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

Yujing Wang<sup>†‡</sup>

Hainan Zhang†\*

Sijia Wen†

Wangjie Qiu<sup>†‡</sup>

Binghui Guo†

†Beijing Advanced Innovation Center for Future Blockchain and Privacy Computing, School of Artificial Intelligence, Beihang University † Beijing Academy of Blockchain and Edge Computing {eugenia,zhanghainan,wen\_sijia,wangjieqiu,guobinghui}@buaa.edu.cn

#### **Abstract**

Federated learning is highly 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 RLbased Adaptive Aggregation method, to defend against sophisticated poisoning attacks. Specifically, we first utilize distribution learning to simulate the clients data distributions. Then, we use the 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.

# 1 Introduction

Federated Learning (FL) is a promising machine learning method where model training is distributed across multiple local devices rather than centralized on a single server. It can not only ensure the security of users' private data but also enhance model performance by collaborating diverse data from various local sources. It finds applications in diverse scenarios, including smart healthcare [8], financial services [6], IoT [21], and intelligent transportation [35]. However, FL systems are vulnerable, and the performance of the aggregated model is highly susceptible to model poisoning attacks from unknown clients, 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

<sup>\*</sup>Corresponding author

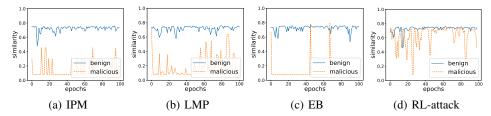


Figure 1: The statistical results of the similarity between the current client data distribution and its historical data distributions under four types of attacks vary with the training epochs on MNIST dataset. The x-axis denotes the number of client update rounds, and the y-axis represents the similarity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

defense methods are primarily targeted at manually crafted myopic attack strategies, and their defense stability is noticeably lacking when facing advanced attacks. For example, Li et al. [17] 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, as shown in Figure 3. Therefore, it is urgent to develop adaptive defenses against such learnable advanced poisoning attacks.

Most benign clients exhibit significant data distribution stability in federated learning. In normal cases, the current training data distribution simulated by the parameters passed through the client should align with its historical simulated data distribution. This is because benign clients typically employ random sampling and undergo multi-round training based on their local data, ensuring the stability of the training data distribution used for global updates. 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 [34], LMP [10], EB [4] and RL-based attacks [17], measuring the similarity of their data distribution with history after each round of updates on MNIST dataset, 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 E. Therefore, malicious clients can be recognized by observing the stability of their simulated data distribution.

In this paper, we propose an RL-based Adaptive Aggregation method, AdaAggRL, to thwart sophisticated poisoning attacks. It determines the 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 the above three distribution similarities and the reconstruction similarity of distribution learning.

Experimental results on four real-world datasets demonstrate that the proposed AdaAggRL defense method significantly outperforms existing defense approaches [5, 36, 29] and achieves a more stable global model accuracy even facing sophisticated attacks, such as RL-based attacks [17].

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 federated learning, 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.

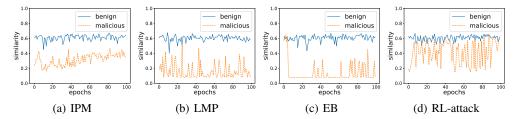


Figure 2: The statistical results of the similarity between the current client data distribution and the global model data distribution under four attacks vary with the training epochs on MNIST dataset.

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.

#### 2 Motivation

In most cases, benign clients' data distribution remains stable, while malicious clients lack regularity. 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. Though the stability of advanced attack RL-attack is significantly higher than other ordinary attack methods, there are still significant differences compared to benign clients. We also observe the same phenomenon on NLP tasks, as shown in Appendix E.

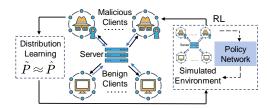
Moreover, we observe a certain degree of similarity between current client data distribution and global model data distribution, as shown in Figure 2. The data distribution of benign clients for current steps is similar to the global model and remains stable, but the similarity of malicious clients is lower. 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. Though the advanced RL-attack has high similarity with the global model due to collecting the data distribution of the server, it is still not as stable as benign clients. 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 C.

Considering that the quality of distribution learning greatly affects the accurate calculation of these three 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 obtained from distribution learning and the gradients updated by clients. Higher reconstruction similarity indicates a higher level of confidence in current distribution learning.

#### 3 RL-based Adaptive Aggregation Methods

#### 3.1 Task Definition

In the context of FL [20], 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 data 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



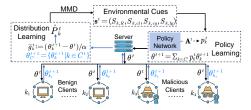


Figure 3: An overview of RL-attack

Figure 4: An overview of our AdaAggRL

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  $\theta^t$ . The chosen clients train on their local datasets, updating parameters as  $\theta_k^{t+1} = \theta^t - \alpha \nabla F_k(\theta)$ , where  $\alpha$  is the learning rate. Then they upload the updated model parameters. The server aggregates the received parameters according to a specific aggregation rule, denoted as Aggr, to obtain the new global model parameters  $\theta^{t+1} = Aggr(\theta_{Ct}^{t+1})$ .

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.

#### 3.2 Framework Overview

In AdaAggRL framework (see Algorithm in Appendix A), the server determines the weights for local model aggregation by assessing the stability of client data distributions, as shown in Figure 4. Firstly, we employ distribution learning to simulate the client's data distribution  $\hat{P}_k^t$  based on the locally uploaded model parameters  $\theta_k^{t+1}$ . 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.

#### 3.3 Distribution Learning

According to local model parameters  $\theta_k^{t+1}$  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 [12] 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  $\bar{g}_k^t \coloneqq (\theta_k^{t+1} - \theta^t)/\alpha$ . The server then solves the following optimization problem with a batch of randomly generated dummy data and labels  $D_{dummy}$ :  $\min_{D_{\text{dummy}}} 1 - \frac{\langle \nabla_{\theta} F_{D_{\text{dummy}}}(\theta_k^{t+1}).\bar{g}_k^t \rangle}{\|\nabla_{\theta} F_{D_{\text{dummy}}}(\theta_k^{t+1})\|.\|\bar{g}_k^t\|} + \frac{\beta}{B'} \Sigma_{(x,y) \in D_{\text{dummy}}} \text{TV}(x), \text{ where } \beta \text{ is a fixed parameter, } B' \text{ is the size of the dummy data batch, } F_{D_{\text{dummy}}}(\theta) = \frac{1}{B'} \sum_{(x,y) \in D_{\text{dummy}}} l(\theta,x,y), \text{ TV calculates the total variation}[25]. \text{ While solving the optimization problem, } D_{dummy} \text{ is continuously updated. The optimization terminates after } \max_{i} let rs iterations, \text{ then outputs the updated data as the reconstructed data samples } D_{\text{rec}}, \text{ and the reconstruction similarity of client } k, \text{ denoted as } S_{k,R} = \frac{\langle \nabla_{\theta} F_{D_{\text{rec}}}(\theta_k^{t+1}).\bar{g}_k^t \rangle}{\|\nabla_{\theta} F_{D_{\text{rec}}}(\theta_k^{t+1}).\bar{g}_k^t \|}.$ 

## 3.4 Environmental Cues

The distribution of samples is extracted by employing a pre-trained CNN to convert the image samples  $D_{\rm rec}$  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) [1, 31] as  $d_{k,cl} = {\rm MMD}(V_k^{\rm current}, V_k^{\rm history}) \in [0,+\infty)$ . Here,  $V_k^{\rm current}$  and  $V_k^{\rm history}$  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  $d_{k,cg}=\mathrm{MMD}(V_k^{\mathrm{current}},V_g)$ . Similarly, the dissimilarity between the historical data distribution and the global one is  $d_{k,lg}=\mathrm{MMD}(V_k^{\mathrm{history}},V_g)$ . 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.

#### 3.5 Actions Learning

The server dynamically adapts the aggregation weights based on three obtained 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 [11].

**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

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

Then  $\mathbf{w} = f_{\delta}(\tilde{\mathbf{w}})$ . The function  $f_{\delta}$  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  $\delta$  within  $f_{\delta}$  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,

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

 $\lambda \in [1, +\infty)$  is a hyperparameter used to indicate the severity of the penalty for malicious behavior. A higher value of  $\lambda$  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  $\theta^{t+1}$  to the clients. Each client then locally trains on its dataset to obtain local parameters  $\theta^{t+1}_k$ , 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  $\theta^{t+1}$ .

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

	AdaAggRL	Krum	Median	C-Median	FLtrust	Clipping
MNIST	1.6520	1.7438	1.0690	1.5556	1.1066	1.3265
Cifar10	7.2009	8.6983	3.5374	4.4695	3.5892	3.6956

#### 3.6 Computational Complexity

Compared to classical FL, AdaAggRL's increased computational complexity mainly stems from the Environmental Cues 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 Environmental Cues 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.

# 4 Experiments

#### 4.1 Experimental Settings

#### 4.1.1 Dataset

We conduct experiments on four datasets: MNIST [16], F-MNIST [32], EMNIST [9], and Cifar10 [15]. Addressing the non-i.i.d. challenge in FL, we follow the approach from prior work [10] 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).

#### 4.1.2 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. The classification accuracy is the ratio of correctly predicted test examples to the total number of test examples. Higher testing accuracy of the global model under various attacks indicates stronger robustness of the defense.

#### 4.1.3 Baselines

We compare AdaAggRL with five other defense algorithms: Krum [5], coordinate-wise median (Median) [36], norm clipping (Clipping) [29], an extension of the vanilla coordinate-wise median (C-Median) where a norm clipping step is applied [17], and FLtrust [7]. 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 a small amount of root data.

We consider four poisoning attacks in FL: explicit boosting (EB) [4], inner product manipulation (IPM) [34], local model poisoning attack (LMP) [10], and RL-based model attack (RL-attack) [17]. IPM manipulates the attacker's gradients to ensure the inner product with true gradients becomes

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

#### (a) CNN global model, MNIST-0.1 (b) CNN global model, MNIST-0.5 (c) CNN global model, F-MNIST

·	_					_	_					` '	_			
	EB	IPM	LMP	RL-attac	k		EB	IPM	LMP	RL-attack			EB	IPM	LMP	RL-attac
C-Median	0.9598	0.9537	0.9329	0.5550		C-Median	0.9466	0.9479	0.9198	0.4250		C-Median	0.7935	0.8123	0.7722	0.6300
Clipping	0.9151	0.9654		0.5750		Clipping	0.7867	0.9576	0.0986	0.4900		Clipping	0.7249	0.8261	0.6015	0.4600
Ltrust	0.9231	0.9591	0.9618	0.7300		Ltrust	0.9488	0.9414	0.9412	0.4375		FLtrust	0.8154	0.8344	0.8386	0.6150
Krum	0.9325	0.7897	0.9458	0.6875	ŀ	Krum	0.9274	0.7608	0.9259	0.1563		Krum	0.8082	0.6610	0.7942	0.5625
Aedian	0.0981	0.9549	0.1135	0.2750	N	Median	0.0974	0.9448	0.1032	0.1563		Median	0.1002	0.8171	0.1001	0.0938
AdaAggRL	0.9659	0.9658	0.9636	0.9655	A	AdaAggRL	0.9608	0.9617	0.9604	0.9559		AdaAggRL	0.8411	0.8398	0.8400	0.8337
		_	C Madian	EB	IPM 0.8724	LMP	RL-attack		Madian	EB 0.7001	IPM	LMP	RL-attack	_		
		_												_		
		_														
			C-Median	0.8780	0.8724	0.8289	0.1857		-Median	0.7091	0.736		0.1150			
			Clipping	0.6694	0.8834	0.0008	0.1700		lipping	0.1002	0.658		0.0850			
			FLtrust Krum	0.8618	0.8741	0.8684 0.8135	0.2400 0.0312		Ltrust	0.5786 0.7238	0.670		0.6450 0.0900			
			Median	0.0388	0.8716	0.4331	0.0312		fedian	0.7238	0.730		0.0900			
			AdaAggRL	0.8816	0.8805	0.8776	0.8786		daAggRL	0.7452	0.749		0.7531			
		_	AddAggKL	0.0010	0.0005	0.0770	0.0700		danggitt	0.7432	0.747	0.7505	0.7551	_		
1.0				1.0				7 1.0	· —				1.0			- Clipping
																- C-Media
0.8		umarana ana		0.8	-			0.8	31	E SHOW THE PARTY OF	·		0.8			- FLtrust - Median
	A STATE OF THE PARTY OF THE PAR	Marina M	Alternative code	<b>&gt;</b>	James Holle	March Johnson	Ludy A.	<b> </b>	450	A STATE OF THE PARTY OF THE PAR	The same	A Marcalle	S /	W.	1	Krum
0.6				cuacy	PX ' 🛴			cuacy	51 /	_ '   '	'اـــ	11	O.6- M	dile	a de de la Carte de la car	— AdaAggi
1 11/1/				3 I 🛭	•			1 3	1 1/1 1/1		در 7	YIY Maala	3 I f	Million .	All Laborators	P)

Figure 5: The testing accuracy variation of the global model on Cifar10 dataset under four attacks.

(c) EB

(d) RL-attack

(b) LMP

negative during aggregation. LMP generates malicious model updates by solving an optimization problem in each FL epoch. EB generates malicious 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.

## 4.1.4 Parameter Settings

(a) IPM

We train diverse global models on distinct datasets to showcase the versatility of AdaAggRL. Specifically, for MNIST, F-MNIST, and EMNIST, a Convolutional Neural Network (CNN) serves as the global model. In the case of Cifar10, the ResNet18 architecture [13] is utilized as the global model.

For the FL system, there are 100 clients, denoted as K = 100, with 20 malicious clients. The FL system trains for 500 epochs, wherein each epoch, the server randomly selects 10% of the clients to participate. Each client conducts one round of training on its local dataset with a learning rate of 0.05.

For defense strategies based on RL, given the continuous action and state spaces, we select Twin Delayed DDPG (TD3) [11] algorithm to train the defense policies in experiments. We configure parameters such that the policy model uses 'MlpPolicy', the learning rate is set to  $1\times 10^{-5}$  for the Adam optimizer, the batch size is 64 for each gradient update, and the discount factor  $\gamma$  is set to 0.99. We maintain fixed initial models and random seeds for data sampling to ensure a fair comparison.

#### 4.2 Defense Performances

We demonstrate our experiment results on four datasets. Table 2 presents the testing accuracy of different FL aggregation methods. The results indicate that AdaAggRL achieves the desired robustness. Across different datasets and models, AdaAggRL demonstrates consistent defensive efficacy against all four attack scenarios, while other defense methods may fail under certain attacks. Particularly against RL-attack, where other defense methods face significant challenges and experience a noticeable degradation in defensive performance, AdaAggRL maintains stable effectiveness.

Figure 5 illustrates the testing accuracy variation of the global model on Cifar10 dataset, considering four attacks. The server employs different aggregation rules. AdaAggRL outperforms in all settings, even without the need for a root dataset to guide model aggregation, unlike FLtrust. While Krum and C-Median achieve commendable accuracy against IPM, LMP, and EB attacks, AdaAggRL exhibits more stable performance after 200 epochs. Particularly against RL-attack, AdaAggRL demonstrates a

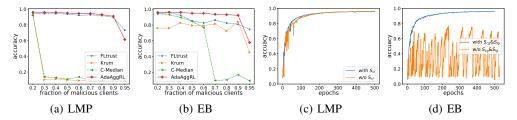


Figure 6: The testing accuracy of FL methods on MNIST-0.5 under LMP and EB as the proportion of malicious clients increases (a-b). The defense performance of AdaAggRL on MNIST-0.5 compared to the case where  $S_{cl}$  is not considered under LMP (c) and the case where  $S_{cg}$  and  $S_{lg}$  are not considered under EB (d).

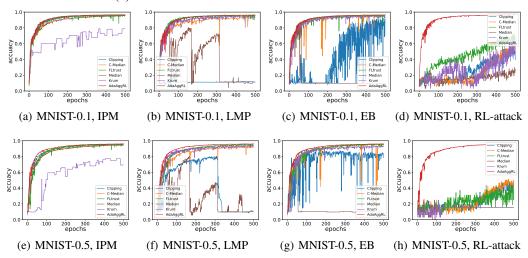


Figure 7: The performance of FL defense algorithms under different distribution conditions.

distinct advantage with superior accuracy and convergence speed compared to other defense methods. AdaAggRL's performance on other datasets is shown in Appendix C.

## 4.3 Ablation Studies

#### 4.3.1 Impact of Current-history Similarity

To illustrate the impact of the similarity metrics  $S_{cl}$  between the current client and historical data distribution on the stability of the FL process, Figure 6(c) 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.

#### 4.3.2 Impact of Current (history)-global Similarity

Figure 6(d) 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, it can significantly degrade the defense effectiveness.

#### 4.4 Analysis

# 4.4.1 Impact of the Number of Attackers

Figure 6(a) and Figure 6(b) illustrate the testing accuracy of different FL methods under LMP and EB attacks as the proportion of malicious clients increases from 0% to 95%. We observe that, under the current attacks, both AdaAggRL and FLtrust can tolerate up to 90% of malicious clients.

Although, for AdaAggRL, there is a slight decrease in testing accuracy as the proportion of malicious clients increases. In contrast, the remaining FL aggregation algorithms can only tolerate malicious clients below 30% under LMP attack. This indicates that AdaAggRL can maintain a relatively stable defensive performance even when facing a high percentage of malicious clients.

#### 4.4.2 Impact of non-i.i.d. Degree

Table 2(b) presents testing accuracy of various FL defense algorithms against diverse attacks on MNIST-0.5. Figure 7 provides a comparative analysis of the performance of FL defense algorithms under different distribution conditions. Under non-i.i.d. condition (q=0.5), AdaAggRL's advantage in faster convergence becomes more pronounced, particularly against IPM, LMP, and EB. In the case of LMP, baseline algorithms exhibit a certain degree of accuracy reduction, and under EB, the oscillation magnitude of C-Median's testing accuracy increases. Especially against RL-attack, the impact of non-i.i.d. on baseline algorithms is significant, while AdaAggRL maintains relatively stable performance. This indicates the robustness of AdaAggRL in handling non-i.i.d. client datasets.

#### 5 Related Work

#### **5.1 Poisoning Attacks**

Based on the attacker's objectives, poisoning attacks can be classified into targeted poisoning attacks aiming to misclassify specific input sets [4, 3, 2] and untargeted attacks aimed at reducing the overall accuracy of the global model [10, 34, 27]. Current untargeted attack methods typically employ heuristic-based approaches [34] or optimize myopic objectives [10, 27, 28]. Bhagoji et al. [4] generate malicious updates through explicit enhancement, optimizing for a malicious objective strategically designed to induce targeted misclassification. Xie et al. [34] manipulate the attacker's gradients to ensure the inner product with the true gradients becomes negative. Fang et al. [10] generate malicious updates by solving an optimization problem. However, these attack methods typically 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 et al. [17] propose a model-based RL framework for guiding untargeted poisoning attacks in FL system. They utilize model updates from the server to approximate the server's data distribution, subsequently employing the learned distribution as a simulator for FL environment. Based on the simulator's behavior, they utilize RL to automatically generate effective attacks, resulting in significantly reducing the global model's accuracy. Even when the server employs robust aggregation rules, this customized method can still maintain a high level of attack effectiveness.

## 5.2 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 [7, 26, 39], and the other is based on designing novel Byzantine fault-tolerant aggregation algorithms using mathematical statistics to improve the robustness of aggregation [5, 36, 29, 24, 33]. In evaluation mechanisms, Sattler et al. [26] 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 [7, 22] require the server to collect a portion of clean data as a basis for validating model updates. In robust aggregation algorithms, statistical methods [5, 36, 29] compare local updates and remove statistical outliers before updating the global model. For example, Blanchard et al. [5] employ the square-distance metric to measure distances among local updates and select the local update with the minimum distance as the global model parameters. Yin et al. [36] sort the values of parameters 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. [29] 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 [4, 10].

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.

## 6 Conclusion

In this paper, we propose 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.

Limitations & Future Work AdaAggRL assumes that the simulated data distribution obtained from poisoned updates is unstable. However, if malicious clients can launch attacks based on a stable distribution, AdaAggRL may struggle to identify malicious updates by computing the similarity of simulated distributions. Therefore, future work will focus on addressing defense strategies against such covert poisoning attacks.

## References

- [1] Michael Arbel, Anna Korba, Adil Salim, and Arthur Gretton. Maximum mean discrepancy gradient flow. *Advances in Neural Information Processing Systems*, 32, 2019.
- [2] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International conference on artificial intelligence and statistics*, pages 2938–2948. PMLR, 2020.
- [3] Gilad Baruch, Moran Baruch, and Yoav Goldberg. A little is enough: Circumventing defenses for distributed learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
- [4] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 634–643. PMLR, 09–15 Jun 2019.
- [5] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- [6] David Byrd and Antigoni Polychroniadou. Differentially private secure multi-party computation for federated learning in financial applications. In *Proceedings of the First ACM International Conference on AI in Finance*, pages 1–9, 2020.
- [7] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. Fltrust: Byzantine-robust federated learning via trust bootstrapping. *arXiv* preprint arXiv:2012.13995, 2020.
- [8] Ahmad Chaddad, Yihang Wu, and Christian Desrosiers. Federated learning for healthcare applications. *IEEE Internet of Things Journal*, pages 1–1, 2023. doi: 10.1109/JIOT.2023. 3325822.
- [9] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. Emnist: Extending mnist to handwritten letters. In 2017 International Joint Conference on Neural Networks (IJCNN), pages 2921–2926, 2017. doi: 10.1109/IJCNN.2017.7966217.
- [10] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local model poisoning attacks to {Byzantine-Robust} federated learning. In 29th USENIX security symposium (USENIX Security 20), pages 1605–1622, 2020.
- [11] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

- [12] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradientshow easy is it to break privacy in federated learning? *Advances in Neural Information Processing Systems*, 33:16937–16947, 2020.
- [13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [14] Yoon Kim. Convolutional neural networks for sentence classification. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1181. URL https://aclanthology.org/D14-1181.
- [15] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.
- [16] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [17] Henger Li, Xiaolin Sun, and Zizhan Zheng. Learning to attack federated learning: A model-based reinforcement learning attack framework. Advances in Neural Information Processing Systems, 35:35007–35020, 2022.
- [18] Yuping Luo, Huazhe Xu, Yuanzhi Li, Yuandong Tian, Trevor Darrell, and Tengyu Ma. Algorithmic framework for model-based deep reinforcement learning with theoretical guarantees. *arXiv preprint arXiv:1807.03858*, 2018.
- [19] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P11-1015.
- [20] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.
- [21] Dinh C Nguyen, Ming Ding, Pubudu N Pathirana, Aruna Seneviratne, Jun Li, and H Vincent Poor. Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3):1622–1658, 2021.
- [22] Jungwuk Park, Dong-Jun Han, Minseok Choi, and Jaekyun Moon. Sageflow: Robust federated learning against both stragglers and adversaries. *Advances in neural information processing systems*, 34:840–851, 2021.
- [23] Sungwon Park, Sungwon Han, Fangzhao Wu, Sundong Kim, Bin Zhu, Xing Xie, and Meeyoung Cha. Feddefender: Client-side attack-tolerant federated learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1850–1861, 2023.
- [24] Shashank Rajput, Hongyi Wang, Zachary Charles, and Dimitris Papailiopoulos. Detox: A redundancy-based framework for faster and more robust gradient aggregation. *Advances in Neural Information Processing Systems*, 32, 2019.
- [25] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.
- [26] Felix Sattler, Klaus-Robert Müller, Thomas Wiegand, and Wojciech Samek. On the byzantine robustness of clustered federated learning. In *ICASSP* 2020-2020 *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8861–8865. IEEE, 2020.
- [27] Virat Shejwalkar and Amir Houmansadr. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. In *NDSS*, 2021.

- [28] Virat Shejwalkar, Amir Houmansadr, Peter Kairouz, and Daniel Ramage. Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning. In 2022 IEEE Symposium on Security and Privacy (SP), pages 1354–1371. IEEE, 2022.
- [29] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. Can you really backdoor federated learning? arXiv preprint arXiv:1911.07963, 2019.
- [30] Viktor Valadi, Xinchi Qiu, Pedro Porto Buarque De Gusmão, Nicholas D Lane, and Mina Alibeigi. Fedval: different good or different bad in federated learning. In *Proceedings of the 32nd USENIX Conference on Security Symposium*, pages 6365–6380, 2023.
- [31] Wei Wang, Haojie Li, Zhengming Ding, Feiping Nie, Junyang Chen, Xiao Dong, and Zhihui Wang. Rethinking maximum mean discrepancy for visual domain adaptation. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
- [32] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.
- [33] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance. In *International Conference on Machine Learning*, pages 6893–6901. PMLR, 2019.
- [34] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation. In *Uncertainty in Artificial Intelligence*, pages 261–270. PMLR, 2020.
- [35] Waleed Yamany, Nour Moustafa, and Benjamin Turnbull. Oqfl: An optimized quantum-based federated learning framework for defending against adversarial attacks in intelligent transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, 2021.
- [36] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, pages 5650–5659. PMLR, 2018.
- [37] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.
- [38] Xuezhou Zhang, Yuzhe Ma, Adish Singla, and Xiaojin Zhu. Adaptive reward-poisoning attacks against reinforcement learning. In *International Conference on Machine Learning*, pages 11225–11234. PMLR, 2020.
- [39] Zaixi Zhang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. 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*, pages 2545–2555, 2022.

# A Method pseudocode

## Algorithm 1 AdaAggRL

```
Input: K clients; learning rate \alpha; number of global iterations R_q; number of local iterations R_l
Output: Global model \theta
\theta^0 \leftarrow random initialization.
for t = 0, 1, \dots, R_g do
    // Transition of environmental state
    The server randomly samples clients subset C^t and sends \theta^t to them.
    for k \in C^t do
       \theta_k^{t+1} \leftarrow \begin{cases} \theta^t - \alpha \nabla F_k(\theta) & \text{Benign clients} \\ * & \text{Malicious clients} \end{cases}
        send \boldsymbol{\theta}_k^{t+1} to the server
    end for
   \mathbf{s}^t \leftarrow EnvironmentalCues(\theta^t, \theta_{C^t}^{t+1})
   // Obtain actions
    \mathbf{A}^t = (\mathbf{a}^t, b^t) \leftarrow Actions(\mathbf{s}^t)
    \tilde{\mathbf{w}} \leftarrow q(\mathbf{s}^t \cdot \mathbf{a}^t)
   // Calculate the reward
    \delta \leftarrow \max(\tilde{\mathbf{w}}) \cdot b^t
    \mathbf{w} = f_{\delta}(\tilde{\mathbf{w}})
   \theta^{t+1} \leftarrow \sum_{k \in C^t} w_k / \lambda^{h_k^{t+1}} \cdot \theta_k^{t+1} \left\{ \lambda^{h_k^{t+1}} \right\} indicates the penalty for malicious behavior.
    reward \leftarrow f(\theta^t) - f(\theta^{t+1})
end for
\mathbf{return}\;\theta=\theta^{t+1}
```

# Algorithm 2 EnvironmentalCues

 $\mathbf{s}^t \leftarrow (s^t_{k_1}, s^t_{k_2}, ..., s^t_{k_{|C^t|}})^T$ 

return  $s^t$ 

```
\begin{array}{l} \theta_{C^t} \\ \textbf{Output:} \ \text{Environment state } \mathbf{s}^t \\ \textbf{for } k \in C^t \ \textbf{do} \\ \bar{g}_k^t \leftarrow (\theta_k^{t+1} - \theta^t)/\alpha \\ D_{rec,k}, S_{k,R}^t \leftarrow \text{IG}(\bar{g}_k^t) \\ V_k^{history} \leftarrow V_k^{current} \\ V_k^{current} \leftarrow \text{CNN}(D_{rec,k}) \\ \textbf{end for} \\ V_g \leftarrow \text{average}(\{V_k^{current}\}) \\ \textbf{for } k \in C^t \ \textbf{do} \\ S_{k,cl} \leftarrow 2 \cdot \cos(\tanh(\text{MMD}(V_k^{current}, V_k^{history})/2)) - 1 \\ S_{k,cg} \leftarrow 2 \cdot \cos(\tanh(\text{MMD}(V_k^{current}, V_g)/2)) - 1 \\ S_{k,lg} \leftarrow 2 \cdot \cos(\tanh(\text{MMD}(V_k^{history}, V_g)/2)) - 1 \\ s_k^t \leftarrow (S_{k,R}, S_{k,cl}, S_{k,cg}, S_{k,lg}) \\ \textbf{and for.} \end{array}
```

**Input:** Globel model parameters  $\theta^t$ ; The set of client model parameters participating in the training

Algorithm 1 presents the comprehensive AdaAggRL approach developed in our study. AdaAggRL is reinforcement learning-based, where the Federated Learning (FL) system undergoes three processes in each epoch: environmental state transition, acquiring actions based on the policy model, and computing rewards for the current step. The calculation of the environmental state, based on the model parameters uploaded by clients, is accomplished through the EnvironmentalCues method outlined in Algorithm 2. The Actions function utilizes reinforcement learning to obtain the policy based on the current environmental state.

In the EnvironmentalCues method of Algorithm 2, IG denotes the application of the gradient reversal technique for reshaping the dataset distribution and computing the reconstruction similarity. CNN represents a convolutional neural network employed for extracting image features, while MMD indicates the utilization of Maximum Mean Discrepancy (MMD) to quantify the distribution disparity between features.

# **B** Details of Experimental Settings

In training setup, the RL attack model first learns AD hoc attack strategies against other baseline defense methods, and then fixes the parameters. In response, we developed a temporary defense strategy to defend against this learned attack strategy. On this basis, the RL attack model refines the attack strategy against RL defense according to our AD hoc defense strategy. We then implement defensive measures based on this improved attack strategy and evaluate the effectiveness of our defense mechanisms. Despite the attacker's awareness of the server's utilization of RL-based defense, our algorithm remains capable of detecting fluctuations in its simulated data distribution, thus identifying malicious behavior and diminishing the attacker's involvement in FL training, ultimately leading to their withdrawal from the training process.

In each round of FL training, we reconstruct 16 images from a single model update of a sampled client to simulate data distribution. The optimization process for gradient inversion is carried out for 30 iterations. However, it's important to note that the reconstructed images are solely used to capture the distribution characteristics and are not intended for precise image reconstruction. The limited number of iterations and reconstructed images do not suffice for accurately restoring image information. Consequently, the reconstructed images do not contain discernible image details. This approach is implemented to fulfill the privacy protection requirements of FL while minimizing computational time as much as possible. The experimental results demonstrate that, under the LMP attack, when reconstructing 16 images from the MNIST dataset, the final accuracy achieves after 10 optimization iterations of gradient inversion is 0.9554, after 20 iterations is 0.9592, after 30 iterations is 0.9636, and after 50 iterations is 0.9664. When reconstructing 32 images, the final accuracy after 30 optimization iterations is 0.9628. To strike a balance between the final performance and computational resources, we opt to train using gradient inversion with 30 iterations for reconstructing 16 images. The method for gradient inversion optimization is Adam, the learning rate is 0.05, and the reconstructed images are initialized to all-zero tensors. And the fixed parameter  $\beta$  in the optimization target is set to 1e-4.

## C Supplementary Experimental Results

Figure 8 illustrates the test accuracy variations over FL epochs for different FL aggregation methods under four attacks (IPM, LMP, EB, and RL-attack) on F-MNIST dataset. We observe that AdaAggRL achieves higher accuracy and more stable convergence, especially when facing the more sophisticated RL-attack.

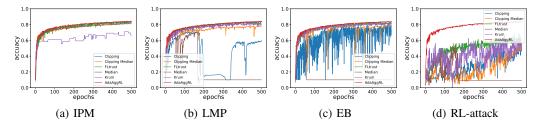


Figure 8: The variation in the testing accuracy of the global model over FL epochs on F-MNIST dataset considering different attacks.

Figure 9 illustrates the test accuracy variations over FL epochs for different FL aggregation methods under four attacks (IPM, LMP, EB, and RL-attack) on EMNIST dataset. We observe that AdaAggRL exhibits faster convergence under IPM, achieves higher accuracy under LMP and EB, and demonstrates particularly stable training outcomes and increased accuracy when faced with RL-attack.

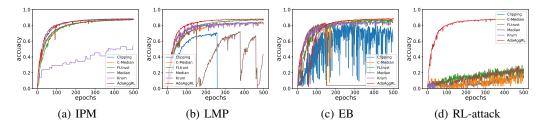


Figure 9: The variation in the testing accuracy of the global model over FL epochs on EMNIST dataset considering different attacks.

Figure 10 illustrates the test accuracy variations over FL epochs for AdaAggRL and FedAvg under no attacks. We observe that in the absence of attacks, AdaAggFL and FedAvg perform similarly on MNIST and F-MNIST datasets. However, in the presence of non-i.i.d. data (MNIST-0.5), AdaAggRL achieves a slightly higher test accuracy than FedAvg after 500 rounds of training.

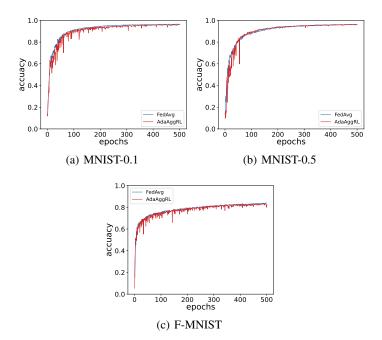


Figure 10: The comparison of test accuracy between FedAvg and AdaAggRL under attack-free conditions on different datasets.

We observe a certain degree of similarity between the historical client data distribution and the global model data distribution, as shown in Figure 11. We find that the data distribution of benign clients for historical steps is similar to the global model and remains stable, but the similarity of malicious clients is lower. This is because the data distribution of the global model represents the average state of normal data, so the distribution of benign clients should always be consistent with the global model. Instead, malicious clients do not possess this property. Though the advanced RL-attack has a high similarity with the global model due to collecting the data distribution of the server, it is still not as stable as the benign client. While considering the similarity between the current client data distribution and the global model data distribution, we also take into account the similarity between historical client data distribution and global model data distribution. By incorporating the historical performance of clients, this approach aims to capture the subtle behaviors of lurking malicious clients engaging in attacks.

To illustrate the impact of the similarity metrics  $S_R$  between the current client data distribution and historical data distribution on the stability of the federated learning process, Figure 12 depicts the defense performance of AdaAggRL compared to the case where  $S_R$  is not considered under LMP

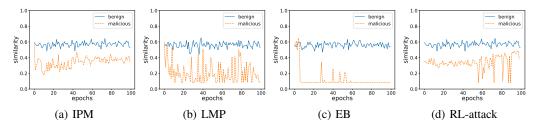


Figure 11: The statistical results of the similarity between the historical client data distribution and the global model data distribution under four types of attacks vary with the training epochs on MNIST dataset.

attack. We observe that considering the variations in  $S_R$  indeed enhances the convergence speed and reduces the oscillation amplitude of testing accuracy.

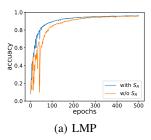


Figure 12: The defense performance of AdaAggRL compared to the case where  $S_R$  is not considered under LMP attack

# D Comparison with Advanced Baselines

To demonstrate the effectiveness of AdaAggRL, we also compare it with more novel baselines, Feddefender [23] and FedVal [30], on the CIFAR-10 dataset. Using ResNet18 as the base model and a learning rate of 0.01, we conducted FL for 100 epochs. We records the accuracy, as shown in the table 3. We can see that our AdaAggRL method is still significantly better than the latest proposed baselines.

Table 3: The testing accuracy of different FL aggregation methods under various attacks

	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

# E NLP Dataset Distribution Changes Caused by Poison Attacks

In order to verify whether the hypothesis that poisoning attacks can cause abnormal changes in the distribution of datasets reconstructed by gradient inversion is valid on NLP datasets, we select IMDB review dataset [19] on TextCNN [14] framework for FL semantic classification. IPM, LMP, EB and RL-attack poisoning attacks are also carried out during training. Unlike the image data set, we record and observe changes in the distribution of embedding. Figure 13 shows the statistical results of the similarity between the current client data distribution and its historical data distributions under four types of attacks that vary with the training epochs. It can be seen that on IMDB dataset, under different poisoning attacks, the similarity between the historical embedding distribution and the current embedding distribution of malicious clients is generally lower than that of benign clients. This result is consistent with the experimental results on image datasets.

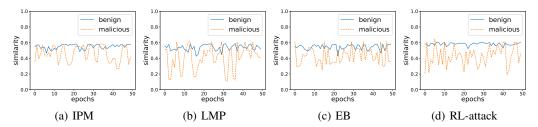


Figure 13: The statistical results of the similarity between the current client data distribution and its historical data distributions under four types of attacks that vary with the training epochs.