FL-PLAS: Federated Learning with Partial Layer Aggregation for Backdoor Defense Against High-Ratio Malicious Clients

Jianyi Zhang, Ziyin Zhou, Yilong Li, Qichao Jin
Beijing Electronic Science and Technology Institute
Beijing, China
zjy@besti.edu.cn

Xiali Hei
University of Louisiana at Lafayette
Lafayette, Louisiana, USA
email@email

Abstract

Federated learning (FL) is gaining increasing attention as an emerging collaborative machine learning approach, particularly in the context of large-scale computing and data systems. However, the fundamental algorithm of FL, Federated Averaging (FedAvg), is susceptible to backdoor attacks. Although researchers have proposed numerous defense algorithms, two significant challenges remain. The attack is becoming more stealthy and harder to detect, and current defense methods are unable to handle 50% or more malicious users or assume an auxiliary server dataset.

To address these challenges, we propose a novel defense algorithm, FL-PLAS, **F**ederated **L**earning based on **P**artial **L**ayer **A**ggregation **S**trategy. In particular, we divide the local model into a feature extractor and a classifier. In each iteration, the clients only upload the parameters of a feature extractor after local training. The server then aggregates these local parameters and returns the results to the clients. Each client retains its own classifier layer, ensuring that the backdoor labels do not impact other clients. We assess the effectiveness of FL-PLAS against state-of-the-art (SOTA) backdoor attacks on three image datasets and compare our approach to six defense strategies. The results of the experiment demonstrate that our methods can effectively protect local models from backdoor attacks. Without requiring any auxiliary dataset for the server, our method achieves a high main-task accuracy with a lower backdoor accuracy even under the condition of 90% malicious users with the attacks of trigger, semantic and edge-case.

***K*eywords** Federated learning · Backdoor resistant · Data poisoning · Partial layer aggregation

1 Introduction

Federated learning (FL) is a machine learning technique that has garnered significant attention due to its ability to protect data privacy in large-scale distributed systems. FL enables collaborative model training without the need to share sensitive data between multiple parties or a central server, making it particularly suitable for cloud and edge computing environments. In these systems, where data is distributed across multiple entities, ensuring privacy and security is critical. The technique allows data to remain secure on each participant's device or server, avoiding the risks associated with centralized data storage [1, 2]. Users train the model locally on their own data, upload updates to a central server, which then aggregates the models into a global model. This iterative process continues until the global model reaches a satisfactory level of accuracy.

Privacy concerns have spurred the adoption of federated learning for collaborative training of shared deep learning models [2]. However, this approach is vulnerable to backdoor attacks [3, 4]. These manipulated models function normally on clean data but maliciously misclassify backdoor samples [5]. In this paper, we focus on trigger [6], semantic (like label-flipping) [7, 8, 9], and edge-case attack [10] as they are challenging problems in FL. Other attacks like directly modifies local weights and optimize the trigger pattern [11], or poisoning backdoor-critical layers [12] also achieved notable effects under specific conditions [13].

Existing defenses strategy are divided into two main types. One type aims to restrict model updates within controlled bounds through regularization or processing anomalous data. RSA [14] and NDC [15] are examples of such defenses, which can weaken the impact of malicious updates on the model. However, this approach involves trade-offs, necessitating model normalization and noise introduction, which can affect accuracy, attack resilience, and main-task accuracy after being maliciously attacked. The other type relies on the classification of malicious models [16] to detect and exclude the malicious local update, such as FLTrust [17], FLAME [18], and Krum [19].

However, the effectiveness of these methods is based on the assumption that the majority of users are honest, which means that they may not work well when there is a large percentage (e.g., over 50%) of malicious users. Additionally, collecting some user data as an auxiliary dataset for the server, such as in the case of FLTrust, conflicts to some extent with the privacy protection of federated learning.

In this paper, we propose FL-PLAS (**F**ederated **L**earning based on **P**artial **L**ayer **A**ggregation **S**trategy), a backdoor defense algorithm. Our approach involves preserving the local classifier in the client, preventing contamination of benign users' local models by global backdoor neurons. This is achieved by dividing the local model into two parts: the feature extractor and the classifier. During each iteration, clients upload only the feature extractor's parameters after local training. The server then aggregates these parameters and shares the results with clients. Each client keeps its own classifier layer to prevent malicious users' backdoor data from affecting benign users' models.

We experimentally evaluate the effectiveness of our FL-PLAS framework against trigger attacks [6], semantic attacks [7], and edge-case attacks [10]. Our results demonstrate that FL-PLAS can successfully defend against all of these attacks without compromising the privacy of user data.

Compared to the five defense methods (RSA, NDC, FLTrust, FLAME, and Krum), we find that most existing methods fail to effectively defend against backdoor attacks when the proportion of malicious users exceeds 50%, except FLTrust and FL-PLAS. However, FLTrust requires the collection of user data, which conflicts with the privacy-preserving nature of federated learning.Specifically, our contributions are:

- We conducted an in-depth analysis of the role different neural network layers play in defending against backdoor attacks. Using both simple and complex datasets, classification models, and extensive experiments with seven defense models and three attack models, we laid the foundation for understanding how specific layers contribute to backdoor defense.
- We implemented a backdoor defense scheme using the partial layer aggregation strategy, specifically targeting scenarios with a high proportion of malicious clients. Our approach effectively addresses the limitations of current methods, particularly in environments with a significant number of malicious clients and scenarios where the server cannot retain user data.
- Through rigorous experiments, we demonstrated that our method maintains superior performance even when the proportion of malicious clients exceeds 90%. This highlights the robustness and efficacy of our approach in highly adversarial federated learning environments.

2 Background

2.1 Federated Learning (FL)

Federated learning is a type of distributed learning. As Figure 1 shows, FL consists of N users and one server. The user is responsible for training the model and passing the trained model to the server. And the server can aggregate the user's model to generate a global model.

We roughly divide FL into three steps in one iteration (illustrated in Figure 1):

- **Step 1**: The server sends the global model to the client, which gets the global model and starts training.
- **Step 2**: The client uploads the trained local model to the model aggregator, which is then aggregated by the server.
- **Step 3**: After the server-side aggregation is completed, the server sends the aggregated global model to the client.

2.2 Backdoor Attacks in Federated Learning

Generally, obtaining high accuracy on the testing dataset is the aim of the model design. Backdoor attacks do not destroy the accuracy of the global model, but induce it to make attacker-chosen mistakes on backdoor tasks [5, 20].

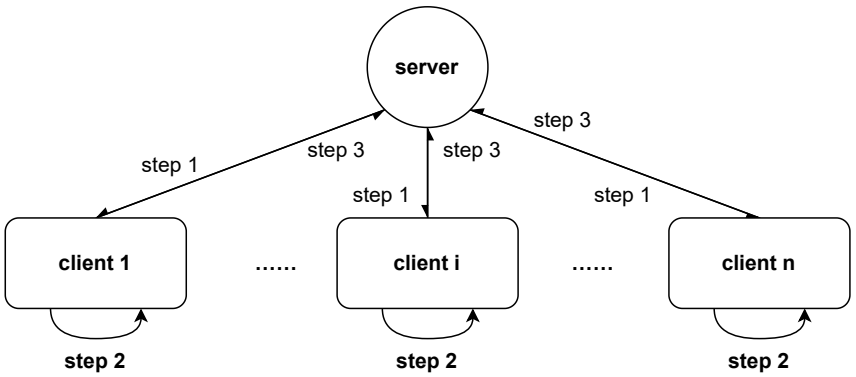


Figure 1: Illustration of the three steps in one iteration of FL. There are N clients and a server. Each client has different classes and sizes of data, representing the heterogeneous distribution of client data.

Backdoor attacks are applied in a variety of contexts, including crowd-sourcing systems [21], recommended systems [22, 23], spam filters [24], etc. In the paradigm of FL, the server has no power to inspect the cleanliness of clients' data. Thus, a malicious client might poison his local data and train a malicious local model update based on it [25]. When the server frequently receives malicious local model updates from such clients, the global model obtained by aggregating the clients' local model updates will be compromised by the backdoor task. As a result, backdoor attacks pose a significant threat to the FL. Since implanting a specific backdoor into the model is much more complex and challenging, the current backdoor attacks commonly use data poisoning.

Data poisoning [26]: The attacker can create backdoor data by adding triggers specified by the attacker (e.g., plus sign, etc.) and modifying the label of the data. This kind of data containing backdoor information is trained together with ordinary benign data. During the model's training process, the backdoor information from the local dataset gradually spreads to the global model, eventually leading to the poisoning of the global model.

Federated learning is more vulnerable to data poisoning attacks due to the larger number of users and the loss of supervision of user data due to the privacy-preserving nature of federated learning. The common data poisoning attacks are the trigger backdoor, the semantic backdoor, and the edge-case backdoor.

Trigger backdoor [6]: The trigger backdoor attack is a prevalent type of backdoor attack that the attacker adds a specific mark or sign to an image and modifies its label. Consequently, the global model will misclassify other images that contain this specific mark. Typically, the triggers used in such backdoor samples are easily visible.

Semantic backdoor [7]: Semantic backdoor attacks are different from trigger backdoor attacks because they do not require any modifications to the input data. Instead, they exploit the semantic information already present in the model by assigning attacker-specified labels to data samples with specific features. For example, an attacker could assign the label "cat" to images of dogs with pointy ears, causing the model to misclassify images of other dogs with pointy ears as cats, or just poison the training data by flipping the labels from "dog" to "cat". This type of attack can cause the model to overlearn the features specified by the attacker, leading to inaccurate predictions for images that contain the same features.

Edge-case backdoor [10]: In the edge-case backdoor attack, the adversary utilizes extremely sparse data in the dataset and modifies its labels, causing the final model to misclassify such data that is hard to find in the training or test set. The attacker mainly exploits the robustness vulnerability in federated learning by modifying the labels of the sparse data in the user dataset. Since the data used in the edge case does not affect most of the data, this backdoor attack is more easily overlooked.

2.3 Backdoor Defense Strategy

There have been several proposed backdoor defense strategies that are effective against common backdoor attacks. The authors of papers [27, 28] discuss why federated learning is vulnerable to backdoor attacks.

Table 1: Comparison of our method with other approaches

	Methods	Defense strategy
Averaging	FedAvg	None
Similarity-based	Krum	Calculates the Euclidean distance Selects one local model as the global model
	FLTrust	Auxiliary dataset as reference Cosine similarity with ReLU to filter out
	FLAME	Calculates the cosine similarity Applied a clustering
Modify updates	RSA	Calculates local update direction Regularize and control the updates
Discard updates	NDC	Sets an upper bound Discards the larger model before aggregation
Layer	Our Method	Partial model aggregation

RSA [14]: For malicious updates of the model, we can regularize and control the updates mainly by using the model's update direction to control each model update within a certain range. Specifically,

$$G_t = \sum \beta_i sign(M_i - G_0) \tag{1}$$

where M_i is the local model of client i, G_{t-1} is the global model in last iteration, β_i is a parameter mainly based on learning rate for $client_i$, and G_t is the global model in this iteration.

NDC [15]: Since backdoor updates may result in larger changes, model updates can be limited in scope by setting a threshold of M. The NDC (Norm Difference Clipping) method employs this strategy to limit the impact of backdoor attacks. The final global model update is calculated as:

$$u_i = \sum \frac{u_{i-1}}{max(1, \frac{||u_{i-1}||}{TS})} \tag{2}$$

where u_i is the model update after i iterations, $||u_i||$ is the l_2-norm of the model update after i iterations, and TS is the threshold set by server.

FLTrust [17]: For FLTrust, the server needs to collect a small clean training dataset. During the training process, the server calculates the cosine similarity between the updates of the clean model and the updates of the user model.

FLAME [18]: FLAME utilizes a clustering approach to identify and remove adversarial model updates. It performs classification analysis and selects the models in the larger class as benign models for aggregation. FLAME calculates the Euclidean distance between the user and global models, takes the median as the benchmark, and calculates the weighted average of the model updates. To attenuate the backdoor, FLAME adds a certain amount of Gaussian noise. Model weights are defined as:

$$e_i = \min(1, \frac{S_{median}}{S_i}) \tag{3}$$

where e_i is the weight of clients, S_{median} is the median Euclidean distance between the user model and the global model, and S_i is Euclidean distance between the user model and the global model of client i.

Krum [19]: Krum assumes that the server knows the number of pairs of malicious users and then calculates the model similarity among the models uploaded by users. Krum first finds the geometric center of the user models and then selects the models most similar to other user models as the final global aggregation models. Specifically,

$$G = \arg\min ||M_i - M_j|| \tag{4}$$

where G is the global model, M_i is the model of client i, $||M_i - M_j||$ is the Euclidean distance between the model i and the model j. The key differences between our method and other defense strategies can be seen in Table 1.

3 Related Work

DNNs are vulnerable to both data and model attacks, including backdoor [6, 7], evasion [29, 30], fault injection [31], etc. In the backdoor attack, hidden triggers cause DNNs to make false predictions with attacker-specific data while behaving

normally with benign data [23, 32, 33, 34, 35]. Popular attack methods involve poisoning data or directly embedding triggers into models. A typical poisoning data attack involves inserting malicious samples or modifying training data to influence the behavior of the model [5, 6, 21, 22, 23, 24, 32, 33, 34, 35, 36, 37, 38, 39, 40]. Federated learning, due to its structure, is inherently more susceptible to data poisoning attacks [7, 10, 14, 27, 41, 42, 43, 44]. From the perspective of the purpose of the attack, these attacks can be divided into untargeted attacks [14, 27, 41, 42, 43] and targeted attacks [7, 10, 44, 45]. Untargeted attacks aim to deteriorate the global model. And targeted attacks aim to induce the global model to make some attacker-chosen mistakes in certain inputs, without deteriorating the global model. In terms of attack types, these attacks in federated learning are categorized as trigger, semantic, and edge-case attacks.

In order to cope with challenges of backdoor attacks, defense methods are essential. In backdoor defense, the goal of the defender is to minimize the impact caused by the backdoor. Defenses can be categorized by their mechanisms: trigger-backdoor mismatch [46, 47], backdoor elimination [48, 49, 50, 51, 52, 53], and trigger elimination [54, 55, 56]. In federated learning, backdoor defenses divide into two types: mitigating the impact of malicious models on the global model (limitation) [14, 15], and detecting malicious models [19]. Limitation-based defenses constrain user-uploaded model updates' norms, minimizing malicious models' global impact. For the detection of malicious models [19], servers identify and reject malicious models. Recent research has demonstrated inverting local model updates to exclude malicious updates from aggregation [57], and a reverse engineering-based trigger defense can provide a sufficient condition on the quality of trigger recovery [58].

However, it cannot be neglected that the effectiveness of these methods relies heavily on the assumption that the majority of users behave honestly. Assuming a condition that the number of dishonest users (malicious users) exceed over 50%. Under such circumstance, these methods did not worked well at all. Moreover, some methods collect a certain amount of local data from users as the root dataset [17] or reference model [59]. Since collecting these data might lead to the leakage of sensitive user information, the requirements are too strict for federated learning and may not be applicable to many scenarios. The way to solve these problems is very important.

Existing solutions, such as classification and clustering-based methods, have significant limitations. Classification-based methods struggle when malicious users are in the majority, and clustering methods become ineffective as they rely on the assumption of fewer malicious clients. Moreover, solutions that depend on the server retaining sample data are impractical due to strict data privacy requirements and the inherently distributed nature of data.

Regarding pFL-related research [60, 61, 62, 63, 64, 65], while Gao [66] and Qin et al. [67] demonstrated that partial model aggregation could effectively defend against backdoor attacks, their study merely provided a simple evaluation of various pFL algorithms initially designed to address data heterogeneity. They did not conduct detailed research on the effectiveness of pFL against different proportions of malicious clients, different model parameters, nor did they deeply explain the reasons for its effectiveness.

4 Problem Setup

Threat model: Similar to [41, 17, 68], we make the following assumptions about the attacker: (i) Attacker controls one or multiple users, replacing original data with backdoor data via user data modification; (ii) Attacker obtains complete info of controlled user - user data, loss function, learning rate; (iii) Malicious users select data for training, manipulate local model updates at will - modify local model training's learning rate, scale model update; (iv) Malicious users can attack any typical deep neural network (DNN); (v) Malicious users do not need to be omniscient and collude with each other.

Defense goals: Similar to [17], we evaluate our method via **fidelity**, **robustness**, and **efficiency**. For fidelity, we target maximal benign update retention during non-attack states. For robustness, our method's efficacy is expected across scenarios with diverse malicious user ratios, backdoor attack types, etc. Efficiency-wise, we refrain from imposing excessive computation and memory demands on user devices beyond FedAvg.

Defender's knowledge and capability: The backdoor defense scheme herein is essentially an aggregation rule. Since this aggregation rule mainly runs on the server side, the defender acquires server-side information, including the global model, user-uploaded model parameters, and user count. However, the server side remains uninformed regarding user data and attacker details, such as malicious user ratio or attack type. Additionally, the server cannot collect user data as in FLTrust. Such data collection could risk exposing sensitive user information. Consequently, the proposed algorithm's operational context is more discreet and pragmatic compared to FLTrust.

Evaluation Metrics: For defense methods, we test the global model using a clean test set and a backdoor test set, respectively. To obtain the main-task accuracy (MA) and backdoor accuracy (BA), the backdoor test set is composed by adding a backdoor to the clean test set. We assume that there are BU benign users among all users, M_i are the

samples of the $i-th$ client whose labels are correctly predicted on the clean test set, and $|M_i|$ is the size of M_i. B_i are the samples of the $i-th$ client whose labels are classified as the attacker-chosen class (backdoor attack successfully attacked), and $|B_i|$ is the size of B_i. $|M|$ and $|B|$ are the size of clean test set and backdoor test set.

Here we define MA as the proportion of data the model predicts correctly on a clean test set. That is, $MA = \frac{1}{BU}\sum\frac{|M_i|}{|M|}$. So, the higher the MA is, the more correctly the model predicts. BA is the proportion of data in the backdoor task that the global model classifies as the attacker-chosen class. That is $BA = \frac{1}{BU}\sum\frac{|B_i|}{|B|}$. Therefore, the higher the BA is, the more successful the attack is, which indicates a weaker capability of the defense solution.. For FL-PLAS, we test all local models and average their MA and BA to be the MA and BA of FL-PLAS.

5 FL-PLAS Overview and Design

5.1 High-level Idea

Motivation: In federated learning, preventing backdoor data from poisoning benign client models is crucial. Common backdoor defense algorithms typically rely on the classification or comparison of user models with benign server models. However, when the number of malicious users exceeds 50%, classification-based methods become challenging to handle. In such settings, traditional FL defense mechanisms often fail. Some methods rely on clustering to identify malicious clients, which becomes ineffective when the majority of clients are malicious. Others require the server to maintain a portion of the sample data, which is impractical in many computing and data systems where data privacy and distribution uniformity cannot be guaranteed.

There is a clear gap in developing robust FL defense mechanisms that can effectively operate in environments with a high proportion of malicious clients and where data privacy is paramount. Current methods do not adequately address the challenges posed by highly adversarial conditions and stringent privacy constraints. Our previous research observes that some certain layers in neural networks exhibit distinguishable patterns between malicious and benign updates [69]. Hence, we hypothesize that processing certain layers separately may break the connection between backdoor data and its corresponding labels. We try to isolate the influence of backdoor clients during the training of federated learning models, ensuring that their impact remains confined to their own clients and does not affect benign clients.

Key observation and idea: In order to test our hypothesis, we conducted a simple experiment comparing the main-task accuracy and backdoor accuracy of four types of neural network models. We first trained a clean model and a backdoor model on the MNIST dataset [70] using the `Lenet` [71] architecture. Then we separated these two models into their corresponding feature extractors (FE) and classifiers, and assembled them into four new models: clean FE with clean classifier, clean FE with backdoor classifier, backdoor FE with clean classifier, and backdoor FE with backdoor classifier.

As shown in Table 2, the backdoor accuracy is dependent on whether the classifier is poisoned or not, regardless of whether the feature extractor is poisoned or not. The same observations are also demonstrated on the CIFAR-100 dataset [72] with `ResNet-18` [73]. Hence, the key idea of our algorithm is that the server only aggregates part of the model uploaded by the user, and the user keeps their classifiers locally.

Table 2: The main-task accuracy (MA) and backdoor accuracy (BA) of neural networks with backdoor in different Feature Extractors (FE) and Classifiers.

MA/BA	Clean FE	Backdoor FE
Clean Classifier	0.99/0.1	0.99/0.1
Backdoor Classifier	0.98/1	0.98/1

5.2 FL-PLAS Design

According to our interesting findings, we divide our model into two parts: the feature extractor and the classifier. In each iteration, the clients only upload the parameters of the feature extractor after local training. Then, the server aggregates these local parameters and returns the results to the clients. With the partial layer aggregation strategy, every client keeps its own classifier layer to isolate the malicious users' backdoor data from benign users.

As the data distribution is unknown to us, we use empirical risk as our risk function. The risk function of client i is :

$$R_{exp}(f) = \frac{1}{S}\sum_{i=1}^{S}(L(y_i, f(x_i))) \tag{5}$$

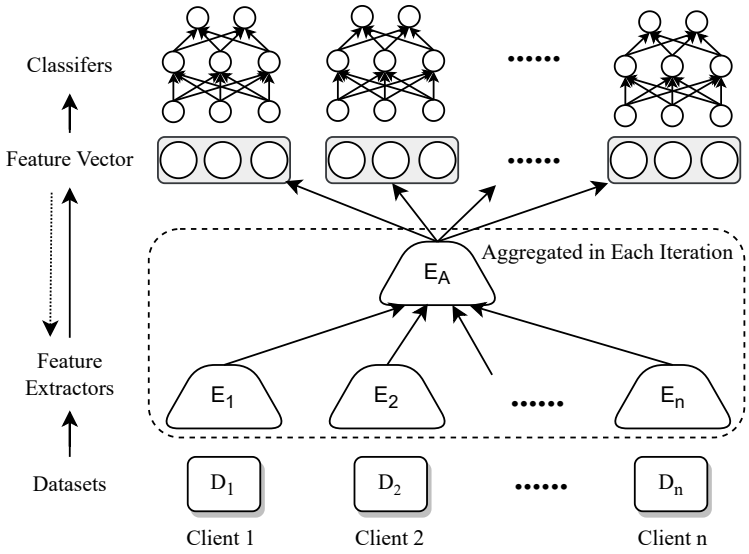


Figure 2: Illustration of FL-PLAS workflow in round t.

where S is the sample content of client i.

For example, if we choose classifier as our cutting layer, we can divide the model into feature extractor (FE) and classifier (CL). This is how model works.

$$\hat{y_i} = FE(x_i) \tag{6}$$

$$y_i = CL(\hat{y_i}) \tag{7}$$

where x_i is the data of client i, $\hat{y_i}$ is the feature extraction result (or embedding), and y_i is the prediction result.

We upload the feature extractor part of the model, which is aggregated by the aggregator, while keeping the classifier locally on the user side. We denote the number of clients as r.

$$G = FedAvg(M_{1,FE}, \cdots, M_{r,FE}) \tag{8}$$

$$M_i = G + M_{i,CL} \tag{9}$$

Figure 2 and Algorithms 1 and 2 illustrate the working mechanism of FL-PLAS in round t.

Step 1: User i uses their local dataset D_i to train a local model M_i and then uploads their model to the server.

Step 2: After receiving the models uploaded by the users, the server aggregates specific layers of the models (e.g., feature extractors) to obtain the global model G. The server then returns the aggregated model to the users.

Step 3: Users replace their partial model with the global model G, then use their local dataset D_i and the new local model to calculate the feature vectors and losses. They then use these to optimize the entire model, including both the feature extractor and classifier.

Algorithm 1 FL-PLAS (aggregation rule)

Input: The number of received clients r, number of cutting layers l, total iterations T, initial model G^0, size of local examples n
Output: The global model G.

for t = *1, 2, ···,* T **do**
  for i = *1, 2, ···,* r **do**
    $M_i^t \leftarrow ClientUpdate(G^{t-1}, l)$
  end
  for *layer p in* M_i **do**
    $if\ p < l\ then\ G_p^t \leftarrow \sum \frac{n_i}{n} M_{i,p}^t$
  end
end
return G

Algorithm 2 ClientUpdate

Input: The local dataset of client i D_i, global model from server G, number of cutting layers l.
Output: The local model ${M_i}^t$.

for *layer p in* M_i **do**
  $if\ p < l\ then\ M_p^t \leftarrow G_p$
end
for *each batch* $b \in D_i$ **do**
  $M_i \leftarrow M_i - \eta \Delta l(b, M_i)$, η is the local learning rate, Δl denotes the loss using data b and model M_i
end
return ${M_i}^t$

The algorithm complexity of FL-PLAS covers two aspects, global model update and client update (Algorithm 2). According to Algorithm 2, the main complexity of client update is $O(p + \frac{D_i}{b})$, which relies on two iterations with p times and $\frac{D_i}{b}$ times. Therefore the complexity of Algorithm 2 equals to Linear complexity $O(N)$. According to Algorithm 1, the main complexity of global model update is $O(p * n)$, which covers n models to aggregate with p times iteration. Therefore, the complexity of FL-PLAS is $O(r * O(N) + p * n)$ and equals to $O(N^2)$.

Table 3: The default FL system parameter settings.

Dataset and Partition	MNIST	CIFAR-10			CIFAR-100
Total number of clients	100				
Client per round	30				
Backdoor type	Trigger	Trigger	Semantic	Edge-case	Trigger
Learning rate	6.7×10^{-3}	2.7×10^{-3}			1.5×10^{-5}
Local iterations	1				
Global training round	200				
Batch size	32				
Combined learning rate	learning rate $\times 0.998^t$				
Optimizer	SGD				
Momentum	0.9				
Weight decay	10^{-4}				

6 Evaluation

6.1 Experimental Setup

1) *Datasets*: We use three image datasets to evaluate the effectiveness of FL-PLAS. The datasets are also divided according to the number of users (M). For the non-independent and identically distributed (non-i.i.d.) data, we divide the data according to the Dirichlet distribution, where the users get different data in terms of distribution and data volume. In addition, the parameter used for the Dirichlet distribution in this experiment is 0.2, *i.e.*, the data distribution $\chi \sim Dir(0.2, M)$.

MNIST: The MNIST dataset is widely used in computer vision tasks and consists of handwritten digits. It contains a training set of 60,000 samples and a testing set of 10,000 samples, which are normalized to 28×28 pixels and centered at a fixed size.

CIFAR-10: The CIFAR-10 dataset is a color image classification dataset that consists of 50,000 training images and 10,000 test images. Similar to MNIST, the CIFAR-10 dataset classifies the images into 10 categories, which include airplanes, cell phones, and birds.

CIFAR-100: CIFAR-100 is very similar to CIFAR-10, but it contains 100 classes instead of 10. Each class in CIFAR-100 contains 500 training images and 100 test images.

For comparability, we used a consistent experimental setup. The MNIST dataset employed the trigger attack for the backdoor, employing a `Lenet` model. CIFAR-10 evaluated FL-PLAS against various attacks (trigger, semantic, edge-case) using a `MobileNet` model [74]. CIFAR-100 focused on the trigger attack, employing a `ResNet-18` model.

The MNIST dataset tested FL-PLAS's backdoor defense on simple images and validated its performance on the `Lenet` model. CIFAR-10, coupled with `MobileNet`, assessed defense effectiveness on color images and complex models,

demonstrating robustness against new attacks. CIFAR-100 showcased backdoor defense across datasets with up to 100 classes.

2) *Attack settings*: In our experiment, we evaluate the effectiveness of FL-PLAS against three types of backdoor attacks: the trigger attack, the semantic attack, and the edge-case attack. In each attack, the attacker applies a backdoor processing technique to a certain percentage p of user data to inject backdoor patterns for subsequent attack processing.

Trigger attack: The trigger attack modifies user data by adding the same trigger logo to each sample (a 2×2 white box at the top right for MNIST and a 5-pixel white plus sign at the top right for CIFAR-10/CIFAR-100) and changing the label to a specified attacker label (in our experiment, the attacker label is set to 0).

Semantic attack: As in [7], in the training and evaluation datasets, we select the “green car" as the backdoor images. We modify the label of the “green car" to be “bird" for this attack.

Edge-case attack: As in [10], we use aircraft images oriented in the southwest direction to generate the backdoor data and the backdoor test set. The southwest-oriented aircraft represent an extremely small percentage of the global data. For the generated backdoor data, we modify its data label to category 9 for trucks. Additionally, if the backdoor is not triggered, the above backdoor data will be classified as category 1 for cars.

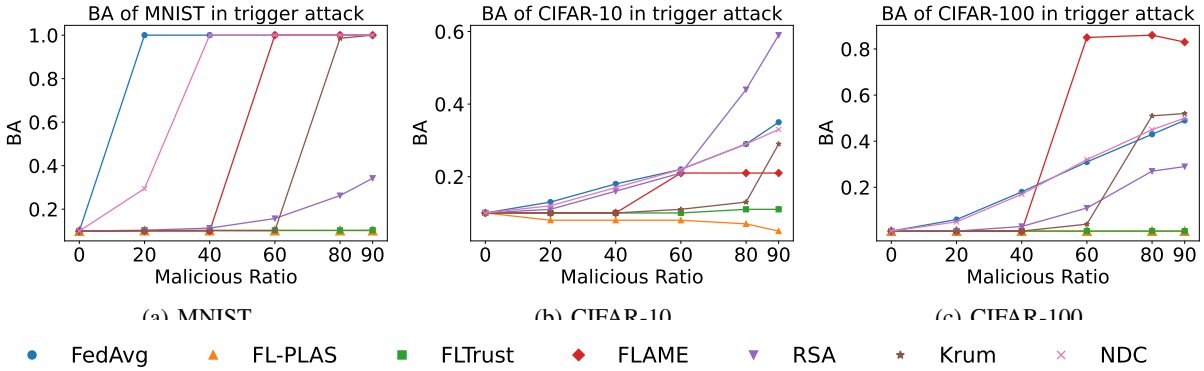


Figure 3: BA of various datasets under the trigger attack. 3(a) depicts how the BA of MNIST changes against different ratios of malicious clients; 3(b) depicts how the BA of CIFAR-10 changes; and 3(c) depicts how the BA of CIFAR-100 changes.

3) *System Settings*: In this experiment, we use the same experimental settings as in [6]. We set the total number of users to 100 and the number of users selected each time to 30%, where a constant number of malicious users appear in each training according to the proportion of malicious users. Moreover, the malicious users will backdoor 30% of their data to generate backdoor data. The detailed model parameters are shown in Table 3.

4) *Defenders' Settings*: For FLtrust, we set the size of the clean small training dataset (called “root dataset") to be 100 and the local iteration to be 1 as [17]. For FLAME, we set σ (the noise level bound in FLAME) to 0.01 according to [18].

6.2 Experimental Results

6.2.1 Robustness

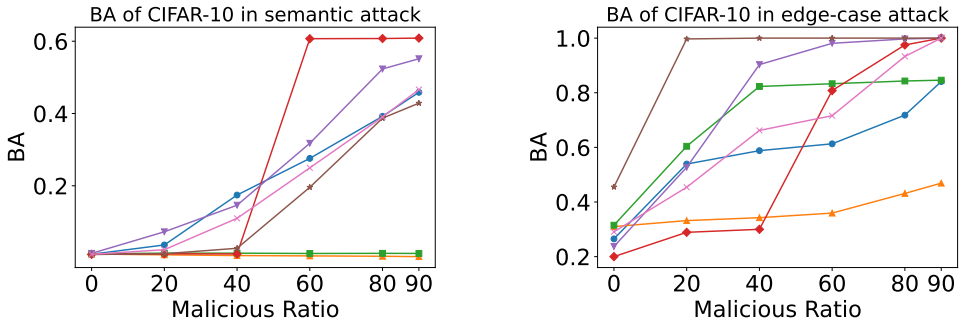
Figure 3(a) illustrates the defense effects against backdoor attacks on the non-i.i.d. MNIST dataset. Existing defense strategies show satisfactory results when the proportion of malicious users is under 40%. On the contrary, FL-PLAS, FLTrust, and Krum outperform others when malicious users exceed 50%. FLAME is a cluster-based method that performs the best when the number of malicious users is less than 40%, and its performance gradually declines when the malicious users exceeds 40%. The performance of RSA is decreasing as the number of malicious users is increasing. Krum is effective with 60% malicious users due to pre-trained models aiding convergence and benign model selection. However, Krum becomes ineffective as malicious users reach 80%.

Figure 3(b) illustrates the pronounced superiority of our approach in the CIFAR-10 dataset. All six strategies exhibit robust backdoor defense below 50% malicious users proportion. However, as malicious users increasing, only FL-PLAS and FLTrust maintain better defense.

Figure 3(c) illustrates a more substantial disparity among defense methods in the non-i.i.d. CIFAR-100 dataset. Under 40% malicious users, FL-PLAS, FLTrust, FLAME, and Krum exhibit stronger defense. Only FL-PLAS and FLTrust sustain effective defense against backdoor attacks when the malicious users exceed 40%

6.2.2 Two New Types of Attacks

Besides the basic trigger attacks, there are two new backdoor attacks emerging recently: semantic attacks and edge-case attacks. Taking the CIFAR-10 dataset as an example, the effectiveness of the above mentioned backdoor defense strategies for the new types of attacks is shown in Figure 4.



(a) semantic (b) edge-case

Figure 4: BA of CIFAR-10 in two new types of attacks. 4(a) shows semantic attack, 4(b) shows edge-case attack.

As it shown in Figure 4(a), for the semantic attack on the CIFAR-10 dataset, FL-PLAS and FLTrust can still ensure the backdoor content remaining within a reasonable range even being attacked by 80% of malicious users. FLAME and Krum show a strong ability to identify malicious models when the proportion of malicious users is less than 40%. When the malicious users is higher than 40%, the defense effect becomes weaker. Since the strategies of RSA and NDC fail to identify malicious models and benign models under semantic attacks, the global models of these methods gradually lose their resistance against backdoor attacks, which causes the value of BA in Figure 4(a) increasing gradually, and indicates an increase in the backdoor content in the model.

For edge-case attacks, as shown in Figure 4(b), most of the backdoor defense methods are difficult to defend against edge-case attack when the proportion of malicious users is high. Only FL-PLAS performs good defense capability against edge-case backdoor. FLAME prevents the improvement of model backdoor task accuracy by aggregating benign users and adding noise when the proportion of malicious users is less than 40%. When the proportion of malicious users is large, malicious users will also participate in aggregation, resulting in an increase in backdoor accuracy.

Table 4: The BA when there are 90% malicious users.

(a) Trigger attack

	FedAvg	FLTrust	FLAME	RSA	Krum	NDC	FL-PLAS
MNIST	1	0.10	1	0.34	1	1	**0.10**
CIFAR-10	0.35	0.11	0.21	0.59	0.29	0.33	**0.05**
CIFAR-100	0.49	0.01	0.83	0.29	0.52	0.50	**0.01**

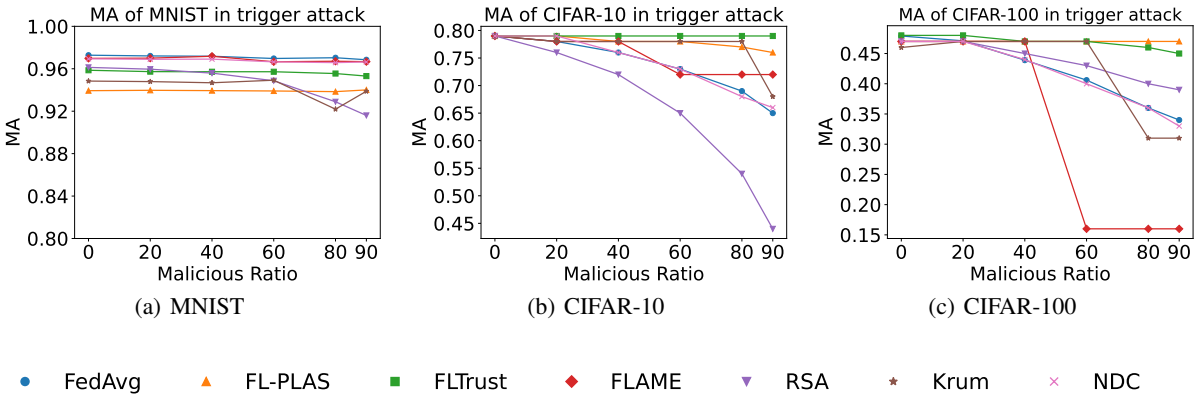
(b) New types of attack

	FedAvg	FLTrust	FLAME	RSA	Krum	NDC	FL-PLAS
Semantic	0.46	0.01	0.61	0.55	0.43	0.47	**0.004**
Edge-case	0.84	0.85	1	1	1	1	**0.47**

When the proportion of malicious users is up to 90%, as shown in Table 4, our method FL-PLAS can still be effective in reducing the backdoor in the model.Moreover, our method improves defense capability by over 44% compared to other methods, reaching up to 99.34% (FLAME).

6.2.3 Main-task Accuracy

An important evaluation factor for a defensive strategy is how well it performs on the main task while simultaneously restricting the backdoor.



(a) MNIST (b) CIFAR-10 (c) CIFAR-100

Figure 5: MA of datasets in trigger attacks. 5(a) shows how the MA of MNIST changes, 5(b) shows how MA of CIFAR-10 changes, and 5(c) shows how MA of CIFAR-100 changes.

When there is no attack, as shown in Figure 5(a) for the MNIST dataset, FLAME and NDC demonstrate good accuracy for the main task. FLAME utilizes clustering-based aggregation, which reduces noise via users update convergence as the proportion of malicious users increasing. NDC truncates update in surpassing a norm threshold, which mitigating impact from dissimilar user updates. This maintains the level of main-task accuracy (MA) closing to FedAvg. The limitation of Krum comes from its reliance on individual user models regards as the global model, and curtails data utilization. Our reduced main-task accuracy (MA) stems from the simplicity of the `Lenet` model. With two classifiers in the four-layer model, our method's division leads to a two-layer overall model.

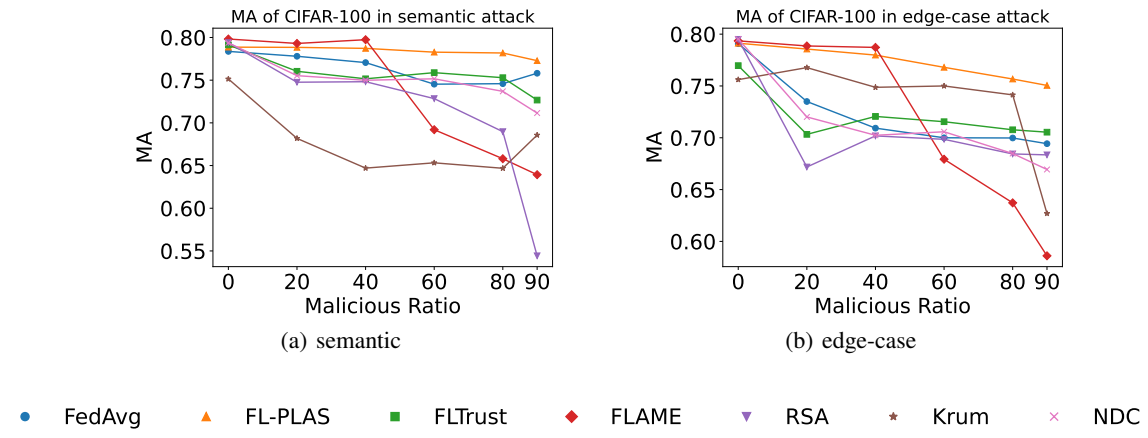
As malicious user rate is increasing, FLTrust, NDC, FLAME, and FedAvg display slight decrease in MA but RSA sharply drops on the contrary. RSA emphasizes on updating direction to render it susceptible to malicious influences, which leads to pronounced MA reduction. The accuracy of Krum fluctuation stems from its user model selection, causing notable discrepancy in the final model for high malicious user proportions. Our method displays reduced sensitivity to malicious rate compared to RSA or Krum. Nonetheless, its effectiveness is limited on the MNIST dataset using the `Lenet` global model. This is elaborated in "Limitation" Section.

For CIFAR-10 with `MobileNet` in Figure 5(b), only RSA shows a lower level of effectiveness and the highest degree of decline among the backdoor defense strategies. It is because the strategy of RSA sends the direction of model updates as a parameter to the global model. If the global model is only aggregated by the update direction of the client model, it will greatly impact the MA. With the increasing proportion of malicious users, only FLTrust and FL-PLAS show a better MA, while others all have a certain degree of decline.

For the CIFAR-100 dataset with `ResNet-18` depicted in Figure 5(c), when there is no attack, various defense strategies exhibit robust model aggregation without attacks except FLAME and Krum. With increasing malicious users, FL-PLAS, FLTrust, and RSA sustain MA within an acceptable range. FedAvg and NDC experience gradual MA decline, while FLAME and Krum face sharper drops at 40% and 60% respectively. This is attributed to FLAME and Krum selection or utilization of certain users' models for aggregation or as the global model. When the proportion of malicious users reaches a certain threshold, these methods favor malicious models, causing abrupt MA deterioration.

In the semantic attack (Figure 6(a)), Krum's initial MA is lower due to it focuses on a subset model. With increasing malicious users, the drawbacks of FLAME and RSA gradually become more evident and the MA decreases rapidly. Meanwhile, the MA of FedAvg and FLTrust also decline in a certain degree, whereas our method FL-PLAS always performs the best MA.

In the edge-case attack shown in Figure 6(b), only Krum has a lower MA when there is no attack. When the percentage of malicious users is less than 40%, FLAME can still show a good MA, which means FLAME can detect malicious models. On the other hand, when the percentage of malicious users gradually increases, especially when the percentage of malicious users reaches 90%, the MA of all these defense strategies is droping. In contrast, our method FL-PLAS



(a) semantic (b) edge-case

Figure 6: MA of CIFAR-10 in two new types of attacks. 6(a) shows how the MA of CIFAR-10 changes in semantic attack, and 6(b) shows in edge-case attack.

defense method performs the best. Even when the proportion of malicious users is as high as 90%, FL-PLAS still achieves a considerable reduction in MA, the results is shown in Table 5.

Table 5: The MA when there are 90% malicious users.

(a) Trigger attack

	noattack	FedAvg	FLTrust	FLAME	RSA	Krum	NDC	FL-PLAS
MNIST	0.97	**0.97**	0.95	**0.97**	0.92	0.94	**0.97**	0.94
CIFAR-10	0.79	0.65	**0.79**	0.72	0.44	0.68	0.66	0.76
CIFAR-100	0.48	0.34	0.45	0.16	0.39	0.31	0.33	**0.47**

(b) New types of attack

	noattack	FedAvg	FLTrust	FLAME	RSA	Krum	NDC	FL-PLAS
Semantic	0.78	0.76	0.73	0.64	0.54	0.69	0.71	**0.77**
Edge-case	0.79	0.69	0.71	0.59	0.68	0.63	0.67	**0.75**

6.2.4 Efficiency

In contrast to the fundamental FedAvg, FL-PLAS doesn't necessitate additional computational resources or runtime space for users or servers. It entails partial user model aggregation and replacement of the server's global model by users for the aggregation's completed part. This process is resource-efficient, requiring no extra consumption. Conversely, most existing algorithms demand more computational power than FedAvg. Moreover, our method reduces server layers, minimizing data transfer volume and the risk of model compromise during transfers.

Table 6: The screening complexity for seven aggregation methods. ζ denotes the number of local model update parameters, M means the number of label classes and τ is the number of collected local model updates in each iteration.

Method	Screen Complexity
FedAvg	***O(0)***
FL-PLAS	***O(0)***
Multi-Krum	$O(\tau^2\zeta)$
Krum	$O(\tau^2\zeta)$
RFA	$O(\tau\zeta R^*)$
RSA	$O(\tau\zeta)$
NDC	$O(\tau\zeta)$

In order to evaluate efficiency, we use screening complexity. It refers to how the server handles the received updates from local models in FL. Screening complexity involves screening, processing, and aggregating local model updates to ensure certain performance and efficiency. Thus we compare screening complexity of the proposed aggregation methods. In Table 6, the server using FL-PLAS doesn't need to screen the received local model updates, which is similar to FedAvg. Therefore, both of their screening complexities are $O(0)$. Krum and Multi-Krum compute the mutual distances between τ client local model updates. NDC and RSA prune and regularize τ local model updates respectively. RFA finds the geometric center by considering τ client local model updates until a defined condition is met. In summary, all the aforementioned methods classify malicious clients, hence they consider ζ parameter of local model updates. Overall, for evaluating screening complexity, as well as FedAvg, our FL-PLAS method performs better than others.

In Section 6, conclusively, under a high malicious ratio, our method FL-PLAS has the best backdoor defense capability under traditional trigger attacks, semantic attacks, and edge-case attacks without requiring any user data on the server side. Similarly, compared with other defense methods, FL-PLAS also performing well in MNIST, CIFAR-10, and CIFAR-100, better than FedAvg, FLTrust, FLAME and even LFighter (see Appendix A), .etc. Also, we conducted experiments in Resnet-50 and Resnet-101 and the results were similar (see our code). In a word, it is good performance that FL-PLAS in using different network and defencing different adversarial methods.

7 Discussion

7.1 How Many Layers to Aggregate

In order to determine how many layers to aggregate in our partial layer aggregation method, we change the number of layers selected by our approach and analyze the results. We take the CIFAR-10 dataset using `MobileNet`, and the percentage of malicious users is 90% as an example. We use $BA_{atk} = \frac{1}{MU}\sum\frac{|B_i'|}{|B|}$ to evaluate the depth of poisoning where B_i' refers to the samples of the i^{th} client whose labels are correctly predicted on the backdoor test set with the local model of malicious users and $||$ refers the size.

As we can see in Table 7, increasing the aggregation layer leads to higher main-task accuracy but also higher backdoor accuracy, indicating poorer defense. Therefore, the selection of aggregation layers needs to be analyzed for the corresponding network structures, rather than simply aggregating all feature extraction layers.

Table 7: Effect of the number of aggregation layers.

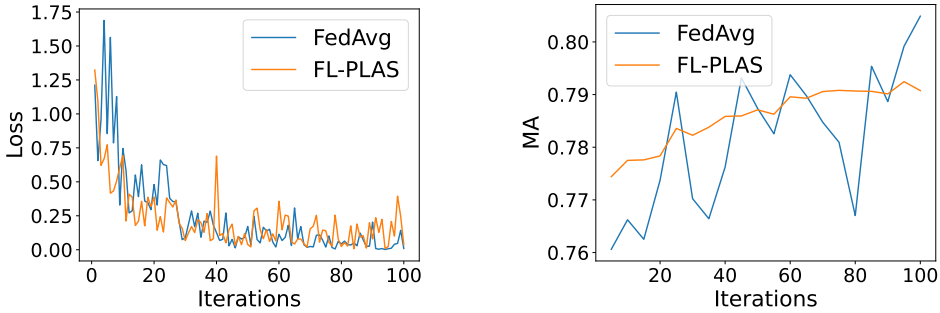
Layers	MA	BA	BA_{atk}	BA_{atk}-BA
10	0.770	0.110	0.455	0.345
11	0.781	**0.109**	0.463	0.354
12	**0.785**	0.369	**0.769**	**0.400**
13 (classifier)	0.7897	0.900	0.900	-

7.2 Convergence

In common network architectures like VGG9 (classifier scale:0.06%), EfficientNet (0.09%), GoogleNet (0.17%), DenseNet (0.38%),and SeNet (0.05%), the parameters in the classifier impact on overall model is minimal. As the number of classifier layers is a small part of the model, not aggregating classifiers has a limited effect on convergence. Figure 7(a) shows that the loss of FL-PLAS is 0.4 at the beginning, while the loss of FedAvg is 1.6. Their losses are both down to 0.1 after about 40 iterations.

Furthermore, Figure 7(a) illustrates FL-PLAS and FedAvg converging similarly in 40 iterations. In CIFAR-10 attack (Figure 7(b)), convergence speeds match, but FedAvg fluctuates more, less robust than FL-PLAS.

If we use an over large classifier (i.e., split too many layers for classifier in the DNN), the model's convergence or accuracy would be impacted. Fortunately, we do not need to split too many layers for classifier to defend against backdoor attacks. It is because the backdoor activation neurons mainly aggregate on the last layers of DNN based on our experimental observation. So, we can keep the classifier small, which helps the model defend against backdoor attacks and maintain the model's accuracy.



(a) Loss of MNIST (b) MA of CIFAR-10

Figure 7: Effect of FL-PLAS on the model. 7(a) shows how the loss of MNIST changes in the trigger attack, and 7(b) shows how MA if CIFAR-10 changes in the trigger attack in 100 iterations.

7.3 Limitation

Based on experiments, FL-PLAS excels in robustness and main-task accuracy (MA) in most cases. However, some prior works like FLAME, NDC and FLTrust are better in certain cases while our method performs the best BA for robustness.

As for FLAME, when facing a high proportion of malicious users, the classification scheme may misclassify a moderate proportion of malicious users (more than 40%) as normal users and consist them in model aggregation, even if their model accuracy on the MNIST dataset performs well as FedAvg. On the contrary, NDC relies on restricting model updates, resulting in higher MA but weaker defense effectiveness. As for FLTrust, although its MA is slightly better than our method on CIFAR-10, it performs slightly worse in BA. Moreover, the condition that server requires a certain amount of user data is quite controversial, whereas our method does not require any user data.

As for FL-PLAS, with a smaller model such as `LeNet`, the classifier occupies a higher proportion, and reduces the remaining part after clipping by our method. As the end-user's model learns less from others in the local classifier, using a lightweight neural network leads to decrease MA. It represents a trade-off between privacy and utility, while the BA and MA remain acceptable.

7.4 Backdoor Attack and Byzantine Attack

Our method focuses on the backdoor attack, while some prior works focus on the Byzantine attacks [17, 19, 27, 41, 75]. A Byzantine attack means an unknown number of malicious clients are omniscient, collude with each other, and send arbitrary vectors to the server to disrupt the learning process. A backdoor attack embeds hidden malicious behaviors into deep learning models, which only activate and cause misclassifications on inputs containing a specific trigger.

In federated learning, a Byzantine attack typically refers to an untargeted poisoning attack [17, 20] that aims to destroy the accuracy of the global model. On the other hand, a backdoor attack aims to have the global model mislabel a specific portion of the samples without affecting the overall accuracy of the model. This type of attack is more threatening to the robustness and integrity of federated learning. Backdoor attacks can be carried out by a single attacker or multiple attackers, who may or may not collude with each other. If these attackers collude, the backdoor attack falls under the category of Byzantine attacks. Our method is designed to be insensitive to whether or not the malicious clients collude with each other. This means that our approach can defend against both Byzantine and non-Byzantine backdoor attacks.

7.5 Personalized Federated Learning

For federated learning, the non-IID characteristics of each user's local data will cause federated learning to face the problem of data heterogeneity. Federated learning itself needs to learn the information of participating users through distributed learning, which also causes the server to be unable to obtain the information of each user, which reduces the performance of the final model. A common solution to the problem of data heterogeneity is to personalize user data so that each device can obtain a higher-quality personalized model. Personalized Federated Learning (pFL) methods [63] not only strive to develop a global model but also aim to create a local personalized model for each client. They are more adaptable to the unique characteristics of each client's local dataset. As a result, pFL methods significantly outperform general FL methods in terms of prediction accuracy, particularly in practical scenarios with Non-IID data. pFL is primarily divided into full-model aggregation [64] and partial-model aggregation [62, 65, 67].

Although the concept of partial model sharing is similar with our method, it should be noted that our method is designed to target backdoor attacks. Unlike partial model aggregation, which may share parts of the model, layers, or parameters, our method targets specific layers. Additionally, there are significant differences in the design goals of these two approaches. The aim of pFL is to address the heterogeneity and individualization of data, focusing on localized personalized training. Conversely, our method lies in tackling the issue of a large proportion of malicious clients, preventing backdoor information from being propagated through aggregation.

8 Conclusion

We propose and evaluate a new federated learning backdoor defense scheme called FL-PLAS. Based on some interesting insights, our method leverages the partial layer aggregation strategy to defend against backdoor attacks. We show that our method can handle cases where the percentage of malicious users is greater than 50% without requiring additional user data. We evaluate the performance of our approach on three datasets under three backdoor attacks. The results demonstrate that our method can protect the local models of normal users from backdoor attacks, even when the percentage of malicious clients reaches 90%, without any auxiliary dataset on the server. While our method may appear simple, the experimental results demonstrate the performance of FL-PLAS. It's the first time to propose partial layer aggregation for defending backdoors in federated-learning, and to address the challenge of training in large-presence of malicious clients. These conclusions suggest several important directions for future work, including expanding our research into natural language processing and audio processing, as well as exploring ways to mitigate model poisoning attacks.

Acknowledgment

For the reproducibility of the proposed method, we have published our source code online at https://github.com/BESTICSP/FL-PLAS.

Appendix A

Figure 8(a) and 8(b) illustrate that when there are over 20% and over 40% malicious clients, respectively, LFighter exhibits a sudden increase in the backdoor attack (BA) rate on the MNIST and CIFAR10 datasets. This indicates that LFighter becomes ineffective in defending against backdoor attacks at these higher malicious client ratios. Within the 20% and 40% thresholds, its performance does not significantly differ from other defense methods. Most defense models can withstand attacks from a small number of malicious clients. However, at higher malicious client ratios, FL-PLAS consistently performs the best. On the other hand, Figure 8(c) and 8(d) show that LFighter maintains a high main task accuracy with a low percentage (below 20%) of malicious clients. Yet, when the percentage exceeds 40%, the model's performance rapidly deteriorates, and the main accuracy (MA) is significantly compromised by the backdoor data. Notably, with over 70% malicious clients, LFighter's performance falls below that of the other defense methods.

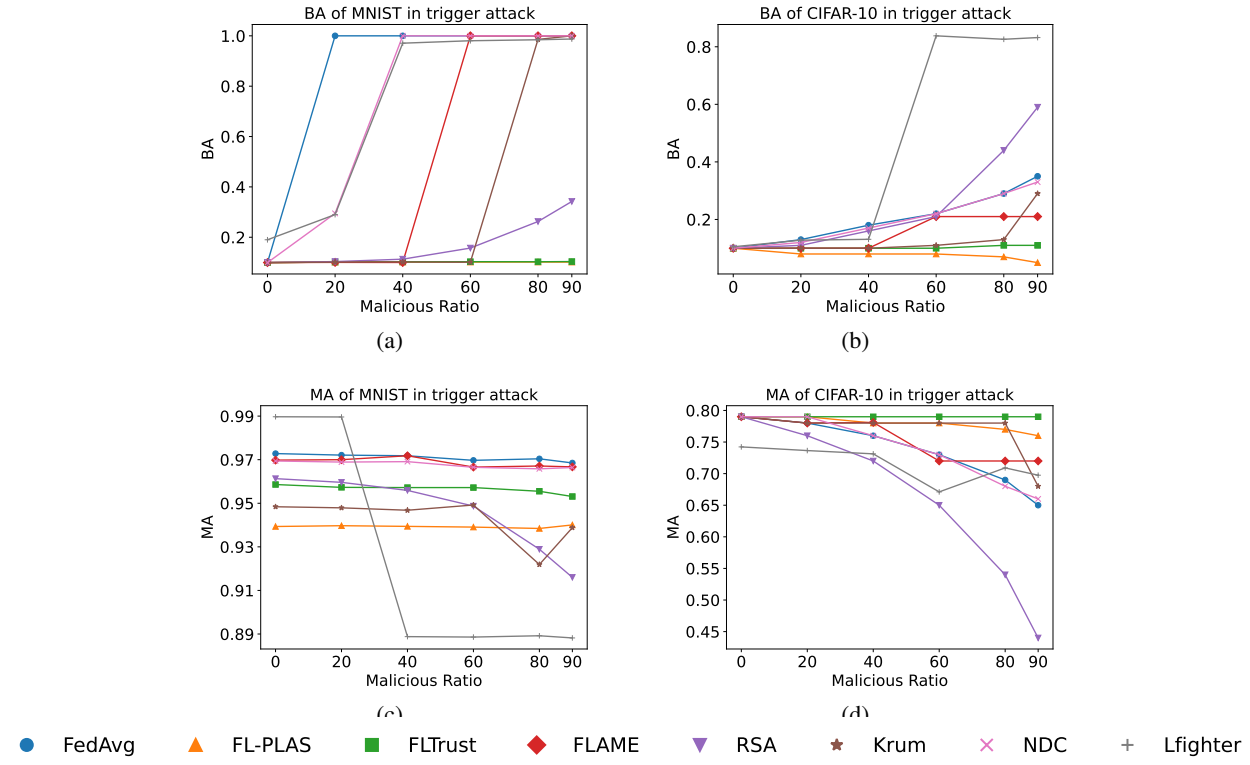


Figure 8: BA and MA of datasets in trigger attacks. 8(a) shows how the BA of malicious ratio changes under MNIST, 8(b) shows how BA of CIFAR-10 changes, 8(c) shows how MA of MNIST changes, and 8(d) shows how MA of CIFAR-100 changes.

References

[1] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[2] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *20th International Conference on Artificial Intelligence and Statistics*. PMLR, 2017.

[3] Nader Bouacida and Prasant Mohapatra. Vulnerabilities in federated learning. *IEEE Access*, 9:63229–63249, 2021.

[4] Mustafa Safa Ozdayi, Murat Kantarcioglu, and Yulia R Gel. Defending against backdoors in federated learning with robust learning rate. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9268–9276, 2021.

[5] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

[6] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

[7] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *23rd International Conference on Artificial Intelligence and Statistics*, volume 108. PMLR, 2020.

[8] Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. Data poisoning attacks against federated learning systems. In *Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25*, pages 480–501. Springer, 2020.

[9] Najeeb Moharram Jebreel, Josep Domingo-Ferrer, David Sánchez, and Alberto Blanco-Justicia. Lfighter: Defending against the label-flipping attack in federated learning. *Neural Networks*, 170:111–126, 2024.

[10] Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. Attack of the tails: Yes, you really can backdoor federated learning. *arXiv preprint arXiv:2007.05084*, 2020.

[11] Pei Fang and Jinghui Chen. On the vulnerability of backdoor defenses for federated learning. In *AAAI Conference on Artificial Intelligence*, 2023.

[12] Haomin Zhuang, Mingxian Yu, Hao Wang, Yang Hua, Jian Li, and Xu Yuan. Backdoor federated learning by poisoning backdoor-critical layers. 2024.

[13] Lingjuan Lyu, Han Yu, and Qiang Yang. Threats to federated learning: A survey. *arXiv preprint arXiv:2003.02133*, 2020.

[14] Liping Li, Wei Xu, Tianyi Chen, Georgios B. Giannakis, and Qing Ling. RSA: byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets. In *33rd AAAI Conference on Artificial Intelligence*, 2019.

[15] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*, 2019.

[16] Suyi Li, Yong Cheng, Wei Wang, Yang Liu, and Tianjian Chen. Learning to detect malicious clients for robust federated learning. *arXiv preprint arXiv:2002.00211*, 2020.

[17] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. Fltrust: Byzantine-robust federated learning via trust bootstrapping. In *Network and Distributed Systems Security (NDSS) Symposium*, 2021.

[18] Thien Duc Nguyen, Phillip Rieger, Roberta De Viti, Huili Chen, Björn B Brandenburg, Hossein Yalame, Helen Möllering, Hossein Fereidooni, Samuel Marchal, Markus Miettinen, et al. {FLAME}: Taming backdoors in federated learning. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1415–1432, 2022.

[19] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *30th Annual Conference on Neural Information Processing Systems*, 2017.

[20] Jacob Dumford and Walter Scheirer. Backdooring convolutional neural networks via targeted weight perturbations. In *2020 IEEE International Joint Conference on Biometrics (IJCB)*. IEEE, 2020.

[21] Minghong Fang, Minghao Sun, Qi Li, Neil Gong, Jin Tian, and Jia Liu. Data poisoning attacks and defenses to crowdsourcing systems. In *Proceedings of the Web Conference*, pages 969–980, 2021.

[22] Minghong Fang, Guolei Yang, Neil Zhenqiang Gong, and Jia Liu. Poisoning attacks to graph-based recommender systems. In *Proceedings of the 34th Annual Computer Security Applications Conference*, pages 381–392, 2018.

[23] Guolei Yang, Neil Zhenqiang Gong, and Ying Cai. Fake co-visitation injection attacks to recommender systems. In *NDSS*, 2017.

[24] Blaine Nelson, Marco Barreno, Fuching Jack Chi, Anthony D Joseph, Benjamin IP Rubinstein, Udam Saini, Charles A Sutton, J Doug Tygar, and Kai Xia. Exploiting machine learning to subvert your spam filter. *LEET*, 8:1–9, 2008.

[25] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[26] Yiming Li, Baoyuan Wu, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor learning: A survey. *arXiv preprint arXiv:2007.08745*, 2020.

[27] Lie He, Sai Praneeth Karimireddy, and Martin Jaggi. Byzantine-robust learning on heterogeneous datasets via resampling. *arXiv preprint arXiv:2006.09365*, 2020.

[28] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[29] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[30] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy*. IEEE, 2016.

[31] Sanghyun Hong, Pietro Frigo, Yiğitcan Kaya, Cristiano Giuffrida, and Tudor Dumitraș. Terminal brain damage: Exposing the graceless degradation in deep neural networks under hardware fault attacks. In *28th USENIX Security Symposium*, 2019.

[32] Benjamin IP Rubinstein, Blaine Nelson, Ling Huang, Anthony D Joseph, Shing-hon Lau, Satish Rao, Nina Taft, and J Doug Tygar. Antidote: understanding and defending against poisoning of anomaly detectors. In *9th ACM SIGCOMM Conference on Internet Measurement*, 2009.

[33] Ali Shafahi, W. Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *31st Annual Conference on Neural Information Processing Systems, NeurIPS*, pages 6106–6116, 2018.

[34] Octavian Suciu, Radu Marginean, Yigitcan Kaya, Hal Daume III, and Tudor Dumitras. When does machine learning fail? generalized transferability for evasion and poisoning attacks. In *27th USENIX Security Symposium*, 2018.

[35] Binghui Wang and Neil Zhenqiang Gong. Attacking graph-based classification via manipulating the graph structure. In *ACM CCS*, 2019.

[36] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress, 2012.

[37] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 19–35. IEEE, 2018.

[38] Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1885–1893, 2016.

[39] Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C Lupu, and Fabio Roli. Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 27–38, 2017.

[40] Huang Xiao, Battista Biggio, Gavin Brown, Giorgio Fumera, Claudia Eckert, and Fabio Roli. Is feature selection secure against training data poisoning? In *Proceedings of the 32nd International Conference on Machine Learning, ICML*, volume 37, pages 1689–1698. JMLR.org, 2015.

[41] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local model poisoning attacks to byzantine-robust federated learning. In *29th {USENIX} Security Symposium*, 2020.

[42] Gilad Baruch, Moran Baruch, and Yoav Goldberg. A little is enough: Circumventing defenses for distributed learning. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8632–8642, 2019.

[43] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. Fall of empires: Breaking byzantine-tolerant SGD by inner product manipulation. In *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI*, volume 115, pages 261–270. AUAI Press, 2019.

[44] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin B. Calo. Analyzing federated learning through an adversarial lens. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 634–643. PMLR, 2019.

[45] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. DBA: distributed backdoor attacks against federated learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[46] Yuntao Liu, Yang Xie, and Ankur Srivastava. Neural trojans. In *International Conference on Computer Design*. IEEE, 2017.

[47] Bao Gia Doan, Ehsan Abbasnejad, and Damith C Ranasinghe. Februus: Input purification defense against trojan attacks on deep neural network systems. In *Annual Computer Security Applications Conference*, 2020.

[48] Pu Zhao, Pin-Yu Chen, Payel Das, Karthikeyan Natesan Ramamurthy, and Xue Lin. Bridging mode connectivity in loss landscapes and adversarial robustness. *arXiv preprint arXiv:2005.00060*, 2020.

[49] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 2018.

[50] Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. *Advances in neural information processing systems*, 31, 2018.

[51] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv preprint arXiv:1811.03728*, 2018.

[52] Di Tang, XiaoFeng Wang, Haixu Tang, and Kehuan Zhang. Demon in the variant: Statistical analysis of dnns for robust backdoor contamination detection. In *30th USENIX Security Symposium*, 2021.

[53] Sebastien Andreina, Giorgia Azzurra Marson, Helen Möllering, and Ghassan Karame. Baffle: Backdoor detection via feedback-based federated learning. In *41st International Conference on Distributed Computing Systems*, pages 852–863. IEEE, 2021.

[54] Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith C Ranasinghe, and Surya Nepal. Strip: A defence against trojan attacks on deep neural networks. In *Proceedings of the 35th Annual Computer Security Applications Conference*, 2019.

[55] Mahesh Subedar, Nilesh Ahuja, Ranganath Krishnan, Ibrahima J Ndiour, and Omesh Tickoo. Deep probabilistic models to detect data poisoning attacks. *arXiv preprint arXiv:1912.01206*, 2019.

[56] Kaidi Jin, Tianwei Zhang, Chao Shen, Yufei Chen, Ming Fan, Chenhao Lin, and Ting Liu. A unified framework for analyzing and detecting malicious examples of dnn models. *arXiv preprint arXiv:2006.14871*, 2020.

[57] Bo Zhao, Peng Sun, Tao Wang, and Keyu Jiang. Fedinv: Byzantine-robust federated learning by inversing local model updates. In *36th AAAI Conference on Artificial Intelligence*, 2022.

[58] Kaiyuan Zhang, Guanhong Tao, Qiuling Xu, Siyuan Cheng, Shengwei An, Yingqi Liu, Shiwei Feng, Guangyu Shen, Pin-Yu Chen, Shiqing Ma, et al. Flip: A provable defense framework for backdoor mitigation in federated learning. In *International Conference on Learning Representations (ICLR)*, 2023.

[59] Ali Raza, Shujun Li, Kim-Phuc Tran, and Ludovic Koehl. Using anomaly detection to detect poisoning attacks in federated learning applications. *arXiv preprint arXiv:2207.08486*, 2022.

[60] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *International conference on machine learning*, pages 2089–2099. PMLR, 2021.

[61] Zhixiong Chen, Wenqiang Yi, Hyundong Shin, Arumugam Nallanathan, and Geoffrey Ye Li. Efficient wireless federated learning with partial model aggregation. *IEEE Transactions on Communications*, 2024.

[62] Krishna Pillutla, Kshitiz Malik, Abdel-Rahman Mohamed, Mike Rabbat, Maziar Sanjabi, and Lin Xiao. Federated learning with partial model personalization. In *International Conference on Machine Learning*, pages 17716–17758. PMLR, 2022.

[63] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818*, 2019.

[64] Canh T Dinh, Nguyen Tran, and Josh Nguyen. Personalized federated learning with moreau envelopes. *Advances in Neural Information Processing Systems*, 33:21394–21405, 2020.

[65] Jiahao Tan, Yipeng Zhou, Gang Liu, Jessie Hui Wang, and Shui Yu. **pFedSim**: Similarity-aware model aggregation towards personalized federated learning. *arXiv preprint arXiv:2305.15706*, 2023.

[66] Yansong Gao, Minki Kim, Sharif Abuadbba, Yeonjae Kim, Chandra Thapa, Kyuyeon Kim, Seyit A Camtepe, Hyoungshick Kim, and Surya Nepal. End-to-end evaluation of federated learning and split learning for internet of things. *arXiv preprint arXiv:2003.13376*, 2020.

[67] Zeyu Qin, Liuyi Yao, Daoyuan Chen, Yaliang Li, Bolin Ding, and Minhao Cheng. Revisiting personalized federated learning: Robustness against backdoor attacks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, page 4743–4755, New York, NY, USA, 2023. Association for Computing Machinery.

[68] Shanjiaoyang Huang, Weiqi Peng, Zhiwei Jia, and Zhuowen Tu. One-pixel signature: Characterizing cnn models for backdoor detection. In *European Conference on Computer Vision*, pages 326–341. Springer, 2020.

[69] Jianyi Zhang, Fangjiao Zhang, Qichao Jin, Zhiqiang Wang, Xiaodong Lin, and Xiali Hei. Xmam: X-raying models with a matrix to reveal backdoor attacks for federated learning. *Digital Communications and Networks*, 10(4):1154–1167, 2024.

[70] 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.

[71] Yann LeCun, Bernhard Boser, John Denker, Donnie Henderson, Richard Howard, Wayne Hubbard, and Lawrence Jackel. Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems*, 2, 1989.

[72] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[73] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE conference on computer vision and pattern recognition*, 2016.

[74] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[75] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *ICML*, pages 5650–5659. PMLR, 2018.