ARIBA: Towards Accurate and Robust Identification of Backdoor Attacks in Federated Learning

Yuxi Mi[1], Jihong Guan[2], and Shuigeng Zhou[1]

[1] Shanghai Key Lab of Intelligent Information Processing, and School of Computer Science, Fudan University, Shanghai 200438, China
{yxmi20, sgzhou}@fudan.edu.cn

[2] Department of Computer Science and Technology, Tongji University, Shanghai 201804, China
jhguan@tongji.edu.cn

Abstract. The distributed nature and privacy-preserving characteristics of federated learning make it prone to the threat of poisoning attacks, especially backdoor attacks, where the adversary implants backdoors to misguide the model on certain attacker-chosen sub-tasks. In this paper, we present a novel method ARIBA to accurately and robustly identify backdoor attacks in federated learning. By empirical study, we observe that backdoor attacks are discernible by the filters of CNN layers. Based on this finding, we employ unsupervised anomaly detection to evaluate the pre-processed filters and calculate an anomaly score for each client. We then identify the most suspicious clients according to their anomaly scores. Extensive experiments are conducted, which show that our method ARIBA can effectively and robustly defend against multiple state-of-the-art attacks without degrading model performance.

Keywords: Federated learning · Backdoor attack · Anomaly detection.

1 Introduction

Federated learning (FL) [15,22] has emerged as a leading deep learning framework for massively distributed training of models among a multitude of clients under the orchestration of a central server. FL is designated to take the advantage of the non-i.i.d. training data from thousands or even millions of clients while preserving their privacy, as the private data and the local training processes are not visible to any other parties.

However, the distributed architectural design and the lack of transparency in client updates make federated learning generically vulnerable to poisoning attacks [10,18,21,31]. An adversary introducing poisoning attacks tries to corrupt the global model at training time to degrade the model's performance on some subtasks or as a whole. Concretely, to achieve the attack goal, an attacker may alter the private training data [3] of some compromised clients, or rather, manipulate the model updates sent to the server [14,20].

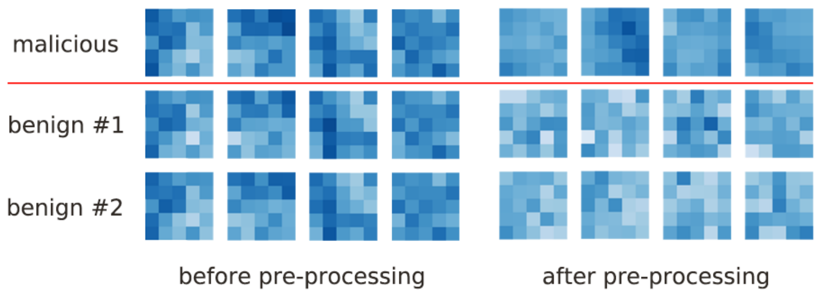

Fig. 1: Comparison between malicious and benign filters, before and after pre-processing. Filters directly sampled from clients' updates are similar. Yet, pre-processed filters show discernible differences, as a sign of backdoor attacks taking place. ARIBA identifies suspicious clients by exploiting these differences.

This paper focuses on the defense of a branch of poisoning attacks called *backdoor attacks*, where the corrupted model's performance on the attacker-chosen backdoor subtask(s) is degraded while retaining high performance on the primary task. For instance, a backdoored model of image classification may misclassify images that trigger certain backdoors into an attacker-desired class, while classifies the others correctly. Recent works [1,2,11,24] successfully implant backdoors of different types, including attacker-chosen images, single-pixel or pattern triggers, and semantic features.

Since backdoor attacks do not affect the performance of the primary task, it is more difficult to detect them than those attacks that degrade the entire model. Besides, the privacy-preserving design of federated learning prevents the server from auditing the integrity and honestness of clients, which makes backdoor attack detection a particularly challenging task.

The main branches of existing defenses include dedicated techniques [9,19,27], Byzantine-resilient aggregations [4,6,30], and those exploiting model traits such as accuracy and l2-norm [2]. Though these methods push the field a big leap forward, there is still much room to be improved. They have obvious limitations: requiring *a priori* knowledge (e.g. clean dataset) or empirical thresholds, relying on the assumption of data i.i.d.-ness, and easily being bypassed by an advanced attacker. More importantly, many of these methods depend on the *overall* characteristics of the model, which are too coarse for precise detection.

In this paper, we turn our attention to more *delicate* model characteristics. Specifically, by empirical study we observe that pre-processed backdoored and benign updates show discernible patterns, which can be exploited for attack detection. Fig. 1 shows some filters of CNN layers of malicious and benign updates before and after pre-processing. We can see the obvious difference in pre-processed filter patterns between malicious and benign clients. Based on this finding, we propose a new method ARIBA to accurately and robustly identify backdoor attacks. Our method first pre-processes the filters to obtain zero-centered gradients, then feeds them into an unsupervised anomaly detection algorithm, where their anomaly scores are evaluated. Following that, anomaly scores of clients are calculated, with which the most suspicious clients are identified finally. Our extensive experiments validate its effectiveness and robustness.

The main contributions of this paper are as follows: 1) We propose a novel idea of identifying backdoor attacks via the filter patterns of model updates; 2) We develop an effective and robust method ARIBA to implement the above idea by applying unsupervised anomaly detection over the filters, then evaluating their anomaly scores to detect the most suspicious clients. 3) We conduct extensive experiments to evaluate ARIBA, which shows that it can effectively and robustly defend the state-of-the-art backdoor attacks.

2 Related Work

2.1 Federated Learning (FL)

Federated learning [15,22] trains a deep neural network collaboratively by iteratively merging local updates to the global model. A federated learning framework consists of a central server and a group of clients, each with exclusive access to its private local data $\mathcal{D}_i$. At each round t, n clients are randomly chosen by the server to share the current global model weight w_G^t. Each chosen client updates w_G^t for several local epochs on its private data $\mathcal{D}_i$ to reach a new local weight w_i^{t+1} that is sent to the server. The server synchronizes these updates to obtain a renewed global weight w_G^{t+1} for the next iteration, by performing aggregation such as the standard FedAvg [22]:

$$w_G^{t+1} = w_G^t + \frac{\eta}{n} \sum_{i=1}^{n} (w_i^{t+1} - w_G^t), \tag{1}$$

where η is a task-specific global learning rate set by the server.

2.2 Poisoning Attacks

Poisoning attacks alter the training pipeline of a deep learning system to degrade model performance, which can take place in both centralized and federated settings. State-of-the-art attacks include data poisoning that manipulates the model's behavior by compromising some of the training data [12,23], and model poisoning by inserting malicious operations or components into the training process [13,32].

Backdoor attacks make up a more dedicated branch of poisoning attacks, where the attacker makes the trained model misbehave only on certain *backdoors*. Centralized backdoor attacks [5,11,21] have been well explored. These methods are not applicable for federated learning, since the malicious model updates, once produced, are quickly diluted in the aggregation with a massive multitude of benign models. In recently proposed federated backdoor attacks [1,2,24,26,29], the attacker survives the aggregation by boosting its update gradients. Baseline attackers carry out the attack directly while advanced attackers also take countermeasures to conceal themselves from possible detections.

2.3 Defenses

Existing defenses to backdoor attacks in FL roughly fall into three types: examining model traits, Byzantine-resilient aggregations, and dedicated defenses.

Examining model traits. The most intuitive defense in FL is to examine the overall statistical traits of candidate local updates. The server may check their test accuracy, or compare the l2-norm of their gradients [2]. These defenses are sometimes effective against a baseline attacker but can be easily bypassed by advanced ones. Moreover, these defenses usually require prior knowledge on thresholds, which is hardly available in real applications.

Byzantine-resilient aggregations. Aggregations such as Krum, coordinate-wise median, and GeoMed [4,6,30] are designed to eliminate local updates with skewed distributions. Most of them rely on the i.i.d.-ness of training data and the convexity of loss function. As recent research [1] shows, these defenses can hardly counteract backdoor attacks if proper attack strategies are chosen.

Dedicated defenses. Gu et al. [11] showed that a backdoor attack can activate certain compromised neurons. Based on this discovery, recent work [27] proposes a pruning defense to identify and remove suspicious neurons based on the ranking of their averaged activation values. This defense works only at inference time on pixel-pattern backdoors. The reported activation value also reveals the client's local information. FoolsGold [9] shares some common ground with our method in that both are based on the diversity of client updates. Yet, their method identifies attacks more coarsely by evaluating the cosine similarity between gradients while ours looks into the patterns of model filters. Recently, Li et al. [19] proposed a spectral anomaly detection framework that detects the malicious model updates in their low-dimensional embeddings. However, their framework requires extra public datasets and a centralized training process to provide unbiased model updates, which are hardly available in real-world federated scenarios.

3 Threat Model

The attacker's capability. The attacker has all the basic abilities of a standard model poisoning adversary. It takes full control of one or several (say l) compromised *malicious clients* $\{mal_1, \cdots, mal_l\}$. It can access and alter the local training data of all malicious clients, and manipulate the local training of malicious clients by changing objective functions or adjusting hyperparameters such as the learning rate. Once the local training is finished, the attacker can modify the trained weights before updating them to the server. However, the attacker has not access to or control of any benign client. In addition, we extend the attacker's ability by imposing three strong assumptions: (1) According to Shannon's maxim [25], the attacker is assumed to be aware of all possible defenses performed at the server-side. (2) Though FL by design chooses a fraction of clients in each iteration randomly, we assume all the malicious clients can be selected anytime at the attacker's will. (3) If the attacker has control of multiple malicious clients, these clients can share the attacking workload by collusion.

The attacker's goal. The attacker desires to replace the benign global model with an adversarial model, which retains high accuracy on the primary task, yet produces false outputs on chosen backdoors.

To this end, the attacker first train each compromised client locally on its malicious objective $\arg\min_{w_{mal_j}^{t+1}} \mathcal{L}_{mal}$ to obtain a trained weight $w_{mal_j}^{t+1}$. Note that $\{w_{mal_j}^{t+1}\}_{j\in[l]}$ are all very close to an attacker-desired w_{mal}^{t+1} since they share the same backdoor and malicious objective function. The attacker may also try to evade possible detection by adding anomalous terms to the loss. Assume that FedAvg is performed, the attacker then takes a "scaling" strategy to survive the averaging. Especially, we generalize the strategy to a group of colluded malicious clients. As the global model converges, the update gradients of $(n-l)$ benign clients will be cancelled out as $\sum_{i\in[n\setminus l]}(w_i^{t+1}-w_G^t)\approx 0$. Therefore, the attacker can calculate a scaling factor $\alpha=\frac{\tilde{n}}{\eta}$, where $\tilde{n}$ is an estimation of the number of participated clients, and assigns part of the attacking workload to each malicious client: $\alpha=\sum_{j\in[l]}\alpha_j$. The latter then uploads $\tilde{w}_{mal_j}^{t+1}=w_G^t+\alpha_j(w_{mal_j}^{t+1}-w_G^t)$. Therefore, the global model w_G^{t+1} will be replaced by the desired w_{mal}^{t+1} as

$$\begin{aligned} w_G^{t+1} &= w_G^t+\frac{\eta}{n}\sum_{i=1}^{n}(w_i^{t+1}-w_G^t) \\ &= w_G^t+\frac{\eta}{n}\left(\sum_{j\in[l]}(\tilde{w}_{mal_j}^{t+1}-w_G^t)+\sum_{i\in[n\setminus l]}(w_i^{t+1}-w_G^t)\right) \\ &\approx w_G^t+\frac{\eta}{n}\sum_{j\in[l]}(\tilde{w}_{mal_j}^{t+1}-w_G^t) \\ &\approx w_G^t+\frac{\tilde{n}}{n}(w_{mal}^{t+1}-w_G^t)\approx w_{mal}^{t+1}. \end{aligned} \tag{2}$$

The attack scenarios. We consider three types of state-of-the-art backdoors in image classification: (1) *Targeted backdoors.* The attacker holds a target dataset of one or several images with false labels and performs training on the dataset. The trained global model is expected to misclassify the same set of images if they appear again at inference time. (2) *Pixel-pattern backdoors.* The attacker picks some training images from a certain class and overlays them with a pattern composed of a single or a group of bright pixel(s). Images with the same pattern

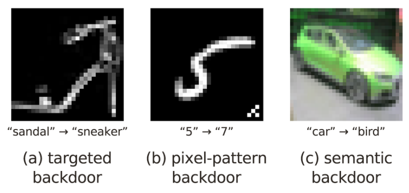

Fig. 2: Examples of three types of backdoors. Here, triggers are (a) one chosen "sandal" image falsely labeled as "sneaker", (b) a fraction of "5"s, each overlayed with a group of bright pixels on one corner, and (c) some "car"s painted in green.

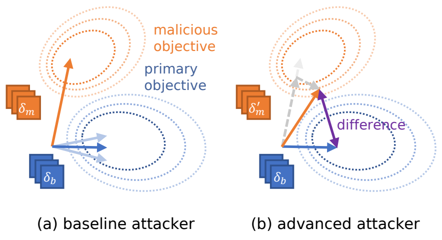

Fig. 3: An illustrative explanation of our idea. (a) The benign (blue) and malicious (orange) gradients differ in magnitude and direction. (b) The attacker may change its gradient to imitate the benign ones. Yet, a delicate algorithm can still tell the difference.

should be misclassified to the attacker-desired class. (3) *Semantic backdoors.* Instead of adding artificial patterns, the attacker chooses a naturally occurring feature (e.g. cars painted in green) as the backdoor. This allows the attacker to trigger it without accessing or modifying any inference-time image. Fig. 2 shows examples of three types of backdoors.

4 Method

4.1 Basic Idea and Rationale

Gu et al. [11] showed that pixel-pattern attacks can introduce compromised neurons that fire only in the presence of backdoors. However, such a neuron-wise pattern is so delicate that is discernible only when the attacker sticks to a very specific, fixed backdoor. Furthermore, to detect these neurons, a reference dataset must be exploited to account for the activation, which is not available in federated settings.

We conducted empirical studies on the behaviors of both malicious and benign clients in three different attack scenarios mentioned above. We found that *backdoor attacks can introduce discernible patterns in model weights.* Such patterns can serve as much more discernible clues for detecting malicious clients than the model's overall traits such as test accuracy and distance. As the examples in Fig. 1 show, for a baseline attacker, the difference between pre-processed filters can be easily recognized by human eyes. More sophisticated malicious clients may conceal their intention by taking covert measures or colluding with each other (*i.e.*, engaging Sybil attacks), making the patterns less discernible. Yet, a properly designed algorithm (unsupervised anomaly detection algorithm in this paper) can still detect these patterns with high confidence.

However, is this finding reasonable? We answer this by looking into the principle of backdoor attacks. As illustrated in Fig. 3, when an attack takes place,

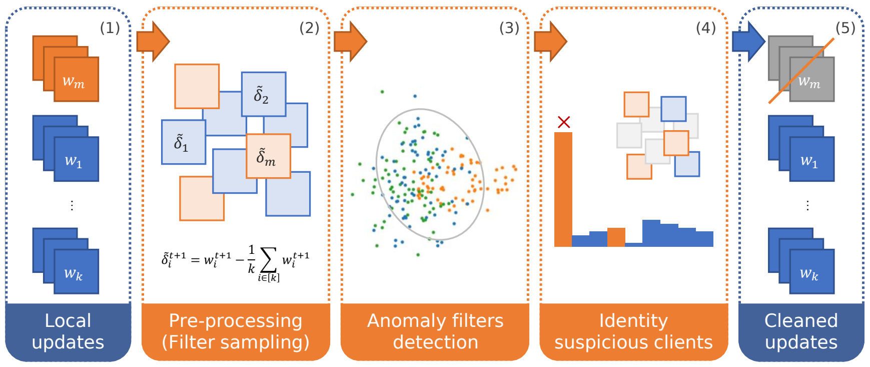
(1)
w_m
w_1
w_k
Local updates
(2)
$\tilde{\delta}_2$
$\tilde{\delta}_1$
$\tilde{\delta}_m$
$\tilde{\delta}_i^{t+1} = w_i^{t+1} - \frac{1}{k}\sum_{i\in[k]} w_i^{t+1}$
Pre-processing (Filter sampling)
(3)
Anomaly filters detection
(4)
Identity suspicious clients
(5)
w_m
w_1
w_k
Cleaned updates

Fig. 4: The ARIBA Framework. ARIBA works as follows: the server (1) collects local updates from clients, (2) pre-processes model weights to obtain discernible filters, (3) detects all anomalous filters via an unsupervised anomaly detection algorithm, (4) calculates the anomaly score of each client, and identifies those with the highest and lowest scores as attackers, and removes them, finally (5) uses the remaining local updates for aggregation. Here, blue/organge indicates benign/malicious weights or filters.

malicious local models approach the backdoor objective with the accumulation of their gradients, while the others head for different benign objectives. The malicious models will thus produce gradients of distinct magnitude and direction. Certainly, the attacker may conceal its intention by adding constraints to malicious gradients, making them less notable. However, the difference always exists as long as the attack takes place. Such difference will inevitably incur the difference in model weights, *wlog.*, in the filters of CNN layers.

4.2 Framework and Algorithm

Based on the idea above, we propose a novel method called ARIBA to support *A*ccurate and *R*obust *I*dentification of *B*ackdoor *A*ttacks. Fig. 4 illustrates the ARIBA framework.

ARIBA can serve as a plug-in block in most federated learning systems. It works in a three-step way: first, we pre-process model weights to obtain discernible patterns of filters in a CNN layer. Then, we collect the filters and feed them into an unsupervised anomaly detection algorithm to identify suspicious filters. After that, we calculate an anomaly score for each client based on the number of identified anomalous filters. Finally, we detect the most suspicious clients as attackers according to clients' anomaly scores. The entire process is described in Alg. 1.

Algorithm 1 ARIBA

```
Input: Clients' updates $\{w_i^{t+1}\}_{i\in[n]}$ of round $t$
Output: Cleaned updates
 1: Choose a CNN layer.
 2: for all $i \in [n]$ do
 3:     Obtain zero-centered gradients: $\tilde{\delta}_i^{t+1} \leftarrow w_i^{t+1} - \frac{1}{n}\sum_{i\in[n]} w_i^{t+1}$.
 4: end for
 5: Obtain filters $\{f_{i\in[n],k\in[m]}\} \leftarrow \{\tilde{\delta}_i^{t+1}\}$.
 6: Evaluate filters' anomaly scores: $\{s_{i,k}\}\leftarrow$FixAndPred$(\{f_{i,k}\})$.
 7: for all $i \in [n]$ do
 8:     Calculate clients' anomaly scores: $S_i \leftarrow \sum_{k\in[m]} s_{i,k}$.
 9: end for
10: Mark clients with the highest and lowest anomaly scores as attackers, and remove
    their ids $\{c\}$.
11: return $\{w_i^{t+1}\}_{i\in[n]\setminus\{c\}}$
```

4.3 Techniques

Pre-processing. Instead of identifying certain backdoor neurons, we focus on filter-wise difference, which is more robust to noise. We start with pre-processing the model weights. In each training round, the server collects from clients their local model weights $\{w_i^{t+1}\}_{i\in[n]}$, which are accumulated by rounds of gradients. Here, we consider only the latest gradient δ_i^{t+1}. Since the server always has access to the global model, δ_i^{t+1} can be evaluated by:

$$\delta_i^{t+1} = w_i^{t+1} - w_G^t. \tag{3}$$

Note that we omit the global learning rate η for simplicity, which does not impact our derivation. For better differentiating the gradients of benign and malicious clients, we do mean-subtraction for all gradients as follows:

$$\tilde{\delta}_i^{t+1} = w_i^{t+1} - \frac{1}{n}\sum_{i\in[n]} w_i^{t+1} \tag{4}$$

With the zero-centered gradients, we take a CNN layer (*wlog.*, the first CNN layer) and extract all of its filters. As an example, we use principal component analysis to visualize the filters' distribution. Fig. 5 shows that these filters do expose discernible patterns, depending on the attacker's capability. Based on the patterns, later we identify the malicious filters by exploiting a distance-based anomaly detection algorithm.

Detecting anomalous filters. Here, we address how to identify anomalous filters. Actually, many unsupervised anomaly detection algorithms can do this work. But which one should we use? We emphasize *two observations*:

1) *The decisions on individual filters do not matter much.* In most cases, we do not expect an anomaly detection algorithm to judge single filters with

Algorithm 2 `FixAndPred` (Mahalanobis Distance)

```
Input: Pre-processed filters $\{f_{i,k}\}$
Parameter: An estimated fraction of anomalous filters $\gamma$
Output: Prediction $\{s_{i,k}\}$
1: Calculate the convariance matrix $\Sigma \leftarrow cov(\{f_{i,k}\})$.
2: Calculate the average of filters $\bar{f} \leftarrow mean(\{f_{i,k}\})$.
3: for all $i, k$ do
4:     Calculate the Mahalanobis distance of $f_{i,k}$: $d_{i,k} \leftarrow \sqrt{(f_{i,k} - \bar{f})^T \Sigma^{-1} (f_{i,k} - \bar{f})}$.
5: end for
6: Set $s_{i,k} \leftarrow 1$ for the $\gamma$-largest $d_{i,k}$
7: Set $s_{i,k} \leftarrow 0$ for the others.
8: return $\{s_{i,k}\}$
```

very high precision. After all, it is not easy to completely see through a 'smart' backdoor attacker. Instead, we treat each client's filters as a whole and pay more attention to the overall state by counting the total numbers of normal and anomalous filters, *i.e.*, a client may be malicious if it has significantly more (or less, as discussed later) anomalous filters than the normal ones. For instance, each of the three clients in Fig. 5(b) has a number of hard-to-judge filters (or outliers/anomalies). Yet, by putting these filters together, we can see that the client in orange is more anomalous because it is less overlapped with the other two clients. In summary, it is acceptable that an anomaly detection algorithm incorrectly reports some filters, as long as it can accurately detect the overall difference between benign and malicious filters.

2) *Algorithm efficiency does matter*. In real-world scenarios, thousands of clients may participate, each has thousands of filters. The detection can therefore be very expensive. Although the workload can be reduced by filter sampling, the number of filters is still very large since by design the defender should examine all the clients. Hence, the chosen algorithm should be of low time complexity.

Based on the above observations and after empirical evaluations, we choose a Mahalanobis-distance-based detection algorithm [7], as described in Alg. 2. Note our aim is to identify backdoor attacks, developing anomaly detection algorithm is not our focus. The purpose that we choose the Mahalanobis-distance-based algorithm is to validate our method. Surely there are many other sophisticated algorithms that can be used, and any defender is free to choose its task-specific algorithm. Assume each of the n clients has m filters, denoted as $f_{i,k}$ ($i \in [n], k \in [m]$), its anomaly score $s_{i,k}$ is evaluated by:

$$\{s_{i,k}\} = \texttt{FixAndPred}(\{f_{i,k}\}),$$
$$where\ s_{i,k} = \begin{cases} 0 & if\ f_{i,k}\ is\ normal \\ 1 & if\ f_{i,k}\ is\ anomalous. \end{cases} \tag{5}$$

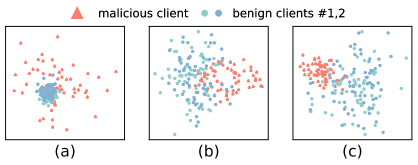

Fig. 5: Possible statistical distributions of filters. The malicious filters may appear more (a) or less (c) scattered than the benign ones. Advanced attacks may produce hard-to-judge points. Yet, the malicious client can still be identified by looking into the overall trend (b).

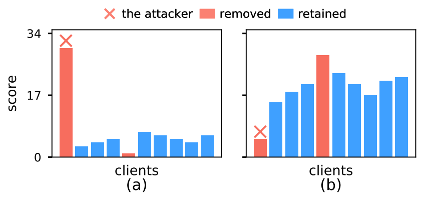

Fig. 6: Examples of two possible anomaly score distributions. The malicious client will obtain a significantly (a) higher or lower (b) anomaly score than other benign clients. The difference is distinguishable for ARIBA.

Identifying suspicious clients. We identify the most suspicious clients as attackers by summing up $\{s_{i,k}\}$ for each client i to obtain its anomaly score S_i:

$$S_i = \sum_{k \in [m]} s_{i,k}. \tag{6}$$

Fig. 6 shows two examples of client anomaly score distribution. The scores of most benign clients are expected to fall within a narrow range. A malicious client, if participated, should has a significantly higher or lower score. It is quite natural to regard a client with a higher score (more anomalous) as a malicious one. Yet, as shown in Fig. 6(b), a malicious client can also has a lower score. Empirically, this usually happens to advanced attackers attempting to conceal their intention, who train on both malicious and primary objectives in parallel. The gradients of the two objectives, if heading for opposite directions, may cancel each other out. Put it in anomaly detection, its filters may appear less scattered than what benign filters should look like, as shown in Fig. 5(c). Ergo, the detection algorithm may regard the malicious filters as normal ones, consequently marks the corresponding clients as benign ones instead. However, in either way, the difference between scores can still provide us significant evidence to recognize the suspicious clients.

After identifying the suspicious clients, the server performs aggregation on the model weights of the rest as usual.

5 Performance Evaluation

5.1 Experimental Setup

We apply our method to the three attack scenarios described in Sec. 3, *i.e.*, targeted attack, pixel-pattern attack, and semantic attack. We consider three commonly used image classification datasets: Fashion-MNIST [28] of grayscale cloths, MNIST [17] of handwritten digits, and CIFAR-10 [16], respectively. For Fashion-MNIST and MNIST, we apply a CNN with two convolutional layers

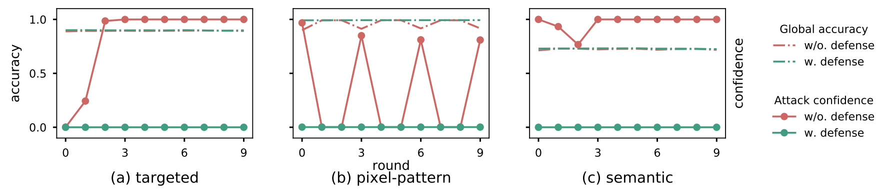

Fig. 7: ARIBA's primary defending results. Red: Without defense, all three attackers succeed with high confidence. Green: Yet, after implementing our ARIBA method, 100% of the attacks are successfully identified and blocked, and the model's performance is maintained.

and two fully connected layers. This is a standard architecture for such tasks. As for CIFAR-10, we use a real-world ResNet18. We implement the method using PyTorch 1.8.1 and CUDA 11.2. Our experiments are done on two Nvidia Titan Xp GPUs each with 12GB RAM, and Ubuntu 18.04 LTS OS.

By default, a group of 10 clients is selected in each experiment, one of them is malicious. For simplicity, the global learning rate is set to $\eta = 1$, *i.e.*, the updated global model replaces the previous one completely. We first train each model via standard federated learning till it reaches a pre-specified test accuracy: 0.9 for Fashion-MNIST, 0.99 for MNIST, and 0.74 for CIFAR-10.

Then, we reproduce the attacks. We set the backdoors following Fig. 2. We apply a local learning rate η_{local}=1e-3, 1e-4, 1e-4 and train $E_{local} = 5, 10, 2$ epochs for three attacks, respectively. We assume that the targeted attacker participates every round, the pixel-wise and semantic attackers are chosen every 3 rounds since the latter two are designed to be one-shot attack. Before aggregation, the malicious client boosts its update by α=10 to survive FedAvg. All attacks succeed as they implant the backdoors with high confidence, while incurring very slight impact on the accuracy of the primary task. The results are illustrated in Fig. 7.

5.2 Primary Defense Results

We then plug in ARIBA before the aggregation and rerun each attack. Fig. 7 shows that our method successfully blocks all three attacks by identifying all the malicious attempts, without degrading the global model.

Note the result defending a single malicious client is discrete: either success (100%) or failure (0%). To quantitively evaluate our defense, we introduce *defense confidence* C, to measure how confident the server is about the result:

$$C = \frac{1}{\log_2 m} \times \max\left(\log_2 \frac{S_{mal}}{\max\left(\{S_{i\in[n]\setminus mal}\}\right)}, \log_2 \frac{\min\left(\{S_{i\in[n]\setminus mal}\}\right)}{S_{mal}}\right). \tag{7}$$

Here, S_i, S_{mal} is the anomaly score of client i and the malicious client, respectively. m is the number of filters. This metric is actually quite straightforward.

Table 1: Comparison with existing defenses. Here, "✓": success; "✗": failure; "-": no experimental results reported.

Defense	T.	P.	S.
Test accuracy [2]	✓	✗	✗
L2-norm [2]	✓	✓	✓
Krum [4]	✗	✗	✗
Coo-med [30]	✗	✗	✗
Wu et al. [27]	-	✓	-
FoolsGold [9]	-	✓	-
Li et al. [19]	-	-	✓
ARIBA	✓	✓	✓

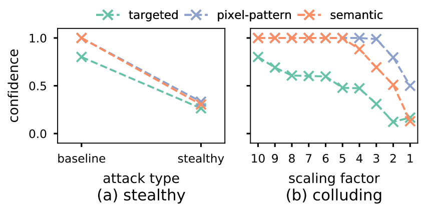

Fig. 8: ARIBA can perform effective defense against advanced attackers, including (a) a norm bounding attacker, and (b) multiple colluding clients, each with a smaller scaling factor α_j.

The defense succeeds if and only if S_m is the largest or smallest, in both cases we have $C>0$. C gradually increases as the difference between malicious and benign filters becomes more discernible. For the three attacks, our ARIBA method achieves high confidence $C = 0.8$, 1.0 and 1.0, respectively.

5.3 Comparison with Existing Methods

Here we compare our method with major existing defenses described in Sec. 2. Results are summarized in Tab. 1. Here T./P./S. stand for targeted, pixel-pattern and semantic attacks, respectively. Our method successfully defend all three attacks, while there are shortcomings in each of the existing ones.

We first check prior defenses that examine model traits, such as the model's test accuracy and l2-norm. Note, due to the non-i.i.d.-ness of data, federated learning is designed to accept updates that appear slightly different (*i.e.*, updates under a threshold). For test accuracy, we assume a threshold of 10%, the same as [2]. Most malicious updates fall in the range, nullifying the defense. Norm bounding defense is effective against all baseline attacks. Yet, as we point out below, an advanced attacker can conceal itself from the norm detection.

To compare with the results of Byzantine-resilient aggregations, we replace FedAvg with Krum [4] and Coo-med [30]. Results show that the attacker can bypass them by choosing a smaller scaling factor α=1. Note it is within the attacker's capability to choose α according to server-side defense mechanisms.

For dedicated defenses, we quote the results of [9,19,27]. Those defenses are attack-specific and are not likely to be generalized against other defenses, since they all expect the attacker to meet certain conditions. For instance, the attacker in [19] is assumed to possess i.i.d. images as the benign ones apart from the overlayed pattern, which does not hold for semantic scenario. The attacker in [27] should massively revise the dataset to produce enough differences in vector space, which is not true for targeted attacks. In [9], gradient updates of attackers should have high cosine similarity.

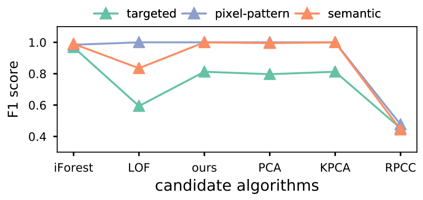

Fig. 9: Performance comparison of different anomaly detection algorithms on three attacks.

Table 2: Computation time (sec.) vs. #filters ($\times 10^3$).

Algorithm	**0.5**	**1**	**2**	**4**
iForest	0.48	0.88	1.47	2.59
LOF	0.23	0.6	1.67	5.03
ARIBA	**0.04**	**0.09**	**0.23**	**0.51**
PCA	0.38	1.16	1.49	2.82
KPCA	18.75	190.22	>300	>300
RPCC	0.4	0.88	1.57	2.77

5.4 Defending Against Advanced Attacks

As mentioned in Sec. 1, advanced attackers can take covert measures to evade detection. We here show ARIBA is still effective against these attackers.

Anomalous terms in objective function. Recent works [1,2] describe a stealthy attacker that conceal its intention by adding an anomalous term $\mathcal{L}_{ano}$ to its training objective: $\arg\min_{w_{mal}^{t+1}} \mathcal{L}_{mal} + \rho\mathcal{L}_{ano}$, where ρ is a hyperparameter. Specifically, we choose $\mathcal{L}_{ano} = \|\delta_{mal}^{t+1} - \delta_G^t\|_2$ and append it to all three attacks. The advanced attack nullifies the norm bounding defense in Sec. 5.3. Our defense results are shown in Fig. 8(a). The decrease in C shows the anomalous term does play its role. Yet, ARIBA still manages to identify the attacker as $C \geq 0.35$.

Multiple colluding clients. An advanced attacker may also perform Sybil attacks by controlling multiple malicious clients. By assigning to each client a smaller scaling factor α_j, individual malicious updates become less notable. We change α_j to simulate multiple colluding clients and rerun the defense. Fig. 8(b) shows that our method is effective ($C>0$) against different α_j. Note $\alpha_j<2$ is actually beyond our scope of discussion since an attacker manipulating a majority of clients already takes full control of the FL system.

5.5 Ablation Study

Comparison with different anomaly detection techniques. Recall in Sec. 4 we choose the anomaly detection algorithm based on its robustness and efficiency. Here, we compare our choice with 5 other commonly used ones: Isolation Forest, LOF, PCA, KPCA, and RPCC. We test their performance in all three attack scenarios. Results of F1-score in Fig. 9 show that LOF and RPCC perform less stably than the others.

Then, we check how their computation costs change with the number of filters. We run each experiment three times and measure the cost by average time. Results in Tab. 2 indicate that our chosen algorithm is the most cost-efficient among all the compared algorithms. Note we choose the fastest and

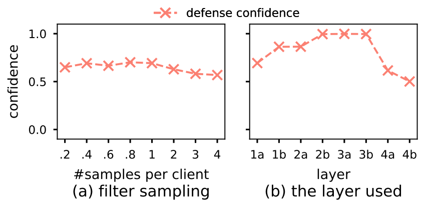

Fig. 10: Performance on (a) different numbers of filters sampled ($\times 10^2$) and (b) different CNN layers where filters are selected from.

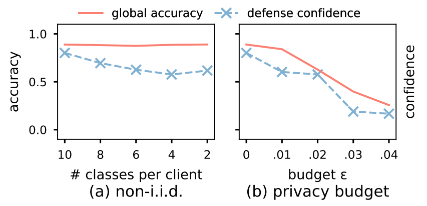

Fig. 11: Performance of ARIBA on noise-tolerant settings where (a) data is non-i.i.d., and (b) Gaussian noise is added for differential privacy.

relatively robust algorithm for the sole purpose to validate our method. The defender is free to replace it with more sophisticated, task-specific algorithms.

Filter sampling. The computation cost can be further reduced by filter sampling, *i.e.*, choosing a fraction of filters from each client. To illustrate that, we sample the filters by different proportions and compare their detection confidences. Results in Fig. 10(a) show that filter sampling incurs almost no performance loss while saving lots of computation cost.

Performance on different CNN layers. By default, we sample filters from the first CNN layer. Now we generalize this setting. We enumerate each CNN layer for targeted attack, as illustrated in Fig. 10(b). Results show confidence is always satisfying (C>0.5) regardless of the layer we choose. On the other hand, layers with larger size (layers 1b–3b) seem to provide better performance.

5.6 Robustness to Noise

Noise in training may jeopardize the detection. It adds random disturbance to the filters and interferes with the difference between them, consequently degrading the anomaly detection performance. In FL, noise can come from natural and manmade sources. To test if ARIBA is robust to noise, we consider two settings:

Noise from non-i.i.d. data. In the real world, the non-i.i.d.-ness of local training data can produce natural noise. We produce unbalanced data following [2]. For each client, we randomly select a fraction of classes and assign data only from the selected classes. We test ARIBA for different numbers of classes sampled. Fig. 11(a) shows that ARIBA performs well (C>0.5) under unbalanced data.

Noise from differential privacy. Differential privacy (DP) [8] is the *de facto* standard privacy-preserving technique. It deliberately introduces noise to protect clients from inference attacks, while as a side effect, degrading the model.

We simulate local DP by adding Gaussian noise to the updates of all benign clients, according to different privacy budgets ϵ. Fig. 11(b) shows that ARIBA can tolerate Gaussian noise as long as the global model is usable ($\epsilon<0.2$).

6 Conclusion

In this paper, we propose a novel method to identify backdoored updates by exploiting unsupervised anomaly detection. It is based on our finding that backdoor attacks expose discernible patterns in model weights. Concretely, we first pre-process the filters to obtain zero-centered gradients, then feed them into an unsupervised anomaly detection algorithm, where the filters are evaluated to calculate an anomaly score for each client. Finally, we identify the most suspicious clients according to their anomaly scores. Extensive experiments show that though our method is simple, it is efficient, accurate and robust, as well as widely applicable.

References

1. Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., Shmatikov, V.: How to backdoor federated learning. In: International Conference on Artificial Intelligence and Statistics. pp. 2938–2948. PMLR (2020)
2. Bhagoji, A.N., Chakraborty, S., Mittal, P., Calo, S.: Analyzing federated learning through an adversarial lens. In: International Conference on Machine Learning. pp. 634–643. PMLR (2019)
3. Biggio, B., Nelson, B., Laskov, P.: Poisoning attacks against support vector machines. arXiv preprint arXiv:1206.6389 (2012)
4. Blanchard, P., El Mhamdi, E.M., Guerraoui, R., Stainer, J.: Machine learning with adversaries: Byzantine tolerant gradient descent. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. pp. 118–128 (2017)
5. Chen, X., Liu, C., Li, B., Lu, K., Song, D.: Targeted backdoor attacks on deep learning systems using data poisoning. arXiv preprint arXiv:1712.05526 (2017)
6. Chen, Y., Su, L., Xu, J.: Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. Proceedings of the ACM on Measurement and Analysis of Computing Systems **1**(2), 1–25 (2017)
7. De Maesschalck, R., Jouan-Rimbaud, D., Massart, D.L.: The mahalanobis distance. Chemometrics and intelligent laboratory systems **50**(1), 1–18 (2000)
8. Dwork, C.: Differential privacy: A survey of results. In: International conference on theory and applications of models of computation. pp. 1–19. Springer (2008)
9. Fung, C., Yoon, C.J., Beschastnikh, I.: Mitigating sybils in federated learning poisoning. arXiv preprint arXiv:1808.04866 (2018)
10. Goldblum, M., Tsipras, D., Xie, C., Chen, X., Schwarzschild, A., Song, D., Madry, A., Li, B., Goldstein, T.: Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses. arXiv preprint arXiv:2012.10544 (2020)
11. Gu, T., Dolan-Gavitt, B., Garg, S.: Badnets: Identifying vulnerabilities in the machine learning model supply chain. arXiv preprint arXiv:1708.06733 (2017)

12. Huang, L., Joseph, A.D., Nelson, B., Rubinstein, B.I., Tygar, J.D.: Adversarial machine learning. In: Proceedings of the 4th ACM workshop on Security and artificial intelligence. pp. 43–58 (2011)
13. Ji, Y., Zhang, X., Ji, S., Luo, X., Wang, T.: Model-reuse attacks on deep learning systems. In: Proceedings of the 2018 ACM SIGSAC conference on computer and communications security. pp. 349–363 (2018)
14. Kairouz, P., McMahan, H.B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A.N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al.: Advances and open problems in federated learning. arXiv preprint arXiv:1912.04977 (2019)
15. Konečnỳ, J., McMahan, H.B., Yu, F.X., Richtárik, P., Suresh, A.T., Bacon, D.: Federated learning: Strategies for improving communication efficiency. arXiv preprint arXiv:1610.05492 (2016)
16. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
17. LeCun, Y., Cortes, C., Burges, C.: Mnist handwritten digit database (2010)
18. Li, Q., Wen, Z., Wu, Z., Hu, S., Wang, N., Li, Y., Liu, X., He, B.: A survey on federated learning systems: vision, hype and reality for data privacy and protection. arXiv preprint arXiv:1907.09693 (2019)
19. Li, S., Cheng, Y., Wang, W., Liu, Y., Chen, T.: Learning to detect malicious clients for robust federated learning. arXiv preprint arXiv:2002.00211 (2020)
20. Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: Challenges, methods, and future directions. IEEE Signal Processing Magazine **37**(3), 50–60 (2020)
21. Liu, Y., Ma, S., Aafer, Y., Lee, W.C., Zhai, J., Wang, W., Zhang, X.: Trojaning attack on neural networks. NDSS (2018)
22. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics. pp. 1273–1282. PMLR (2017)
23. Steinhardt, J., Koh, P.W., Liang, P.: Certified defenses for data poisoning attacks. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. pp. 3520–3532 (2017)
24. Sun, Z., Kairouz, P., Suresh, A.T., McMahan, H.B.: Can you really backdoor federated learning? arXiv preprint arXiv:1911.07963 (2019)
25. van Tilborg, H.C.A., Jajodia, S. (eds.): Shannon's Maxim, pp. 1194–1194. Springer US, Boston, MA (2011)
26. Wang, H., Sreenivasan, K., Rajput, S., Vishwakarma, H., Agarwal, S., Sohn, J.y., Lee, K., Papailiopoulos, D.: Attack of the tails: Yes, you really can backdoor federated learning. arXiv preprint arXiv:2007.05084 (2020)
27. Wu, C., Yang, X., Zhu, S., Mitra, P.: Mitigating backdoor attacks in federated learning. arXiv preprint arXiv:2011.01767 (2020)
28. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747 (2017)
29. Xie, C., Huang, K., Chen, P.Y., Li, B.: Dba: Distributed backdoor attacks against federated learning. In: International Conference on Learning Representations (2019)
30. Yin, D., Chen, Y., Kannan, R., Bartlett, P.: Byzantine-robust distributed learning: Towards optimal statistical rates. In: International Conference on Machine Learning. pp. 5650–5659. PMLR (2018)
31. Zhang, J., Chen, J., Wu, D., Chen, B., Yu, S.: Poisoning attack in federated learning using generative adversarial nets. In: 2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th

IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE). pp. 374–380. IEEE (2019)

32. Zou, M., Shi, Y., Wang, C., Li, F., Song, W., Wang, Y.: Potrojan: powerful neural-level trojan designs in deep learning models. arXiv preprint arXiv:1802.03043 (2018)