Defending Against Sophisticated Poisoning Attacks with RL-based Aggregation in Federated Learning

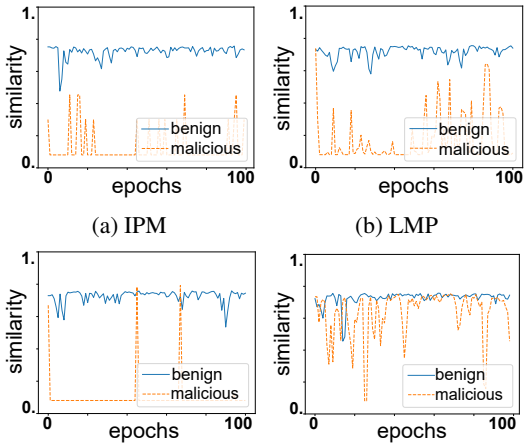
Yujing Wang[1,2], **Hainan Zhang**[1,2*], **Sijia Wen**[1,2], **Wangjie Qiu**[1,2], **Binghui Guo**[2]

[1]Beijing Advanced Innovation Center for Future Blockchain and Privacy Computing
[2] School of Artificial Intelligence, Beihang University, China
{wangyujing, zhanghainan}@buaa.edu.cn

Abstract

Federated learning is susceptible to model poisoning attacks, especially those meticulously crafted for servers. Traditional defense methods mainly focus on updating assessments or robust aggregation against manually crafted myopic attacks. When facing advanced attacks, their defense stability is notably insufficient. Therefore, it is imperative to develop adaptive defenses against such advanced poisoning attacks. We find that benign clients exhibit significantly higher data distribution stability than malicious clients in federated learning in both CV and NLP tasks. Therefore, the malicious clients can be recognized by observing the stability of their data distribution. In this paper, we propose AdaAggRL, an RL-based Adaptive Aggregation method, to defend against sophisticated poisoning attacks. Specifically, we first utilize distribution learning to simulate the clients' data distributions. Then, we use maximum mean discrepancy (MMD) to calculate the pairwise similarity of the current local model data distribution, its historical data distribution, and global model data distribution. Finally, we use policy learning to adaptively determine the aggregation weights based on the above similarities. Experiments on four real-world datasets demonstrate that the proposed defense model significantly outperforms widely adopted defense models for sophisticated attacks.

Code — https://github.com/TAP-LLM/AdaAggRL.



(a) IPM (b) LMP

(c) EB (d) RL-attack

Figure 1: The statistical results of the similarity between the current client data distribution and its historical data distributions under four types of attacks vary with the training epochs on MNIST dataset. The x-axis denotes the number of client update rounds, and the y-axis represents the similarity between the current and its historical data distributions.

Introduction

Federated Learning (FL) enables distributed model training across local devices, preserving data privacy while leveraging diverse local data to improve performance. It is widely applied in areas including smart healthcare (Chaddad, Wu, and Desrosiers 2023), financial services (Byrd and Polychroniadou 2020), IoT (Nguyen et al. 2021), and intelligent transportation (Yamany, Moustafa, and Turnbull 2021). However, FL systems are vulnerable, and the performance of the aggregated model is susceptible to model poisoning attacks from unknown clients (Zheng et al. 2024), especially the sophisticated poisoning strategies tailored for central servers. In this work, we focus on untargeted model poisoning attacks, where malicious devices aim to maximally reduce the accuracy of the global model by sending customized gradients to the server.

Traditional defense methods mainly rely on designing local model update assessment mechanisms or using robust aggregation methods to mitigate the impact of poisoning attacks. However, these defense methods are primarily targeted at manually crafted myopic attack strategies, and their defense stability is lacking when facing advanced attacks. For example, Li, Sun, and Zheng propose using distribution learning to simulate the data distribution of the central server and employing reinforcement learning (RL) to tailor attack policy for the aggregation process, making it less detectable.Therefore, it is urgent to develop adaptive defenses against such learnable advanced poisoning attacks.

Most benign clients exhibit significant data distribution stability in FL. In normal cases, the current training data distribution simulated by the client's parameters should align with its historical simulated data distribution. This is be-

* Corresponding author.

cause benign clients typically employ random sampling and undergo multi-round training based on their local data, ensuring the stability of the simulated data distribution. However, malicious clients require gradient attacks, so its simulated data distribution between current and history lacks regularity. To validate this, we conduct a statistical analysis of mainstream attack methods, namely IPM (Xie, Koyejo, and Gupta 2020), LMP (Fang et al. 2020), EB (Bhagoji et al. 2019) and RL-based attacks (Li, Sun, and Zheng 2022), measuring the similarity of their data distribution with history after each round of updates on MNIST, as shown in Figure 1. We find that benign clients maintain higher data distribution similarity, while attack clients show no discernible patterns. We also observe the same phenomenon on NLP tasks, as shown in Appendix[1]. Therefore, malicious clients can be recognized through the stability of their simulated data distribution.

This paper proposes an RL-based Adaptive Aggregation method, AdaAggRL, to thwart sophisticated poisoning attacks. It determines aggregation weights of local models by comparing the stability of client data distributions. Specifically, we first use distribution learning to emulate client data distributions based on the uploaded model parameters. Next, we use the maximum mean discrepancy (MMD) to calculate the pairwise similarity of the current local model data distribution, its historical data distribution, and the global model data distribution, to evaluate the stability of the client's data distribution. Considering that the accuracy of distribution learning can potentially impact the calculation of the similarities above, we use reconstruction similarity as an evaluation metric for the quality of distribution learning. Finally, we use the policy learning method TD3 to adaptively determine the aggregation weights based on these similarities.

Experimental results on four real-world datasets demonstrate that AdaAggRL defense method significantly outperforms existing defense approaches (Blanchard et al. 2017; Yin et al. 2018; Sun et al. 2019) and achieves a more stable global model accuracy even facing sophisticated attacks, such as RL-based attacks (Li, Sun, and Zheng 2022).

The innovations of this paper are as follows:

- We propose an RL-based adaptive aggregation method AdaAggRL to defend against sophisticated untargeted model poisoning attacks tailored for servers in FL, aiming to advance the development of defense systems.
- We observe stable training data distribution in benign clients over time, contrasting with irregular distribution caused by disruptive attempts from malicious clients in both CV and NLP tasks. Thus, we propose four metrics as RL environmental cues, utilizing policy learning to determine local model aggregation weights based on observed cue changes.
- Experimental results on four datasets demonstrate that the proposed AdaAggRL defense method can maintain more stable global model accuracy than baselines, even when more advanced customized attacks are applied.

[1]Appendix can be found at: https://arxiv.org/abs/2406.14217.

Related Work

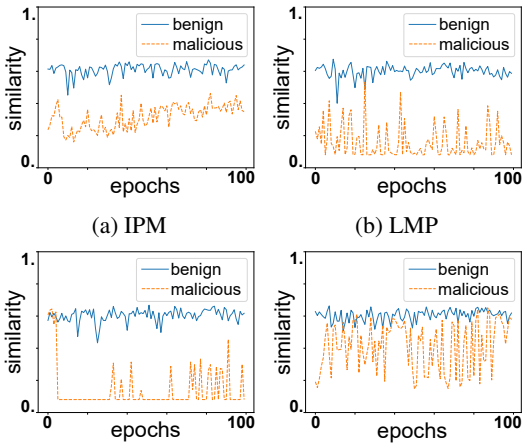
Poisoning Attacks

Based on the attacker's objectives, poisoning attacks can be classified into targeted poisoning attacks aiming to misclassify specific input sets (Bhagoji et al. 2019; Baruch, Baruch, and Goldberg 2019; Bagdasaryan et al. 2020) and untargeted attacks aimed at reducing the overall accuracy of the global model (Fang et al. 2020; Xie, Koyejo, and Gupta 2020; Shejwalkar and Houmansadr 2021). Current untargeted attack methods typically employ heuristic-based approaches (Xie, Koyejo, and Gupta 2020) or optimize myopic objectives (Fang et al. 2020; Shejwalkar and Houmansadr 2021; Shejwalkar et al. 2022). Bhagoji et al. generate malicious updates through explicit enhancement, optimizing for a malicious objective strategically designed to induce targeted misclassification. Xie, Koyejo, and Gupta manipulate the attacker's gradients to ensure the inner product with the true gradients becomes negative. Fang et al. generate malicious updates by solving an optimization problem. However, these attack methods require local updates from benign clients or accurate global model parameters to generate significant adversarial impacts and often yield suboptimal results when robust aggregation rules are employed.

To address these deficiencies, Li, Sun, and Zheng propose a model-based RL framework for guiding untargeted poisoning attacks in FL system. They utilize the server's updates to approximate the server's data distribution, subsequently employing the learned distribution as a simulator for FL environment. Based on the simulator, they utilize RL to automatically generate effective attacks, resulting in significantly reducing the global accuracy. Even when the server employs robust aggregation rules, this customized method can still maintain a high level of attack effectiveness.

Defenses for Poisoning Attacks

Current defense strategies against FL poisoning attacks can be categorized into two types: one involves designing local model update evaluation mechanisms to identify malicious client-submitted model parameters (Cao et al. 2020; Sattler et al. 2020; Zhang et al. 2022), and the other is based on designing novel Byzantine fault-tolerant aggregation algorithms using mathematical statistics to improve the robustness of aggregation (Blanchard et al. 2017; Yin et al. 2018; Sun et al. 2019; Rajput et al. 2019; Xie, Koyejo, and Gupta 2019). In evaluation mechanisms, Sattler et al. divide updates into different groups based on cosine similarity between the model parameters submitted by clients, mitigating the impact of poisoning attacks. In response to more covert poisoning attacks, some evaluation methods (Cao et al. 2020; Park et al. 2021) require the server to collect a portion of clean data as a basis for validating model updates. In robust aggregation algorithms, statistical methods (Blanchard et al. 2017; Yin et al. 2018; Sun et al. 2019) compare local updates and remove statistical outliers before updating the global model. For example, Blanchard et al. employ the square-distance metric to measure distances among local updates and select the local update with the minimum distance as the global parameters. Yin et al. sort the values of parame-



(a) IPM (b) LMP

(c) EB (d) RL-attack

Figure 2: The statistical results of the similarity between the current client data distribution and the global model data distribution under attacks vary with epochs on MNIST.

ters in all local model updates and consider the median value of each parameter as the corresponding parameter value in the global model update. Sun et al. perform gradient clipping on parameter updates before aggregation. Due to the susceptibility of statistical estimates to an outlier, existing aggregation methods cannot guarantee accuracy well and are still susceptible to local model poisoning attacks (Bhagoji et al. 2019; Fang et al. 2020).

These defense methods mainly rely on the local model parameters. For sophisticated attacks, such as RL-based customized attacks, it is difficult to identify malicious updates from the parameter information alone accurately. Moreover, statistics-based robust aggregation is not flexible enough.

Motivation

During FL, benign clients are selected by the server through random sampling and trained on their local data for a certain number of rounds. Therefore, the simulated data distribution obtained through gradient inversion should align with their historical distribution. But the data distribution of malicious clients lacks regularity, due to their need to conduct model attacks. To verify this, we conduct a statistical analysis of the similarity between data distributions for mainstream attack methods such as IPM, LMP, EB, and RL-attack, as shown in Figure 1. The similarity of data distributions for benign clients remains high and stable, while malicious clients lack any regular pattern. We also observe the same phenomenon on NLP tasks, as shown in Appendix.

Moreover, the data distribution of benign clients for current steps is similar to the global model and remains stable, but the similarity of malicious clients is lower, as shown in Figure 2. Since the data distribution of the global model represents the average state of normal data, the distribution of benign clients should always be consistent with the global model. Instead, malicious clients do not possess this property. Therefore, we can compare data distribution similarity between the current client and global model to observe variations and assess the degree of malicious behavior. Similarly, the similarity between historical distribution and global model distribution is higher and more stable for benign clients than malicious ones, as shown in Appendix.

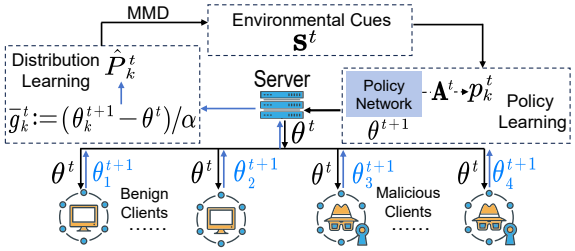


Figure 3: An overview of our AdaAggRL

Since the quality of distribution learning greatly affects the accuracy of distribution similarities, the reconstruction similarity of distribution learning is used as an evaluation metric for assessing the quality of distribution learning. Reconstruction similarity measures the similarity between the inversion gradients from distribution learning and the gradients updated by clients. Higher reconstruction similarity indicates higher confidence in current distribution learning.

RL-based Adaptive Aggregation Methods

Task Definition

In the context of FL (McMahan et al. 2017), a system comprises K clients, with each client k possessing a fixed local dataset $D_k = \{(x_{kj}, y_{kj})\}_{j=1}^{n_k}$, where n_k is the size of D_k. The local objective of client k is $F_k(\theta) = \frac{1}{n_k}\sum_{j=1}^{n_k} l(\theta, x_{kj}, y_{kj})$, where l is the loss function. And (x_{kj}, y_{kj}) is the j-th sample drawn i.i.d. from some distribution P_k. $\hat{P}_k$ denotes the empirical distribution of n_k data samples. The optimization objective of FL is: $\min_\theta f(\theta) = \sum_{k=1}^{K} p_k F_k(\theta)$, p_k represents the weight of client k. During FL, in each epoch $t \geq 0$, the server randomly selects a subset C^t from all clients and distributes the latest global model parameters θ^t. The chosen clients train on their local datasets, updating parameters as $\theta_k^{t+1} = \theta^t - \alpha \nabla F_k(\theta)$, where α is the learning rate. Then they upload the updated model parameters. The server aggregates the received parameters according to a specific aggregation rule $Aggr$, to obtain the new global model parameters $\theta^{t+1} = Aggr(\theta_{C^t}^{t+1})$.

We assume that the server is non-malicious. Malicious clients have sufficient knowledge of the server, including model structure, loss function, learning rate, and other key parameters, to demonstrate the effectiveness of AdaAggRL in defending against strong attacks.

Framework Overview

In AdaAggRL framework (see Algorithm in Appendix), the server determines the weights for local model aggregation by assessing the stability of client data distributions, as shown in Figure 3. Firstly, we employ distribution learning to simulate the client's data distribution $\hat{P}_k^t$ based on

the locally uploaded model parameters θ_k^{t+1}. Secondly, we calculate similarity metrics $S_{k,cl}$ between the current client and historical data distribution, $S_{k,cg}$ between the current client and global model data distribution, and $S_{k,lg}$ between the client's historical and global model data distribution. Finally, RL is utilized to adaptively determine the weight p_k for local model aggregation based on these metrics and the reconstruction similarity $S_{k,R}$ from distribution learning.

Distribution Learning

According to local model parameters θ_k^{t+1} uploaded by clients, the server simulates the local data distribution $\hat{P}_k^t$ of clients by gradient inversion. In this work, we adapt the inverting gradients (IG) method (Geiping et al. 2020) for distribution learning. The IG method reconstructs the data sample by optimizing the loss function based on the cosine similarity between the real gradient and the gradient generated by the reconstructed data.

For each epoch $t \geq 0$, the server receives the clients' locally updated model parameters and calculates the corresponding batch level gradient $\bar{g}_k^t := (\theta_k^{t+1} - \theta^t)/\alpha$. The server then solves the following optimization problem with a batch of randomly generated dummy data and labels D_{dummy}: $\min_{D_{\text{dummy}}} 1 - \frac{\langle \nabla_\theta F_{D_{\text{dummy}}}(\theta_k^{t+1}), \bar{g}_k^t \rangle}{\|\nabla_\theta F_{D_{\text{dummy}}}(\theta_k^{t+1})\| \cdot \|\bar{g}_k^t\|} + \frac{\beta}{B'} \Sigma_{(x,y)\in D_{\text{dummy}}} \mathrm{TV}(x)$, where β is a fixed parameter, B' is the size of the dummy data batch, $F_{D_{\text{dummy}}}(\theta) = \frac{1}{B'} \sum_{(x,y)\in D_{\text{dummy}}} l(\theta, x, y)$, TV calculates the total variation(Rudin, Osher, and Fatemi 1992). While solving the optimization problem, D_{dummy} is continuously updated. The optimization terminates after max_iters iterations, then outputs the updated data as the reconstructed data samples D_{rec}, and the reconstruction similarity of client k, denoted as $S_{k,R} = \frac{\langle \nabla_\theta F_{D_{\text{rec}}}(\theta_k^{t+1}), \bar{g}_k^t \rangle}{\|\nabla_\theta F_{D_{\text{rec}}}(\theta_k^{t+1})\| \cdot \|\bar{g}_k^t\|}$.

Environmental Cues

The distribution of samples is extracted by employing a pre-trained CNN to convert the image samples D_{rec} into a collection of feature vectors V. For each client k, the difference $d_{k,cl}$ between this feature distribution and the history distribution is calculated using the maximum mean discrepancy (MMD) (Arbel et al. 2019; Wang et al. 2021) as $d_{k,cl} = \mathrm{MMD}(V_k^{\text{current}}, V_k^{\text{history}}) \in [0, +\infty)$. Here, V_k^{current} and V_k^{history} are feature vectors of current and historical data distributions respectively. V_g represents feature vectors of the global model data distribution, obtained by averaging feature vectors from all participating clients. Subsequently, the dissimilarity between the current data distribution of client k and the global one is $d_{k,cg} = \mathrm{MMD}(V_k^{\text{current}}, V_g)$. Similarly, the dissimilarity between the historical data distribution and the global one is $d_{k,lg} = \mathrm{MMD}(V_k^{\text{history}}, V_g)$. And the similarity between the current data distribution and the historical data distribution $S_{k,cl}$ is calculated using the following formula:

$$S_{k,cl} = 2 \cdot \cos(\tanh(\frac{d_{k,cl}}{2})) - 1 \tag{1}$$

So $S_{k,cl} \in (0, 1]$, and as the current data distribution obtained through gradient inversion becomes more consistent with the historical data distribution, the value of $S_{k,cl}$ increases. Therefore, we can determine $S_{k,cg}$ between the current client data distribution and the global model data distribution, as well as $S_{k,lg}$ between the historical client data distribution and the global model data distribution.

Actions Learning

The server dynamically adapts the aggregation weights based on three similarity metrics and the reconstruction similarity associated with each client. By utilizing experiences sampled from the simulated environment, the server engages in learning a collaborative defense strategy aimed at minimizing empirical loss. This learning process employs the RL algorithm TD3 (Fujimoto, Hoof, and Meger 2018).

State: To simulate the training process of FL, including malicious clients and their behaviors, an environment for RL is set up. For each epoch t in FL, let $\mathbf{s}^t = (s_{k_1}^t, s_{k_2}^t, ..., s_{k_{|C^t|}}^t)^T \in (0, 1]^{|C^t|\times 4}$ be the state of the reinforcement learning simulation environment, where k_j denotes the client identifier participating in the training. Here, $s_k^t := (S_{k,R}, S_{k,cl}, S_{k,cg}, S_{k,lg})$ represents the state of client k with the reconstruction similarity and three obtained metrics. So the state search space is $(0, 1]^{|C^t|\times 4}$, where $|C^t|$ is the number of clients participating in federated aggregation.

Action: Through RL, the server is trained to make the decision $\mathbf{A}^t = (\mathbf{a}^t, b^t)^T \in [0, 1]^5$ based on the current FL environment state $\mathbf{s}^t$. Here, $\mathbf{a}^t$ is a four-dimensional vector, and $\Sigma_{i=1}^4 a_i^t = 1$, where $a_i^t \in [0, 1]$ represents the weighted weight for four environmental parameters $(S_{k,R}, S_{k,cl}, S_{k,cg}, S_{k,lg})$, and $b^t \in [0, 1]$ represents a threshold. So the action space is $[0, 1]^5$.

State transition: The FL system changes based on the server's decision $\mathbf{A}^t$. Specifically, the FL system first obtains $\hat{\mathbf{w}} = \mathbf{s}^t \cdot \mathbf{a}^t \in \mathbb{R}^{|C^t|}$, i.e., weighting the environmental parameters. $\hat{w}_k = s_k^t \cdot \mathbf{a}^t$ indicates the score of client k. Then, function $g(\cdot)$ maps $\hat{\mathbf{w}}$ to $[0, 1]$ interval and normalizes it, resulting in $\tilde{\mathbf{w}} = g(\hat{\mathbf{w}})$. Let $\delta = \max(\tilde{\mathbf{w}}) \cdot b^t$, define

$$f_\delta(x) = \begin{cases} x & \text{if } x > \delta \\ 0 & \text{if } x \leq \delta \end{cases} \tag{2}$$

Then $\mathbf{w} = f_\delta(\tilde{\mathbf{w}})$. The function f_δ denotes clients with lower scores, which are considered to exhibit malicious behavior and are excluded from aggregation. The generation process of the server policy reveals that the threshold δ within f_δ is also adaptively adjusted based on the state.

Additionally, a vector $\mathbf{h}^t \in \mathbb{N}^K$ is introduced in the FL system environment to record the malicious behaviors of each client. $h_k^0 = 0$, when $\tilde{w}_k \leq \delta$, $h_k^{t+1} = h_k^t + 1$, otherwise, $h_k^{t+1} = \max(h_k^t - 1, 0)$, representing the occurrence of malicious behavior by client k. Based on these outcomes, the FL system determines the aggregation strategy for global model parameters of the new round as follows,

$$\theta^{t+1} = \Sigma_{k\in C^t} \frac{w_k}{\lambda^{h_k^{t+1}}} \cdot \theta_k^{t+1} \tag{3}$$

	Our	Krum	Median	C-Median	FLtrust	Clipping
MNIST	1.652	1.744	1.069	1.556	1.107	1.327
Cifar10	7.201	8.698	3.537	4.469	3.589	3.696

Table 1: The training time of different algorithms for one round of FL (s)

$\lambda \in [1, +\infty)$ is a hyperparameter used to indicate the severity of the penalty for malicious behavior. A higher value of λ corresponds to a stronger punitive impact. In the subsequent epoch $t+1$, the FL system selects a new subset of clients for training, denoted as C^{t+1}, and disseminates the updated model parameters θ^{t+1} to the clients. Each client then locally trains on its dataset to obtain local parameters θ_k^{t+1}, resulting in a new state $\mathbf{s}^{t+1}$.

Reward: The FL system calculates the reward at step t as $r := f(\theta^t) - f(\theta^{t+1})$ based on the newly obtained global model parameters θ^{t+1}.

Computational Complexity

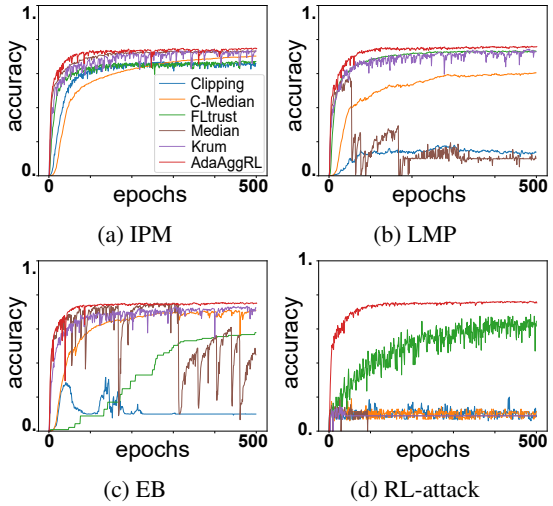
Compared to classical FL, AdaAggRL's increased computational complexity mainly stems from EnvironmentalCues step. In IG algorithm, computation involves model forward propagation, loss function calculation, and backpropagation for optimization. The computational complexity depends on model complexity M, optimization rounds max_iters, and reconstructed images num_images, totaling $O(max\_iters \times M \times num\_images)$. Extracting image features via pre-trained CNN incurs a complexity of $O(C_{CNN} \times num\_images)$. Considering MMD's constant complexity $O(C_{MMD})$, the overall computational complexity for EnvironmentalCues is $O(|C^t| \times num\_images \cdot (M \times max\_iters + C_{CNN})) + O(3|C^t| \cdot C_{MMD}) + O(1)$.

Gradient inversion simulates data distributions, but the server's goal isn't precise image reconstruction. Thus, in our experiments, gradient inversion optimization is restricted to 30 steps, with 16 dummy images. Table 1 compares AdaAggRL's training time per FL round with other algorithms. AdaAggRL's time increases by an average of 21.4% on MNIST and 50.1% on CIFAR-10 compared to baselines.

Experiments

Experimental Settings

Dataset We conduct experiments on four datasets: MNIST (LeCun et al. 1998), F-MNIST (Xiao, Rasul, and Vollgraf 2017), EMNIST (Cohen et al. 2017), and Cifar10 (Krizhevsky, Hinton et al. 2009). Addressing the non-i.i.d. challenge in FL, we follow the approach from prior work (Fang et al. 2020) by distributing training examples across all clients. Given an M-class dataset, clients are randomly divided into M groups. The probability q of assigning a training sample with label l to its respective group is set, with the probability of assigning it to other groups being $\frac{1-q}{M-1}$. Training samples within the same client group adhere to the same distribution. When $q = 1/M$, the distribution of training samples across M groups is uniform, ensuring that all clients' datasets follow the same distribution. In cases where $q > 1/M$, the datasets among clients are not identically distributed. Using MNIST dataset, we set $q = 0.5$ to distribute training samples among clients unevenly, denoted as MNIST-0.5. MNIST-0.1 represents the scenario where MNIST is evenly distributed among clients ($q = 0.1$).



(a) IPM (b) LMP

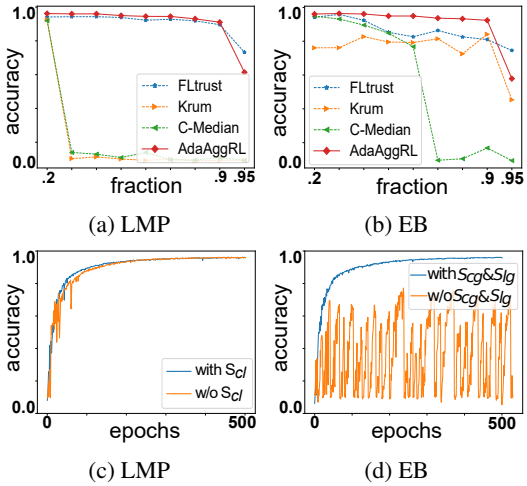
(c) EB (d) RL-attack

Figure 4: The testing accuracy variation of the global model on Cifar10 dataset under four attacks.

Metrics We assess FL defense methods by evaluating the global model's image classification accuracy after 500 epochs of FL training since these attacks aim to diminish testing accuracy. Higher accuracy of the global model under various attacks indicates stronger defense robustness.

Baselines To verify the effectiveness and stability of AdaAggRL, we mainly compare it with five other defense algorithms: Krum (Blanchard et al. 2017), coordinate-wise median (Median) (Yin et al. 2018), norm clipping (Clipping) (Sun et al. 2019), an extension of the vanilla coordinate-wise median (C-Median) where a norm clipping step is applied (Li, Sun, and Zheng 2022), and FLtrust (Cao et al. 2020). Krum filters malicious updates at the client level, Clipping performs gradient clipping on parameter updates before aggregation, Median and C-Median select the median or clipped median of individual parameter values from all local model updates as global model parameters, and FLtrust requires the server to have access to an amount of root data. To further illustrate its performance, AdaAggRL is compared with Feddefender (Park et al. 2023) and FedVal (Valadi et al. 2023) on CIFAR-10 dataset.

We consider four poisoning attacks in FL: explicit boosting (EB) (Bhagoji et al. 2019), inner product manipulation (IPM) (Xie, Koyejo, and Gupta 2020), local model poisoning attack (LMP) (Fang et al. 2020), and RL-based model attack (RL-attack) (Li, Sun, and Zheng 2022). IPM manipulates the attacker's gradients to ensure the inner product with true gradients becomes negative during aggregation. LMP generates malicious model updates by solving an optimization problem in each FL epoch. EB generates mali-



(a) LMP (b) EB

(c) LMP (d) EB

Figure 5: The testing accuracy of FL methods on MNIST-0.5 under LMP and EB as the proportion of malicious clients increases (a-b). The defense performance of AdaAggRL on MNIST-0.5 compared to the case where S_{cl} is not considered under LMP (c) and the case where S_{cg} and S_{lg} are not considered under EB (d).

cious updates through explicit enhancement, optimizing for a malicious objective designed to induce targeted misclassification. RL-attack adaptively generates attacks on the FL system using RL.

Parameter Settings For MNIST, F-MNIST, and EMNIST, a Convolutional Neural Network (CNN) serves as the global model. In the case of Cifar10, the ResNet18 architecture (He et al. 2016) is utilized as the global model. In FL, there are 100 clients, denoted as $K = 100$, with 20 malicious clients. For defense strategies based on RL, given the continuous action and state spaces, we select Twin Delayed DDPG (TD3) (Fujimoto, Hoof, and Meger 2018) algorithm to train the defense policies in experiments. Details on parameter determination are provided in the Appendix.

Defense Performances

Table 2 reports the testing accuracy of various FL aggregation methods across four datasets, indicating AdaAggRL's robustness. Across different datasets and models, AdaAggRL demonstrates consistent defensive efficacy against all four attack scenarios. Notably, AdaAggRL maintains stable effectiveness against RL-attack, where other defense methods face significant challenges and experience a noticeable degradation in defensive performance. Figure 4 illustrates the testing accuracy of the global model on Cifar10 dataset under four attacks, with different aggregation rules. AdaAggRL consistently outperforms other methods, achieving superior accuracy and convergence speed, particularly against RL-attack. Unlike FLtrust, it requires no root dataset and demonstrates more stable performance after 200 epochs, surpassing Krum and C-Median in all scenarios. AdaAggRL's performance on other datasets is shown in Appendix.

LMP and EB attempt to optimize the objective function

	EB	IPM	LMP	RL-attack
C-Median	0.9598	0.9537	0.9329	0.5550
Clipping	0.9151	0.9654	0.1944	0.5750
FLtrust	0.9231	0.9591	0.9618	0.7300
Krum	0.9325	0.7897	0.9458	0.6875
Median	0.0981	0.9549	0.1135	0.2750
AdaAggRL	**0.9659**	**0.9658**	**0.9636**	**0.9655**

(a) CNN global model, MNIST-0.1

	EB	IPM	LMP	RL-attack
C-Median	0.9466	0.9479	0.9198	0.4250
Clipping	0.7867	0.9576	0.0986	0.4900
FLtrust	0.9488	0.9414	0.9412	0.4375
Krum	0.9274	0.7608	0.9259	0.1563
Median	0.0974	0.9448	0.1032	0.1563
AdaAggRL	**0.9608**	**0.9617**	**0.9604**	**0.9559**

(b) CNN global model, MNIST-0.5

	EB	IPM	LMP	RL-attack
C-Median	0.7935	0.8123	0.7722	0.6300
Clipping	0.7249	0.8261	0.6015	0.4600
FLtrust	0.8154	0.8344	0.8386	0.6150
Krum	0.8082	0.6610	0.7942	0.5625
Median	0.1002	0.8171	0.1001	0.0938
AdaAggRL	**0.8411**	**0.8398**	**0.8400**	**0.8337**

(c) CNN global model, F-MNIST

	EB	IPM	LMP	RL-attack
C-Median	0.8780	0.8724	0.8289	0.1857
Clipping	0.6694	**0.8834**	0.0008	0.1700
FLtrust	0.8618	0.8741	0.8684	0.2400
Krum	0.8309	0.5418	0.8135	0.0312
Median	0.0388	0.8716	0.4331	0.1850
AdaAggRL	**0.8816**	0.8805	**0.8776**	**0.8786**

(d) CNN global model, EMNIST

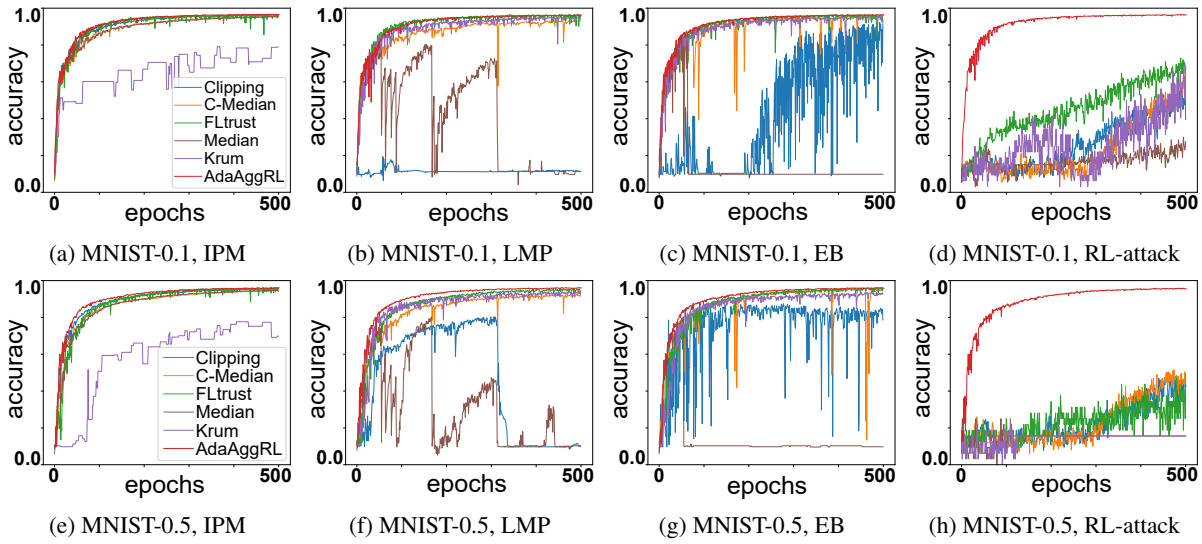
	EB	IPM	LMP	RL-attack
C-Median	0.7091	0.7364	0.6048	0.1150
Clipping	0.1002	0.6589	0.1388	0.0850
FLtrust	0.5786	0.6709	0.7312	0.6450
Krum	0.7238	0.7028	0.7283	0.0900
Median	0.4484	0.7309	0.1018	0.0900
AdaAggRL	**0.7452**	**0.7497**	**0.7583**	**0.7531**

(e) ResNet18 global model, Cifar10

Table 2: The testing accuracy of different FL aggregation methods under various attacks

to make the poisoned gradients statistically inconspicuous, while RL-attack mimics the behavior of normal clients. As a result, defense methods like Median, C-Median, and Clipping, which rely solely on mean or median information through gradient clipping or selection, are prone to misjudge poisoned gradient updates. These methods may end up clipping correct updates while preserving erroneous ones.

AdaAggRL's performance is evaluated against Feddefender and FedVal on the CIFAR-10 dataset using ResNet18



(a) MNIST-0.1, IPM (b) MNIST-0.1, LMP (c) MNIST-0.1, EB (d) MNIST-0.1, RL-attack

(e) MNIST-0.5, IPM (f) MNIST-0.5, LMP (g) MNIST-0.5, EB (h) MNIST-0.5, RL-attack

Figure 6: The performance of FL defense algorithms under different distribution conditions.

	IPM	LMP	EB	RL-attack
Feddefender	0.4711	0.4196	0.1008	0.4150
FedVal	0.6052	0.0980	0.4338	0.1392
AdaAggRL	**0.6984**	**0.7063**	**0.7143**	**0.7106**

Table 3: The accuracy of aggregation methods under attacks

over 100 FL epochs, shown in Table 3. AdaAggRL is still significantly better than the latest baselines.

Ablation Studies

Impact of Current-history Similarity To illustrate the impact of the similarity metrics S_{cl} on the stability of the FL process, Figure 5c depicts the defense performance of AdaAggRL compared to the case where S_{cl} is not considered under LMP attack. We observe that considering the variations in S_{cl} indeed enhances the convergence speed and reduces the oscillation amplitude of testing accuracy.

Impact of Current (history)-global Similarity Figure 5d illustrates the defense performance of AdaAggRL compared to the case where S_{cg} and S_{lg} are not considered under EB attack. We observe that the malicious client data distribution obtained through gradient reversal may stably deviate from the normal distribution, leading to an inflated S_{cl}. If S_{cg} and S_{lg} are not considered, the defense effectiveness degrades.

Analysis

Impact of the Number of Attackers Figures 5a and 5b show testing accuracy under LMP and EB attacks as malicious client proportion increases from 0% to 95%. Both AdaAggRL and FLtrust can tolerate up to 90% of malicious clients. AdaAggRL shows a slight accuracy decline as malicious client proportions increase, while the remaining FL aggregation algorithms can only tolerate malicious clients below 30% under LMP attack. This highlights AdaAggRL's stability against high percentages of malicious clients.

Impact of non-i.i.d. Degree Table 2b reports testing accuracy of FL defense algorithms on MNIST-0.5 under various attacks, while Figure 6 compares their performance under different distribution conditions. With non-i.i.d. data (q=0.5), baselines show reduced accuracy under LMP and increased oscillations under EB, and non-i.i.d. conditions significantly impact their performance against RL-attack. AdaAggRL demonstrates faster convergence, particularly against IPM, LMP, and EB, with minimal performance decline across attacks, demonstrating robustness to non-i.i.d. data. Under significant non-i.i.d. impact, RL can adaptively lower the weights of global model-related similarity scores. More results for q>0.5 can be found in the Appendix.

Conclusion

This paper proposes AdaAggRL, an RL-based Adaptive Aggregation method, to counter sophisticated poisoning attacks. Specifically, we first utilize distribution learning to simulate clients' data distributions. Then, we use MMD to calculate the pairwise similarity of the current local model data distribution, its historical data distribution, and the global model data distribution. Finally, we use policy learning to adaptively determine the aggregation weights based on the above similarities and the reconstruction similarity. Experiments on four real-world datasets demonstrate that AdaAggRL significantly outperforms the state-of-the-art defense model for sophisticated attacks. Future work could explore novel attacks on adaptive defense. One potential scheme to construct malicious update parameters is to solve an optimization problem under the condition of controlling the stability of simulated data distribution changes.

Acknowledgments

This work was funded by the National Natural Science Foundation of China (NSFC) under Grants No. 62406013, the Beijing Advanced Innovation Center Funds for Future Blockchain and Privacy Computing(GJJ-23-006) and the Fundamental Research Funds for the Central Universities.

References

Arbel, M.; Korba, A.; Salim, A.; and Gretton, A. 2019. Maximum mean discrepancy gradient flow. *Advances in Neural Information Processing Systems*, 32.

Bagdasaryan, E.; Veit, A.; Hua, Y.; Estrin, D.; and Shmatikov, V. 2020. How to backdoor federated learning. In *International conference on artificial intelligence and statistics*, 2938–2948. PMLR.

Baruch, G.; Baruch, M.; and Goldberg, Y. 2019. A Little Is Enough: Circumventing Defenses For Distributed Learning. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Bhagoji, A. N.; Chakraborty, S.; Mittal, P.; and Calo, S. 2019. Analyzing Federated Learning through an Adversarial Lens. In Chaudhuri, K.; and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 634–643. PMLR.

Blanchard, P.; El Mhamdi, E. M.; Guerraoui, R.; and Stainer, J. 2017. Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Byrd, D.; and Polychroniadou, A. 2020. Differentially private secure multi-party computation for federated learning in financial applications. In *Proceedings of the First ACM International Conference on AI in Finance*, 1–9.

Cao, X.; Fang, M.; Liu, J.; and Gong, N. Z. 2020. Fltrust: Byzantine-robust federated learning via trust bootstrapping. *arXiv preprint arXiv:2012.13995*.

Chaddad, A.; Wu, Y.; and Desrosiers, C. 2023. Federated Learning for Healthcare Applications. *IEEE Internet of Things Journal*, 1–1.

Cohen, G.; Afshar, S.; Tapson, J.; and van Schaik, A. 2017. EMNIST: Extending MNIST to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*, 2921–2926.

Fang, M.; Cao, X.; Jia, J.; and Gong, N. 2020. Local model poisoning attacks to {Byzantine-Robust} federated learning. In *29th USENIX security symposium (USENIX Security 20)*, 1605–1622.

Fujimoto, S.; Hoof, H.; and Meger, D. 2018. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, 1587–1596. PMLR.

Geiping, J.; Bauermeister, H.; Dröge, H.; and Moeller, M. 2020. Inverting gradients-how easy is it to break privacy in federated learning? *Advances in Neural Information Processing Systems*, 33: 16937–16947.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Li, H.; Sun, X.; and Zheng, Z. 2022. Learning to attack federated learning: A model-based reinforcement learning attack framework. *Advances in Neural Information Processing Systems*, 35: 35007–35020.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, 1273–1282. PMLR.

Nguyen, D. C.; Ding, M.; Pathirana, P. N.; Seneviratne, A.; Li, J.; and Poor, H. V. 2021. Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3): 1622–1658.

Park, J.; Han, D.-J.; Choi, M.; and Moon, J. 2021. Sageflow: Robust federated learning against both stragglers and adversaries. *Advances in neural information processing systems*, 34: 840–851.

Park, S.; Han, S.; Wu, F.; Kim, S.; Zhu, B.; Xie, X.; and Cha, M. 2023. Feddefender: Client-side attack-tolerant federated learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1850–1861.

Rajput, S.; Wang, H.; Charles, Z.; and Papailiopoulos, D. 2019. DETOX: A redundancy-based framework for faster and more robust gradient aggregation. *Advances in Neural Information Processing Systems*, 32.

Rudin, L. I.; Osher, S.; and Fatemi, E. 1992. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4): 259–268.

Sattler, F.; Müller, K.-R.; Wiegand, T.; and Samek, W. 2020. On the byzantine robustness of clustered federated learning. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 8861–8865. IEEE.

Shejwalkar, V.; and Houmansadr, A. 2021. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. In *NDSS*.

Shejwalkar, V.; Houmansadr, A.; Kairouz, P.; and Ramage, D. 2022. Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning. In *2022 IEEE Symposium on Security and Privacy (SP)*, 1354–1371. IEEE.

Sun, Z.; Kairouz, P.; Suresh, A. T.; and McMahan, H. B. 2019. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*.

Valadi, V.; Qiu, X.; De Gusmão, P. P. B.; Lane, N. D.; and Alibeigi, M. 2023. FedVal: different good or different bad in federated learning. In *Proceedings of the 32nd USENIX Conference on Security Symposium*, 6365–6380.

Wang, W.; Li, H.; Ding, Z.; Nie, F.; Chen, J.; Dong, X.; and Wang, Z. 2021. Rethinking maximum mean discrepancy for visual domain adaptation. *IEEE Transactions on Neural Networks and Learning Systems*.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Xie, C.; Koyejo, O.; and Gupta, I. 2020. Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation. In *Uncertainty in Artificial Intelligence*, 261–270. PMLR.

Xie, C.; Koyejo, S.; and Gupta, I. 2019. Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance. In *International Conference on Machine Learning*, 6893–6901. PMLR.

Yamany, W.; Moustafa, N.; and Turnbull, B. 2021. OQFL: An optimized quantum-based federated learning framework for defending against adversarial attacks in intelligent transportation systems. *IEEE Transactions on Intelligent Transportation Systems*.

Yin, D.; Chen, Y.; Kannan, R.; and Bartlett, P. 2018. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, 5650–5659. PMLR.

Zhang, Z.; Cao, X.; Jia, J.; and Gong, N. Z. 2022. FLDetector: Defending federated learning against model poisoning attacks via detecting malicious clients. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2545–2555.

Zheng, J.-Y.; Zhang, H.; Wang, L.; Qiu, W.; Zheng, H.-W.; and Zheng, Z.-M. 2024. Safely Learning with Private Data: A Federated Learning Framework for Large Language Model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 5293–5306.