use two standard datasets in image classification, MNIST [61] and SVHN [62]. *MNIST* is a grayscale image dataset containing 10-class handwriting digits containing 60,000 training and 10,000 test samples. *SVHN* is an RGB image dataset with 10-class house numbers of Google Street View, containing 73,257 training samples and 26,032 test samples. MNIST and SVHN are both 10-label datasets. We run experiments with different non-IID levels, specifically considering $q = [0.1, 0.25, 0.5, 0.75, 1.0]$. In this way, $q = 0.1$ represents an IID case, while $q \neq 0.1$ indicates non-IID cases. For the MNIST dataset, a model consisting of 3 convolutional layers and 1 fully connected layer is used, while the model of the SVHN dataset consists of 5 convolutional layers and 2 fully connected layers. Both are commonly used model architectures in related studies.

Auxiliary Dataset We note that some existing solutions [26], [27] highly rely on an assumption of auxiliary data on PS, which positively affects defense effectiveness. Evaluation results of model usability and robustness show significant differences between those solutions with or without the assumption of auxiliary data. Therefore, we use an additional dataset in experiments when necessary, sampling 200 clean samples from the original dataset by the PS. A recent study has justified the sampling method [35], which reduces the amount of auxiliary data with a negligible performance loss.

Hyperparameters: All experiments are conducted using the TensorFlow backend. We try our best to compare Romoa-AFL with related work using the same hyper-parameter setting, which is as follows: learning rate $\eta = 0.001$, residual rate $\beta = 0.9$, moving rate $\gamma = 0.9$, sync period $\tau = 4$, batch size 64, the number of participants $n = [10, 20, 100, 200]$. We will use this default setting without explicit statements. Both UPA and TPA attacks will be evaluated in two adversarial settings: a single adversary and multiple adversaries ($\leq \lfloor n/2 \rfloor$). We will run 5 replicate experiments under each setting and use identical hyperparameters for each replicate experiment. The evaluation results shown in the section are averaging results across 5 replicate experiments.

Compared Solutions: We choose FedAvg [3] as a baseline and 10 SOTA Byzantine-robust solutions for the comparison of model usability and robustness, i.e., Krum [22], RFA [21], Median [10], Trimmed-mean [10], SageFlow [27], FLTrust [26], Bulyan [28], DnC [7], FLdetector [30] and FedDefender [29]. Since these solutions have not taken into account learning fairness explicitly, we compare Romoa-AFL with 2 solutions for FL fairness, Ditto [58] and RFFL [63].

Fake Clients and Adaptive Attack: To further evaluate the resistance of Romoa-AFL against poisoning attacks, we also consider that an attacker uses fake clients [38] or an adaptive attack [6]. Fake clients enhance the adversarial effect by enlarging the collusive set. The adaptive attack on a defense solution can help improve poisoning attacks by providing a white-box view of the defense strategy. In the adaptive attack setting, we consider the worst case of a defensive solution, in which the adversary knows the defense strategy and utilizes the information to improve poisoning attacks. The PS in Romoa-AFL calculates sanitizing factors in lookahead iterations and applies them for aggregation after lookahead iterations. The adversary can make use of this design characteristic, behaving normally in the lookahead iterations to deceive the server for a good sanitizing factor. Then, the adversary injects the poison into the global model when the aggregation actually happens. This adaptive attack is quite strong and effective. However, we note that this adaptive attack is too realistic to be implemented unless the server colludes with the adversary, which is impossible. Otherwise, the adaptive attack will fail if the server uses a random number of lookahead iterations, the adaptive attack will fail.

B. Usability Evaluation

We evaluate the usability of different solutions by using the global model test accuracy across all classes. For the scalability of Federated Learning (FL) applications, we consider different numbers of participants, specifically $n = [10, 20, 100, 200]$. The usability evaluation results under attacks are presented in Table III. It is important to note that when the number of participants is $n = 10$, we assign each participant with only one data class, resulting in a completely non-independent and identically distributed setting. For other cases, we follow the data distribution setting of $q = 0.75$.

In previous work by Romoa [1], a global model achieved 95% test accuracy when subjected to the TPA attack, and its test accuracy remained between 70% and 98% during the UPA attack. It is worth mentioning that Romoa outperformed Romoa-AFL in specific TPA evaluation scenarios. However, in the worst-case scenario of UPA, Romoa-AFL improves the global model test accuracy from 70% to approximately 76%.

Please note that TPA aims at specific targets, moderately harming global test accuracy. Therefore, Romoa-AFL has a slight accuracy drop under the TPA. Since the UPA intends to corrupt the global model indiscriminately, model usability will be significantly damaged without protection. Romoa-AFL obviously outperforms other solutions under the UPA, especially when the proportion of adversarial participants increases. When collusive attacks happen, the accuracy of the global model will inevitably suffer, even if it is protected by other solutions. We can conclude that Romoa-AFL has the advantage of defending against collusive poisoning attacks.

We compare multiple widely used secure aggregation methods under identical conditions to assess their relative strengths and weaknesses. Different from other methods performed on the server side, FedDefender is a robust FL algorithm that incorporates noise injection on the client side, leveraging an auxiliary network for knowledge distillation. While FedDefender, DnC, and several other methods can effectively safeguard the model's availability in scenarios with a limited number of attackers, their performance often suffers a significant decline as the number of adversaries nears half. In contrast, Romoa-AFL demonstrates remarkable resilience, maintaining consistently good performance even in the worst-case scenario, where the number of adversaries is high. Furthermore, in certain instances, the accuracy of Romoa-AFL is comparable to that of FLTrust and SageFlow, two methods that assume the availability of a clean dataset on the server. However, Romoa-AFL achieves this level of performance without relying on such prior knowledge, making it a versatile option in the realm of secure aggregation.

C. Robustness Evaluation

The robustness will be evaluated by the error rate of the poisoning target, i.e., the global model for UPA and the target class for TPA. The resistance to the collusion of multiple attackers will also be evaluated for defense solutions. A lower error rate implies a better defense capability. We recall that Romoa [1] outperforms Krum and RFA in most cases of IID data. The lowest error rate of Romoa achieves 0.1% and 2% when defending against TPA and UPA, respectively. Table V shows the error rate results in detail. In contrast to Table III, the average error rate is tested on benign clients to show the defense performance, verifying the robustness.

From the robustness evaluation results, we can conclude that Romoa-AFL has the advantage of defending against strong collusive attacks when other solutions fail, especially for large-scale FL tasks. It is worth mentioning that in TPA cases, Romoa-AFL can defend against poisoning attacks and preserve higher original label confidence than other solutions, as shown in Table V and Table III. We note that fair FL solutions [58], [63] are not evaluated because they pay more

TABLE III
MODEL USABILITY EVALUATION RESULTS

participants (n)			10			20			100			200		
adversarial participants			0	1	4	0	1	9	0	1	49	0	1	99
MNIST	TPA	baseline	96.58%	96.22%	94.52%	93.99%	96.48%	96.17%	94.07%	93.45%	92.17%	92.80%	91.76%	88.43%
		Krum	94.38%	88.35%	95.21%	98.33%	94.17%	90.43%	86.10%	87.88%	80.41%	89.14%	61.74%	66.82%
		RFA	93.20%	94.43%	96.25%	95.62%	95.38%	94.17%	92.10%	92.88%	83.15%	92.42%	91.16%	66.62%
		Median	94.27%	93.48%	93.88%	94.73%	95.91%	94.63%	94.13%	93.55%	84.25%	93.25%	91.56%	87.93%
		Trimmed-mean	95.62%	94.33%	95.76%	95.85%	95.79%	94.70%	93.18%	90.26%	88.46%	92.57%	90.24%	88.50%
		FLTrust	96.20%	***97.81%***	97.75%	97.29%	***97.51%***	**97.60%**	**97.53%**	93.62%	92.48%	**96.96%**	94.05%	91.57%
		Sageflow	96.15%	96.48%	***97.94%***	***98.10%***	97.44%	96.28%	97.41%	93.15%	***92.73%***	***96.88%***	93.02%	**91.44%**
		Bulyan	94.87%	90.07%	95.33%	97.84%	95.23%	92.10%	91.92%	91.63%	86.49%	92.61%	90.55%	80.95%
		DnC	***97.03%***	96.75%	95.22%	97.12%	97.54%	94.45%	95.36%	94.91%	90.38%	93.95%	***94.87%***	90.13%
		FLDetector	95.23%	95.45%	94.32%	95.85%	96.27%	93.60%	94.58%	***95.17%***	88.35%	93.39%	94.51%	88.05%
		FedDefender	**97.38%**	**98.06%**	97.00%	**98.32%**	97.05%	95.48%	**97.80%**	**95.92%**	90.77%	95.83%	**95.09%**	87.17%
		Romoa-AFL	96.76%	97.88%	**98.08%**	97.99%	**97.60%**	***97.46%***	96.99%	93.68%	**92.80%**	95.88%	93.34%	***91.39%***
	UPA	baseline	96.71%	94.77%	26.42%	96.49%	33.76%	11.35%	***96.24%***	30.76%	11.35%	94.18%	34.28%	11.35%
		Krum	20.79%	11.35%	10.28%	23.97%	11.35%	14.25%	20.73%	10.13%	6.56%	14.54%	12.64%	7.56%
		RFA	***98.28%***	77.35%	36.44%	97.74%	97.43%	94.66%	94.71%	93.86%	11.35%	91.14%	90.83%	11.35%
		Median	**98.18%**	96.22%	91.85%	97.63%	94.31%	92.68%	96.46%	92.95%	15.64%	96.22%	92.47%	12.83%
		Trimmed-mean	96.33%	96.67%	92.06%	96.41%	91.34%	93.80%	95.88%	84.42%	22.25%	93.67%	88.44%	20.99%
		FLTrust	97.14%	**98.26%**	***97.43%***	***98.10%***	**98.13%**	96.27%	96.19%	94.08%	92.83%	95.37%	94.08%	**94.44%**
		Sageflow	97.33%	97.21%	97.02%	97.54%	96.42%	***96.89%***	96.23%	**96.12%**	**96.24%**	***95.18%***	**96.02%**	93.57%
		Bulyan	22.38%	14.16%	12.72%	24.15%	15.36%	12.72%	22.49%	18.97%	14.07%	24.09%	16.43%	13.76%
		DnC	95.02%	96.20%	93.18%	96.26%	96.88%	94.15%	94.18%	93.25%	91.45%	94.31%	94.08%	91.29%
		FLDetector	96.43%	95.06%	94.45%	95.92%	96.59%	91.49%	95.27%	***95.41%***	92.09%	91.64%	92.18%	92.32%
		FedDefender	97.85%	***97.60%***	94.77%	**98.50%**	***98.10%***	92.11%	96.07%	92.84%	92.33%	**95.88%**	***95.71%***	93.04%
		Romoa-AFL	97.93%	97.52%	**97.65%**	98.12%	97.90%	**96.93%**	**96.33%**	92.08%	***93.55%***	94.69%	93.55%	***93.61%***
SVHN	TPA	baseline	***92.17%***	82.28%	71.75%	92.43%	84.54%	70.26%	90.72%	**87.56%**	65.23%	89.87%	80.45%	19.59%
		Krum	85.52%	80.39%	73.20%	85.00%	81.67%	66.87%	88.32%	74.13%	77.45%	88.65%	61.37%	65.89%
		RFA	84.10%	82.81%	78.39%	82.93%	80.35%	76.40%	87.03%	85.01%	48.62%	80.41%	75.93%	50.85%
		Median	90.14%	81.58%	73.26%	**93.33%**	82.75%	74.41%	89.43%	88.10%	72.26%	90.05%	***86.65%***	72.14%
		Trimmed-mean	89.18%	79.42%	73.64%	88.98%	85.34%	81.46%	**92.71%**	87.43%	84.39%	**92.82%**	**86.77%**	77.26%
		FLTrust	91.26%	**88.71%**	85.26%	91.97%	86.61%	***85.45%***	90.31%	86.69%	84.04%	90.29%	86.25%	81.44%
		Sageflow	**92.33%**	87.22%	85.38%	92.97%	**87.31%**	84.22%	90.22%	85.71%	***86.14%***	90.26%	86.07%	***82.28%***
		Bulyan	87.33%	81.25%	75.61%	86.49%	83.91%	70.43%	87.17%	76.29%	72.33%	87.92%	68.40%	67.11%
		DnC	90.43%	86.02%	84.22%	91.58%	84.25%	84.30%	***91.16%***	**87.93%**	85.44%	***92.13%***	85.27%	80.46%
		FLDetector	90.56%	***88.49%***	81.26%	90.94%	***87.25%***	82.44%	88.15%	86.52%	81.61%	87.11%	81.46%	75.15%
		FedDefender	92.12%	86.05%	**87.44%**	92.53%	85.43%	82.96%	90.85%	84.76%	81.02%	90.15%	80.44%	78.63%
		Romoa-AFL	91.42%	87.31%	***86.37%***	92.28%	86.97%	**86.52%**	89.23%	86.51%	**86.26%**	89.03%	80.68%	**82.45%**
	UPA	baseline	91.43%	84.59%	58.74%	90.88%	82.56%	55.59%	82.61%	19.59%	19.59%	83.45%	19.60%	19.59%
		Krum	85.19%	78.59%	19.95%	89.98%	75.55%	15.38%	48.33%	45.25%	19.26%	44.98%	20.32%	10.26%
		RFA	93.24%	92.96%	19.59%	92.18%	**93.12%**	19.59%	***89.21%***	**86.41%**	19.59%	79.63%	75.91%	19.59%
		Median	91.41%	88.15%	72.36%	90.31%	85.43%	71.73%	86.23%	84.16%	82.42%	83.89%	75.21%	70.51%
		Trimmed-mean	93.56%	90.18%	75.52%	93.11%	88.17%	74.22%	87.10%	80.41%	76.32%	82.64%	73.45%	70.35%
		FLTrust	***94.32%***	***93.15%***	90.62%	***94.00%***	91.17%	***92.20%***	**89.63%**	84.63%	82.48%	***84.77%***	***74.73%***	73.65%
		Sageflow	93.89%	**93.25%**	***92.52%***	93.46%	92.17%	92.12%	88.72%	84.05%	***82.61%***	83.68%	74.32%	***73.88%***
		Bulyan	87.46%	80.15%	65.47%	80.33%	20.77%	60.21%	60.34%	58.49%	44.15%	54.73%	48.55%	16.32%
		DnC	92.46%	92.15%	90.93%	93.08%	92.87%	90.15%	85.72%	84.29%	81.77%	82.19%	72.55%	71.03%
		FLDetector	92.04%	92.79%	90.33%	**94.12%**	92.94%	91.54%	83.05%	82.92%	81.74%	81.56%	72.18%	71.62%
		FedDefender	**94.79%**	92.64%	90.82%	92.35%	92.00%	89.78%	85.28%	84.05%	82.11%	**84.92%**	**75.98%**	72.01%
		Romoa-AFL	93.18%	93.05%	**92.68%**	92.50%	***92.99%***	**92.87%**	85.91%	***85.72%***	**85.23%**	81.89%	75.56%	**75.98%**

The highest accuracy is **bold**, while the second one is ***italicized bold***.

attention to local model performance rather than robustness. The evaluation results of fair FL solutions show that error rates under attacks reach about 90%, which is extraordinarily high.

When all participants are benign, DnC exhibits a remarkably low error rate, close to 0.04%. However, it falls short compared to other methods in terms of defense against both UPA and UPA attacks. In the case of UPA, advanced robustness methods like FLDetector and FedDefender demonstrate decent performance when the level of attacker collusion is minimal, typically involving only one adversary. Nevertheless, as the number of attackers increases, Romoa-AFL stands out by substantially reducing the error rate and gradually showcasing its superiority over the aforementioned methods.

It is essential to highlight that the robustness evaluation results of FLTrust and Sageflow, presented here, differ from the original work [26], [27]. This discrepancy arises because we have employed the auxiliary dataset consisting of 200 images sampled from the training data. It should be noted that when an auxiliary dataset is sampled from an out-of-distribution dataset, such as ImageNet, the robustness result tends to decrease. This indicates that the quality of the auxiliary dataset plays a significant role in influencing the overall robustness of the methods under evaluation. Furthermore, it is worth mentioning that both FLTrust and Sageflow may exhibit improved robustness results if a more substantial auxiliary dataset is utilized.

In addition to the previous poisoning attacks based on compromised clients, we also take into account another poisoning approach by injecting fake clients. Throughout each local poisoning process, the attacker will update the fake client model to fit a basic model, which has been randomly initialized before. Since the fake client attack concentrates on the UPA, evaluating the attack for the TPA is not reasonable. Table IV presents attack results of UPA with fake clients. When the number of benign clients is much greater than the number of fake clients, the attack has a negligible impact on Romoa-AFL's global model accuracy. Even when the number of attackers reaches the limit, the model accuracy can still exceed 77% with the protection of Romoa-AFL.

D. Fairness Evaluation

The test accuracy of each participant's local model is used to learn fairness evaluation. Intuitively, if all participants' learning outcomes are treated equally, the global model should correctly predict each participant's local test data. As a result, we assess fairness by calculating the standard deviation of test accuracy results across all benign local models. A lower standard deviation indicates better fairness for benign

TABLE IV
MODEL USABILITY RESULTS UNDER FAKE CLIENTS ATTACK

participants (n)		10		20	
adversarial participants		1	4	1	9
MNIST	baseline	82.01%	79.77%	84.59%	28.26%
	Krum	75.77%	45.03%	76.25%	45.50%
	RFA	84.16%	62.98%	86.15%	67.75%
	Median	91.69%	76.68%	***95.47%***	71.25%
	Trimmed-mean	***94.17%***	78.21%	95.23%	75.29%
	FLTrust	93.94%	***84.55%***	**95.48%**	82.29%
	Sageflow	93.85%	84.12%	95.13%	***82.79%***
	Romoa-AFL	**94.36%**	**88.46%**	95.17%	**84.15%**
SVHN	baseline	85.30%	7.69%	8.26%	81.12%
	Krum	82.65%	59.27%	83.71%	64.30%
	RFA	82.26%	54.34%	81.53%	66.26%
	Median	82.37%	71.51%	83.22%	72.79%
	Trimmed-mean	89.33%	69.28%	84.98%	72.64%
	FLTrust	***84.95%***	***76.23%***	***85.74%***	79.76%
	Sageflow	83.18%	75.86%	**86.66%**	***79.81%***
	Romoa-AFL	**85.23%**	**78.12%**	85.70%	**80.23%**
participants (n)		100		200	
adversarial participants		1	49	1	99
MNIST	baseline	92.75%	9.78%	97.20%	9.27%
	Krum	89.09%	49.14%	83.22%	44.08%
	RFA	91.57%	69.50%	94.34%	69.24%
	Median	97.23%	70.71%	**98.39%**	72.61%
	Trimmed-mean	97.56%	72.17%	95.23%	68.21%
	FLTrust	***97.71%***	77.63%	***97.89%***	***84.73%***
	Sageflow	**97.89%**	***78.45%***	96.33%	**85.14%**
	Romoa-AFL	95.93%	**79.11%**	96.25%	84.28%
SVHN	baseline	90.11%	10.04%	84.61%	8.94%
	Krum	86.94%	64.70%	83.27%	66.72%
	RFA	86.28%	69.23%	81.65%	74.78%
	Median	***90.19%***	76.28%	**85.21%**	74.06%
	Trimmed-mean	82.33%	69.28%	84.34%	71.56%
	FLTrust	**90.59%**	***78.15%***	84.77%	**81.95%**
	Sageflow	90.33%	76.21%	85.06%	***79.12%***
	Romoa-AFL	88.98%	**80.29%**	***85.13%***	77.74%

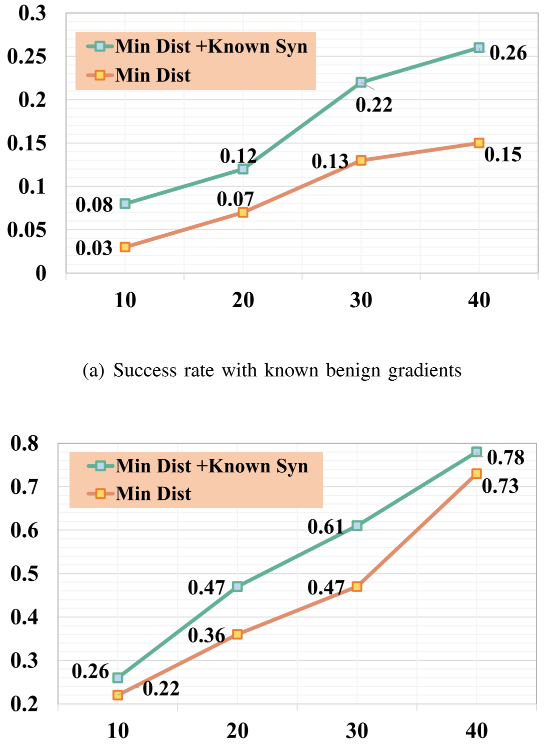
The highest accuracy is **bold**, while the second one is ***italicized bold***.

participants. We compare our solutions with two fair FL solutions, Ditto [58] and RFFL [63]. Since most robust FL solutions have not considered learning fairness, they cannot be compared fairly.

We can conclude from Table VI that Romoa-AFL outperforms Ditto and RFFL in most cases but yields higher standard deviations in the case of 10 participants with one attacker. We note that this data distribution setting is an extreme case because the number of classes equals the number of participants. However, this phenomenon can be relieved by inserting more public or augmented samples. Moreover, it is primarily due to prevailing low test accuracies on local models that Ditto and RFFL have lower deviations. In fact, the average test accuracies of Ditto and RFFL are approximately 20% lower than Romoa-AFL in this case.

E. Security Against Adaptive Attacks

As mentioned in the experimental setup, We consider adaptive attacks for UPA, where the adversary is aware of defense strategies. Based on the existing adaptive attacks, we implement an adaptive attack on Romoa-AFL. Specifically, the PS keeps track of each local model's updates in Romoa-AFL within a fixed synchronization interval. Therefore, the attacker can deceive the server in the lookahead steps and upload the local model with the largest deviation from the global direction in the synchronous iteration. Meanwhile, since Romoa-AFL is based on similarity measurement and clustering results, the attacker could try to reduce distances between other local models for stealthness.



(a) Success rate with known benign gradients

(b) Success rate with unknown benign gradients

Fig. 2. Attack success rate with different number of adversaries. The orange legend Min Dist represents that the adversary minimizes the distance between the poisoned model and the benign model, and the green legend represents that the adversary spoofs the server in the lookahead rounds and performs the poisoning in the synchronization rounds.

Figure 2 illustrates the results of Romoa-AFL under the adaptive attack. The attack is performed on the MNIST dataset with a data distribution parameter of $q = 0.5$ and 100 participants. The attacker's goal is to minimize the distance between the poisoned and benign models using the lookahead similarity metric proposed in Romoa-AFL. Figure 2 shows the attack success rate for two different levels of adversary knowledge: (1) when the attacker has perfect knowledge of the gradients of all players, and (2) when the attacker knows the aggregation algorithm but lacks access to the gradients of other benign clients. When attackers have synchronization knowledge, they can fool the server on the lookahead rounds but perform poisoning on synchronization rounds.

With perfect gradient knowledge, the attacker can construct a model that closely resembles the benign models, leading to a higher attack success rate. For instance, when the number of adversaries reaches 40, the attack's success rate is about 0.78. In a more realistic scenario, where the attacker lacks access to the gradients of other benign clients but knows the aggregation algorithm and synchronization rounds, the attack's success rate drops to 0.26. When a collusive attack happens, especially when the number of attackers approaches half the total number of participants, defenses are seriously sabotaged. Romoa-AFL has an obvious decision error since attackers have deceived considerable aggregation factors, enlarging adversarial effects.

In the adaptive attack scenario, the attacker mimics a normal client during rounds where trust scores are calculated,

TABLE V

ROBUSTNESS RESULTS OF DEFENSE AGAINST POISONING

participants (n)			10			20			100			200		
adversarial participants			0	1	4	0	1	9	0	1	49	0	1	99
MNIST	TPA	baseline	0.13%	99.86%	99.94%	0.08%	99.34%	99.93%	**0.00%**	99.93%	99.99%	0.02%	90.00%	99.99%
		Krum	0.06%	99.97%	99.32%	0.07%	5.30%	**0.04%**	0.08%	**0.00%**	86.56%	0.05%	0.07%	41.64%
		RFA	0.12%	99.02%	99.91%	0.06%	48.70%	99.95%	0.03%	***0.04%***	99.99%	0.04%	0.53%	99.99%
		Median	***0.04%***	6.33%	16.83%	0.44%	9.23%	29.12%	0.03%	0.46%	58.74%	**0.01%**	0.72%	43.28%
		Trimmed-mean	**0.02%**	6.20%	10.47%	0.43%	10.25%	8.72%	**0.00%**	0.05%	61.52%	0.11%	0.27%	54.62%
		FLTrust	***0.04%***	7.88%	***8.26%***	0.21%	**4.47%**	5.18%	0.04%	0.11%	50.20%	**0.00%**	***0.06%***	0.85%
		Sageflow	0.08%	8.21%	9.24%	0.73%	8.79%	7.75%	0.06%	0.15%	***48.92%***	**0.00%**	0.14%	***0.28%***
		Bulyan	0.06%	90.13%	90.28%	0.08%	6.49%	10.10%	0.04%	90.37%	90.85%	0.05%	0.16%	50.17%
		DnC	0.06%	2.61%	6.77%	0.08%	5.24%	12.79%	***0.02%***	0.36%	41.33%	**0.00%**	0.09%	0.97%
		FLDetector	0.15%	***1.31%***	8.47%	**0.05%**	8.77%	20.73%	0.08%	***0.04%***	61.94%	0.22%	**0.00%**	30.18%
		FedDefender	0.05%	5.76%	10.28%	***0.06%***	10.25%	14.95%	0.09%	***0.04%***	54.83%	0.05%	0.08%	4.56%
		Romoa-AFL	0.08%	***1.96%***	***5.83%***	0.12%	***4.98%***	***4.23%***	***0.02%***	2.23%	**46.90%**	**0.00%**	**0.00%**	**0.04%**
	UPA	baseline	2.98%	90.24%	91.47%	3.24%	89.97%	9.35%	2.25%	86.68%	88.65%	***5.57%***	83.29%	88.65%
		Krum	4.73%	89.72%	88.69%	6.35%	90.58%	89.40%	3.28%	89.87%	93.44%	10.04%	89.03%	89.62%
		RFA	5.98%	25.43%	88.67%	4.15%	6.02%	9.08%	**0.54%**	***13.55%***	89.41%	**5.30%**	24.82%	89.68%
		Median	4.22%	28.93%	12.57%	6.44%	8.08%	14.09%	1.24%	14.13%	36.17%	6.18%	**18.32%**	34.90%
		Trimmed-mean	**1.04%**	24.67%	14.33%	4.12%	8.75%	11.03%	1.59%	14.10%	32.62%	8.31%	22.52%	30.74%
		FLTrust	***1.24%***	24.88%	***8.37%***	***2.08%***	8.42%	***8.02%***	0.88%	13.62%	20.04%	5.95%	21.77%	***27.43%***
		Sageflow	1.83%	24.99%	8.62%	2.88%	10.15%	10.52%	1.10%	13.96%	***18.06%***	6.18%	20.88%	28.56%
		Bulyan	4.78%	80.21%	89.73%	8.65%	84.29%	85.14%	0.35%	85.42%	90.33%	12.62%	85.15%	86.96%
		DnC	5.24%	24.91%	15.36%	8.79%	7.52%	14.98%	0.64%	15.88%	23.06%	7.29%	22.02%	38.28%
		FLDetector	2.46%	27.15%	10.94%	5.92%	8.34%	9.02%	0.51%	16.46%	18.92%	5.91%	24.73%	38.06%
		FedDefender	7.05%	**20.25%**	10.96%	**1.36%**	**3.42%**	18.01%	***0.09%***	***13.28%***	23.95%	6.04%	20.88%	34.75%
		Romoa-AFL	3.12%	***24.51%***	**4.49%**	2.18%	***5.37%***	**6.29%**	0.58%	13.89%	**17.58%**	7.02%	***20.45%***	**26.31%**
SVHN	TPA	baseline	0.08%	99.73%	32.37%	**0.01%**	70.27%	79.06%	**0.00%**	30.26%	61.20%	**0.00%**	15.04%	85.05%
		Krum	**0.05%**	**1.12%**	**0.01%**	0.23%	**0.01%**	7.37%	***0.01%***	**0.00%**	2.14%	0.04%	0.68%	2.47%
		RFA	8.77%	20.14%	97.9%	6.54%	9.12%	88.74%	**0.00%**	**0.00%**	49.19%	**0.00%**	**0.00%**	77.55%
		Median	5.34%	27.66%	1.43%	3.69%	8.92%	2.77%	0.03%	***0.01%***	2.78%	**0.00%**	0.28%	4.51%
		Trimmed-mean	8.26%	25.34%	1.59%	4.25%	10.33%	2.67%	***0.01%***	0.01%	2.10%	**0.00%**	1.02%	1.84%
		FLTrust	***0.06%***	16.64%	2.36%	5.42%	11.44%	**1.80%**	**0.00%**	**0.00%**	***0.08%***	**0.00%**	**0.00%**	***0.46%***
		Sageflow	0.09%	10.32%	3.24%	1.29%	12.05%	2.08%	**0.00%**	**0.00%**	0.020%	**0.00%**	**0.00%**	0.51%
		Bulyan	0.09%	***1.29%***	**0.01%**	0.27%	0.01%	80.92%	**0.00%**	**0.00%**	6.72%	**0.00%**	**0.00%**	3.61%
		DnC	**0.05%**	10.48%	6.81%	**0.01%**	**0.01%**	6.93%	**0.00%**	0.02%	3.48 %	0.12%	0.09%	2.88%
		FLDetector	0.15%	10.56%	1.28%	***0.02%***	0.16%	4.88%	0.05%	0.02%	2.35%	**0.00%**	**0.00%**	1.22%
		FedDefender	***0.06%***	7.14%	0.24%	***0.02%***	0.25%	2.39%	0.29%	0.05%	1.36%	***0.03%***	***0.12%***	6.34%
		Romoa-AFL	0.07%	6.25%	***0.17%***	0.08%	***0.13%***	***1.75%***	**0.00%**	**0.00%**	**0.02%**	**0.00%**	**0.00%**	**0.26%**
	UPA	baseline	**8.12%**	82.49%	81.76%	8.21%	78.43%	80.89%	12.17%	80.43%	80.42%	23.88%	80.41%	80.41%
		Krum	8.55%	92.41%	89.05%	10.14%	27.87%	93.32%	10.05%	59.41%	80.74%	26.87%	78.95%	91.97%
		RFA	14.38%	72.31%	80.41%	7.63%	**12.71%**	80.50%	10.96%	***27.28%***	80.70%	32.11%	47.62%	80.74%
		Median	12.46%	47.79%	18.91%	7.24%	16.87%	15.33%	14.65%	30.19%	42.18%	28.49%	***42.25%***	***52.43%***
		Trimmed-mean	13.88%	51.40%	14.75%	6.84%	20.92%	***14.25%***	12.11%	27.60%	38.25%	**22.39%**	52.10%	53.84%
		FLTrust	10.56%	***46.25%***	16.30%	***6.72%***	13.05%	16.33%	11.93%	28.40%	32.19%	26.87%	48.86%	52.97%
		Sageflow	11.26%	48.25%	15.88%	7.69%	13.34%	15.33%	11.03%	29.14%	***30.45%***	28.11%	48.95%	55.64%
		Bulyan	10.64%	85.18%	90.61%	12.15%	30.73%	92.46%	12.06%	65.17%	82.94%	30.18%	75.49%	80.77%
		DnC	12.92%	50.62%	14.85%	15.51%	15.02%	25.91%	11.48%	35.48%	30.18%	25.88%	50.80%	62.15%
		FLDetector	***8.43%***	45.17%	18.26%	7.25%	14.28%	28.94%	**9.04%**	27.33%	36.76%	24.40%	42.33%	64.28%
		FedDefender	12.26%	45.71%	***13.78%***	**5.33%**	14.21%	24.88%	11.85%	**24.02%**	36.99%	24.77%	**39.03%**	60.55%
		Romoa-AFL	9.93%	**42.32%**	**10.84%**	8.25%	***12.92%***	**13.85%**	***10.42%***	29.97%	**29.72%**	***23.05%***	48.60%	**50.26%**

The lowest error rate is **bold**, the second-lowest one is ***italicized bold***.

TABLE VI

FAIRNESS EVALUATION RESULTS UNDER ATTACKS

participants (n)			10		20	
adversarial participants			1	4	1	9
MNIST	TPA	Ditto	**0.59**	5.87	4.76	***3.24***
		RFFL	***0.62***	***4.02***	***3.85***	4.98
		Romoa-AFL	7.34	**0.07**	**0.16**	**0.13**
	UPA	Ditto	**0.62**	4.06	***4.08***	3.63
		RFFL	***0.63***	***0.99***	4.63	***3.27***
		Romoa-AFL	5.89	**0.14**	**0.18**	**0.38**
SVHN	TPA	Ditto	4.23	10.19	***5.33***	9.97
		RFFL	***2.92***	***7.66***	13.05	***6.78***
		Romoa-AFL	**2.92**	**0.26**	**0.61**	**0.32**
	UPA	Ditto	**2.91**	7.94	***4.64***	***5.38***
		RFFL	***2.92***	***3.57***	9.12	5.45
		Romoa-AFL	***2.92***	**0.24**	**0.62**	**0.38**
participants (n)			100		200	
adversarial participants			1	49	1	99
MNIST	TPA	Ditto	7.59	***5.00***	7.52	***7.25***
		RFFL	***2.54***	5.18	***6.55***	8.06
		Romoa-AFL	**2.33**	**0.70**	**0.92**	**0.94**
	UPA	Ditto	***6.82***	***6.64***	***7.47***	***6.67***
		RFFL	6.94	6.89	11.11	7.57
		Romoa-AFL	**2.51**	**4.20**	**1.90**	**2.09**
SVHN	TPA	Ditto	***4.30***	10.75	***4.10***	***4.38***
		RFFL	4.68	***9.51***	5.22	6.65
		Romoa-AFL	**0.46**	**0.24**	**0.79**	**0.54**
	UPA	Ditto	4.77	***3.92***	***4.68***	***4.35***
		RFFL	***4.01***	8.96	5.28	7.06
		Romoa-AFL	**0.78**	**1.05**	**1.19**	**1.30**

The lowest standard deviation is **bold**, the second-lowest is ***italicized bold***.

thereby securing a higher trust score and a greater aggregation weight. However, during the synchronization round, the attacker utilizes its accumulated weights to carry out poisoning attacks, significantly affecting the global model. The primary limitation of the algorithm in this situation lies in its reliance on historical trust scores for weight allocation, which prevents it from detecting and responding to real-time poisoning behavior. Since the attacker remains covert during trust score calculation rounds, the algorithm cannot issue advance warnings or implement preventative measures. To counter such a sophisticated adversary, simply adjusting the frequency of synchronization rounds can be an effective strategy. We can minimize the attacker's impact by narrowing the time gap between synchronization rounds. Even if the attacker manages to poison the model in a particular round, the damage will be confined to a limited scope as the next synchronization round will swiftly arrive to update and rectify the global model.

VIII. LIMITATION AND CONCLUSION

Romoa-AFL demonstrates effectiveness in defeating poisoning attacks. This resilience is primarily attributed to its method of allocating sanitizing factors based on the similarity measurement of local models, which mitigates the influence of poisoning attacks on the global model. Notably, backdoor attacks aim to manipulate the model's judgment regarding data with specific characteristics, resembling the targeted data poisoning attacks we considered. Additionally, adversaries carrying out backdoor attacks may stealthily evade detection by controlling the poisoning proportion, and the triggers become more challenging to detect in scenarios with high heterogeneity among participant data. Consequently, during the aggregation process with Romoa-AFL, some malicious models that deviate significantly from the majority and potentially harbor backdoors may be excluded, thereby decreasing the success rate of backdoor attacks.

Personalized federated learning [64] aims to generate models tailored to local data, optimizing for factors like high accuracy. To a certain extent, fairness can also be viewed as a prerequisite for personalization. It ensures that all clients can obtain a model that aligns with their unique data characteristics and needs rather than merely favoring users with vast amounts of data or superior quality. As previously mentioned, Romoa-AFL incorporates fairness as a constraint, guaranteeing that each user or client can reap the benefits of the federated learning process, thus providing a broader personalized experience.

However, Romoa-AFL's effectiveness is limited when confronting inference attacks [45], [46], [47]. Inference attacks exploit the information the model outputs to deduce details about the training data. Despite the aggregation algorithm's ability to filter out similar models and mitigate the impact of poisoning attacks, attackers can still deduce sensitive information from the training data by analyzing the model's output or other related information. Furthermore, inference attacks often capitalize on vulnerabilities or inherent features of the model itself, which fall outside the scope of security guarantees provided by Romoa-AFL.

By studying poisoning attacks against FL, we realize that secure model aggregation should be essential to practical FL applications. Although security against poisoning attacks is important, we suggest that fairness should be considered at the same time. Otherwise, the global model learned may be biased towards more common features while neglecting rare features. We model FL with poisoning attacks using AFL and explicitly design the fairness constraint. We propose Romoa-AFL as an alternative to secure FL solutions, ensuring robustness and fairness simultaneously. Based on the analysis and comprehensive evaluations, we conclude that Romoa-AFL can provide guarantees of robustness, fairness, and resistance to poisoning attacks. More importantly, Romoa-AFL outperforms the existing solutions dedicated to a specific aspect of the guarantees mentioned above in most cases, including collusive attacks and large-scale tasks.

Acknowledgment

The authors would like to thank the editor and reviewers for the time and efforts they have kindly made. The article has been improved by following their helpful comments.

References

[1] Y. Mao, X. Yuan, X. Zhao, and S. Zhong, "Romoa: Robust model aggregation for the resistance of federated learning to model poisoning attacks," in *Computer Security—ESORICS*. Cham, Switzerland: Springer, 2021.

[2] C. Li et al., "Towards certifying the asymmetric robustness for neural networks: Quantification and applications," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 6, pp. 3987–4001, Nov. 2022.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. AISTATS*, 2017, pp. 1273–1282.

[4] S. Banabilah, M. Aloqaily, E. Alsayed, N. Malik, and Y. Jararweh, "Federated learning review: Fundamentals, enabling technologies, and future applications," *Inf. Process. Manag.*, vol. 59, no. 6, Nov. 2022, Art. no. 103061.

[5] 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 (SP)*, May 2022, pp. 1354–1371.

[6] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in *Proc. USENIX Secur.*, 2020, pp. 1605–1622.

[7] 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, pp. 1–19.

[8] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," 2017, *arXiv:1712.05526*.

[9] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2018, pp. 19–35.

[10] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. ICML*, 2018, pp. 5650–5659.

[11] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *Proc. ICML*, 2019, pp. 634–643.

[12] P. G. John, D. Vijaykeerthy, and D. Saha, "Verifying individual fairness in machine learning models," in *Proc. UAI*, 2020, pp. 749–758.

[13] S. Yeom and M. Fredrikson, "Individual fairness revisited: Transferring techniques from adversarial robustness," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 1–8.

[14] W. Zhang, S. Pan, S. Zhou, T. Walsh, and J. C. Weiss, "Fairness amidst non-IID graph data: Current achievements and future directions," 2022, *arXiv:2202.07170*.

[15] C. Fung, C. J. Yoon, and I. Beschastnikh, "The limitations of federated learning in Sybil settings," in *Proc. RAID*, 2020, pp. 301–316.

[16] S. Awan, B. Luo, and F. Li, "CONTRA: Defending against poisoning attacks in federated learning," in *Computer Security—ESORICS*. Cham, Switzerland: Springer, 2021.

[17] U. Michieli and M. Ozay, "Are all users treated fairly in federated learning systems?" in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2021, pp. 2318–2322.

[18] H. Chen, T. Zhu, T. Zhang, W. Zhou, and P. S. Yu, "Privacy and fairness in federated learning: On the perspective of tradeoff," *ACM Comput. Surv.*, vol. 56, no. 2, pp. 1–37, Feb. 2024.

[19] Y. H. Ezzeldin, S. Yan, C. He, E. Ferrara, and A. S. Avestimehr, "FairFed: Enabling group fairness in federated learning," in *Proc. AAAI*, 2023, pp. 7494–7502.

[20] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *Proc. ICML*, 2019, pp. 4615–4625.

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

[22] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proc. NIPS*, 2017, pp. 1–11.

[23] C. Dong et al., "Privacy-preserving and Byzantine-robust federated learning," *IEEE Trans. Dependable Secure Comput.*, vol. 21, no. 2, pp. 889–904, Mar./Apr. 2024.

[24] C. Xie, S. Koyejo, and I. Gupta, "Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance," in *Proc. ICML*, 2019, pp. 6893–6901.

[25] C. Xie, S. Koyejo, and I. Gupta, "Zeno++: Robust fully asynchronous SGD," in *Proc. ICML*, 2020, pp. 10495–10503.

[26] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "FLTrust: Byzantine-robust federated learning via trust bootstrapping," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2021, pp. 1–18.

[27] J. Park, D.-J. Han, M. Choi, and J. Moon, "SageFlow: Robust federated learning against both stragglers and adversaries," in *Proc. NIPS*, 2021, pp. 840–851.

[28] E. M. El Mhamdi, R. Guerraoui, and S. Rouault, “The hidden vulnerability of distributed learning in Byzantium,” in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 3521–3530.

[29] S. Park et al., “FedDefender: Client-side attack-tolerant federated learning,” in *Proc. 29th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2023, pp. 1850–1861.

[30] Z. Zhang, X. Cao, J. Jia, and N. Z. Gong, “FLDetector: Defending federated learning against model poisoning attacks via detecting malicious clients,” in *Proc. 28th ACM SIGKDD Conf. Knowl. Disc. Data Min.*, 2022, pp. 2545–2555.

[31] J. So, B. Güler, and A. S. Avestimehr, “Byzantine-resilient secure federated learning,” *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 2168–2181, Jul. 2021.

[32] T. D. Nguyen et al., “FLGUARD: Secure and private federated learning,” *Crytography Secur.*, Jan. 2021.

[33] H. Lycklama, L. Burkhalter, A. Viand, N. Küchler, and A. Hithnawi, “RoFL: Robustness of secure federated learning,” 2021, *arXiv:2107.03311*.

[34] L. Song et al., “PMPL: A robust multi-party learning framework with a privileged party,” in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2022, pp. 2689–2703.

[35] Z. Xiang, T. Wang, W. Lin, and D. Wang, “Practical differentially private and Byzantine-resilient federated learning,” *Proc. ACM Manag. Data*, vol. 1, no. 2, pp. 1–26, Jun. 2023.

[36] G. Baruch, M. Baruch, and Y. Goldberg, “A little is enough: Circumventing defenses for distributed learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.

[37] G. Sun, Y. Cong, J. Dong, Q. Wang, L. Lyu, and J. Liu, “Data poisoning attacks on federated machine learning,” *IEEE Internet Things J.*, vol. 9, no. 13, pp. 11365–11375, Jul. 2022.

[38] 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. Workshops (CVPRW)*, Jun. 2022, pp. 3395–3403.

[39] H. Li, X. Sun, and Z. Zheng, “Learning to attack federated learning: A model-based reinforcement learning attack framework,” in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 35007–35020.

[40] 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.

[41] H. Wang et al., “Attack of the tails: Yes, you really can backdoor federated learning,” in *Proc. NIPS*, 2020, pp. 1–15.

[42] C. Xie, K. Huang, P.-Y. Chen, and B. Li, “DBA: Distributed backdoor attacks against federated learning,” in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–19.

[43] R. Ning, J. Li, C. Xin, and H. Wu, “Invisible poison: A blackbox clean label backdoor attack to deep neural networks,” in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2021, pp. 1–10.

[44] X. Lyu et al., “Poisoning with cerberus: Stealthy and colluded backdoor attack against federated learning,” in *Proc. AAAI Conf. Artif. Intell.*, Jun. 2023, vol. 37, no. 7, pp. 9020–9028.

[45] L. Fowl, J. Geiping, W. Czaja, M. Goldblum, and T. Goldstein, “Robbing the fed: Directly obtaining private data in federated learning with modified models,” in *Proc. ICLR*, 2022, pp. 1–10.

[46] M. Lam, G.-Y. Wei, D. Brooks, V. J. Reddi, and M. Mitzenmacher, “Gradient disaggregation: Breaking privacy in federated learning by reconstructing the user participant matrix,” in *Proc. ICML*, 2021, pp. 5959–5968.

[47] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, “Beyond inferring class representatives: User-level privacy leakage from federated learning,” in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Apr. 2019, pp. 2512–2520.

[48] D. Pasquini, D. Francati, and G. Ateniese, “Eluding secure aggregation in federated learning via model inconsistency,” in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2022, (pp. 2429–2443.

[49] D. Alistarh, Z. Allen-Zhu, and J. Li, “Byzantine stochastic gradient descent,” in *Proc. NIPS*, 2018, pp. 1–11.

[50] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, “Defending against saddle point attack in Byzantine-robust distributed learning,” in *Proc. ICML*, 2019, pp. 7074–7084.

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

[52] R. Shokri and V. Shmatikov, “Privacy-preserving deep learning,” in *Proc. 53rd Annu. Allerton Conf. Commun., Control, Comput.*, Sep. 2015, pp. 909–910.

[53] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, “Data poisoning attacks against federated learning systems,” in *Computer Security—ESORICS*. Cham, Switzerland: Springer, 2020.

[54] O. Suciu, R. Marginean, Y. Kaya, H. Daume III, and T. Dumitras, “When does machine learning FAIL? Generalized transferability for evasion and poisoning attacks,” in *Proc. USENIX Secur.*, 2018, pp. 1299–1316.

[55] 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. (ICASSP)*, May 2020, pp. 8861–8865.

[56] R. Binns, “On the apparent conflict between individual and group fairness,” in *Proc. Conf. Fairness, Accountability, Transparency*, Jan. 2020, pp. 514–524.

[57] L. Lyu, X. Xu, Q. Wang, and H. Yu, “Collaborative fairness in federated learning,” in *Federated Learning: Privacy Incentive*. Cham, Switzerland: Springer, 2020.

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

[59] M. R. Zhang, J. Lucas, G. Hinton, and J. Ba, “Lookahead optimizer: K steps forward, 1 step back,” in *Proc. NIPS*, 2019, pp. 1–12.

[60] K. Fukunaga and L. Hostetler, “The estimation of the gradient of a density function, with applications in pattern recognition,” *IEEE Trans. Inf. Theory*, vol. IT-21, no. 1, pp. 32–40, Jan. 1975.

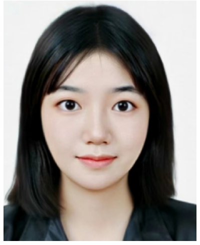
[61] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[62] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, “Reading digits in natural images with unsupervised feature learning,” in *Proc. NIPS Workshop Deep Learn. Unsupervised Feature Learn.*, 2011, p. 7.

[63] X. Xu and L. Lyu, “A reputation mechanism is all you need: Collaborative fairness and adversarial robustness in federated learning,” 2020, *arXiv:2011.10464*.

[64] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, “Towards personalized federated learning,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 12, pp. 9587–9603, Dec. 2023.

Yunlong Mao (Member, IEEE) received the B.S. and Ph.D. degrees in computer science from Nanjing University in 2013 and 2018, respectively. He is currently an Associate Professor with the State Key Laboratory for Novel Software Technology, Nanjing University. His research interests include security, privacy, machine learning, and blockchain.



Zhujing Ye (Student Member, IEEE) is currently pursuing the degree with the Department of Computer Science, Nanjing University. She is interested in security and privacy.

Xinyu Yuan received the degree from the Department of Computer Science, Nanjing University. She is currently with Alibaba. Her research interests include security and privacy.

Sheng Zhong (Fellow, IEEE) received the B.S. and M.S. degrees in computer science from Nanjing University in 1996 and 1999, respectively, and the Ph.D. degree in computer science from Yale University in 2004. He is currently a Professor with the State Key Laboratory for Novel Software Technology, Nanjing University. His research interests include security, privacy, and economic incentives.