Robust federated learning based on voting and scaling

Xiang-Yu Liang
School of Computer Science
Southwest Petroleum University
Chengdu 610500, China
liangxyswpu@163.com

Heng-Ru Zhang*
School of Computer Science
Southwest Petroleum University
Chengdu 610500, China
zhanghrswpu@163.com

Fan Min
School of Computer Science
Southwest Petroleum University
Chengdu 610500, China
minfan@swpu.edu.cn

Wei Tang
School of Computer Science
Southwest Petroleum University
Chengdu 610500, China
twei1123@163.com

***Abstract*—Federated learning is vulnerable to poisoned attacks due to the inability to verify the authenticity of local data. Existing robust federated learning methods maintain a global model by discarding potentially risky local updates. However, they generally assume that the server knows the number of potentially abnormal clients. In this paper, we propose a robust federated learning method based on voting and scaling that relaxes such assumption. Malicious updates usually manifest in abnormal direction and magnitude. On one hand, the server computes the relative-angle between the target and other local updates. Angles greater than 90° are considered negative votes, otherwise positive votes. If the negative votes exceeds a predefined threshold, the target is considered abnormal. On the other hand, the server computes the magnitude median of the remaining updates after filtering out updates in abnormal directions. The magnitudes of local updates above/below the median are scaled down/increased. Experiments were carried out on five datasets in comparison to six state-of-the-art algorithms. Results show that our method can effectively improve the robustness of FL.**

***Index Terms*—Abnormal direction, Abnormal magnitude, Federated learning, Scaling, Voting.**

I. Introduction

Federated learning (FL) is a collaborative machine learning method [1] that does not require shared datasets. There are two roles in this method [2]. One is the server, which is usually performed by a high-performance computer provided by a service provider. The other is the client, which typically consists of edge computing devices with limited resources. Multiple clients collaborate to train the global model by exchanging local updates under the coordination of the server, while keeping the client's training data localized [3]. The advantage of FL is that it does not need to collect training data from the client, which is where it differs from traditional centralized machine learning.

Federated learning cannot verify the authenticity of training data due to its distributed nature, which makes FL vulnerable to poisoned attacks [4]. In a poisoned attack, the attacker can manipulate some clients at will. Thus, the attacker can aggregate malicious updates into the global model [5]. Existing malicious update generation methods are generally divided into two strategies [6]. One is to directly modify the local data of the controlled client to train malicious updates [7]. This strategy typically predicts an attacker-chosen target test example as an attacker-chosen target label (called targeted attacks). The other is to directly modify the local update of the controlled client [8]. This strategy makes the target model indiscriminately make false predictions on a large number of test examples (called untargeted attacks).

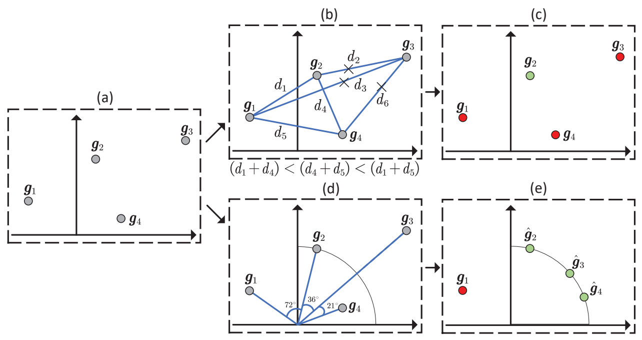


Fig. 1: Comparison of Krum with our method. (a) The local uptates. (b) The anomaly removal with Krum. For each update, Krum discards the m largest distances and computes the sum of the distances. The m is the number of abnormal clients obtained by the server. (c) The aggregation update of Krum. Krum selects the local update corresponding to the smallest sum of distances to maintain the global model. (d) The anomaly flitering and magnitude scaling with our method. For each update, we compute its angle with other local updates. If the negative votes exceeds a predefined threshold, the target is considered abnormal. (e) The aggregation update of our method. Filtered local updates are scaled.

To resist poisoned attacks, robust FL maintains the global model by discarding potentially risky local updates [9]. Some more advanced defenses can identify abnormal behavior in encrypted environments [10]. Existing defenses against poisoned attacks can be roughly divided into two categories [11]. One is to identify and remove potentially poisoned updates through anomaly detection methods. They make detailed assumptions about adversary attack strategies and/or the underlying distribution of benign or adversarial datasets. The other is to recast existing techniques developed in Byzantine fault-tolerant distributed learning into a FL setting. Their core idea is to compare client's local model updates and remove statistical outliers before using them to update the global model. However, these methods typically assume that the server can obtain the maximum number of abnormal clients.

In this paper, we propose robust federated learning based on voting and scaling (VSRFL), which relaxes this assumption. Tainted local updates typically exhibit two phenomena. One of them is that there are a few local updates that exhibit large angular deviations from other local updates. For these potentially risky local updates, we filter them out through a voting mechanism. Another phenomenon is the uneven amplitude between local updates. We mitigate this phenomenon by scaling filtered local updates.

Experiments are under taken on MNIST, Fashion-MNIST, CIFAR-10, Breast Cancer Wisconsin, and Human Activity Recognition datasets. We compare the performance of different FL methods under different poisoned attacks. Results show that our method can effectively improve the robustness of FL. Furthermore, we also compare the computational cost of different FL methods. We can conclude that the computational overhead of VSRFL is close to the fastest FedAvg method.

The rest of this paper is organized as follows. Section II introduces the related work of federated learning and poisoned attacks. Section III describes robust federated learning via voting selection. Section IV presents the performance evaluation results for five datasets. Section V presents remarks and further work. The security analysis of VSRFL is available at https://github.com/lxy980304/VSRFL/blob/main/VSRFLSupplemental.pdf.

II. Related work

In this section, we describe federated learning, aggregation rules, poisoned attacks, and the properties of the attacker.

A. Federated learning

FL provides a decentralized approach [12], [13] to training models in a collaborative manner. It can train a unified deep learning (DL) model across multiple decentralized clients holding local data samples, coordinated by a server [14], [15]. Clients train models locally, so their data never has to leave the device [11], [16]. The server aggregates locally computed updates to obtain an improved version of the shared model [17]. FL embraces the ideas of centralized collection and data minimization, and it can help to avoid many of the systemic privacy problems that classic centralized DL techniques can bring [18], [19].

Suppose we have n clients and the i-th client has a local training dataset D_i. The client's goal is to use the local dataset to get the optimal update value for the global model. The best global model $\boldsymbol{w}^*$ is a answer to the following optimization problem: $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} F(\boldsymbol{w})$, where $F(\boldsymbol{w}) = \mathbb{E}_{D\sim\mathcal{X}}[f(D, \boldsymbol{w})]$ is the expectation of the empirical loss $f(D, \boldsymbol{w})$ on the joint training dataset D. In each epoch, federated learning conducts the following three steps:

Step 1: The global model parameters are encrypted and sent to the client by the server.

Step 2: The client replaces the local model $\boldsymbol{w}_i$ with the provided global model. The i-th client's new local model $\boldsymbol{w}_i^*$ is normally produced by solving the following optimization problem: $\boldsymbol{w}_i^* = \arg\min_{\boldsymbol{w}_i} F(D_i, \boldsymbol{w}_i)$. The client uploads local update $\boldsymbol{g}_i = \boldsymbol{w}_i^* - \boldsymbol{w}_i$ to server.

Step 3: The server produces a new global model by aggregating the gathered local modifications using suitable aggregation rules.

B. Aggregation rules

In federated learning, the aggregation rule is crucial, and the aggregation rules used by different FL methods are quite different. Aggregation rules generally fall into two categories. One is the non-robust aggregation rule, which is often used for FL in non-adversarial settings. Global models based on such rules generally have higher prediction accuracy. It is conditional that there are no abnormal clients in the environment. The other is robust aggregation rules, which have higher stability in adversarial environments. But it sacrifices part of the accuracy of the global model.

FederatedAveraging (FedAvg) [20] is a non-robust aggregation rule. The server sets the mean of the collected local updates as the gradient of the global model update. Unfortunately, FedAvg cannot handle exception updates efficiently. The global model can be perturbed when there is only one abnormal client in the environment [4].

The following are all robust aggregation rules:

Krum [21] updates the global model parameters with one of the local updates that has the smallest sum of distances from the other $(n - m)$ updates.

Trimmed mean (Trim-mean) [22] treats each local update parameter as an individual existence. First, the server sorts the j-th parameter of the collected n local updates, i.e., $\{g_{1j}, g_{2j}, \cdots, g_{nj}\}$, where g_{ij} is the j-th parameter of the i-th local update. Second, the server sets a pruning parameter $k < n/2$ to filter out the maximum k value and the minimum k value of each parameter. Finally, the mean of the filtered $n - 2k$ values constitutes a new local update for updating the global model. It is obvious that this aggregation rule cannot handle the situation where the number of abnormal updates exceeds 50%.

Median [22] is similar to Trim-mean. It also treats each locally updated parameter as a separate whole. The difference is that it replaces the mean in Trim-mean with the median. However, this aggregation rule is also unable to handle cases where the number of abnormal updates exceeds 50%.

FLTrust [4] is a FL aggregation rule based on trust conduction. This aggregation rule assumes that the server itself can collect clean small training datasets for learning tasks. Specifically, the server guides the direction of the global model update, and the client controls the magnitude of the global model update. First, the server trains a server update based on its local dataset. Second, the similarity between the local update and the server update is used as the weighting coefficient of the local update. Finally, the server updates the global model with the weighted mean of the local updates.

C. Poisoned attacks to federated learning

Data poisoned attacks [23]. This type of attack method usually tampers with the client's local dataset. Specifically, the attacker attempts to mix samples with modified features or modified labels into a clean local dataset to achieve the purpose of the attack, e.g., attacker adds specific triggers to image samples [7] or adds a no-parking label that modifies the speed limit to 80 km/h. Data poisoned attacks are easy to implement, but the attack effect is usually not obvious.

Model poisoned attacks [8]. Such methods usually directly control local updates uploaded by the client. Specifically, the attack maximizes the impact of the attack through randomly generated or scaled-generated local updates while evading server anomaly detection [24], e.g., Krum attacks maximize the deviation of local updates from the global model by simulating model aggregation; Median attack is to randomly generate local updates within a certain interval. Model poisoned attacks are more complex than the data poisoned attack, but its stealth is stronger.

D. Adversary goals and capabilities

For untargeted attackers, they aim to make the global model $\boldsymbol{w}$ indiscriminately misclassify the test dataset X_{test}, e.g., $f(\boldsymbol{w}', X_{test}) \neq f(\boldsymbol{w}, X_{test})$. For targeted attackers, they aim to make the global model $\boldsymbol{w}$ provide targeted incorrect predictions, e.g., $f(\boldsymbol{w}', X_{target}) = y_{target} \neq f(\boldsymbol{w}, X_{target})$, where X_{target} and y_{target} are the target dataset and target label selected by the attacker, respectively. In addition, targeted attackers maintain a global model to correctly classify samples that do not belong to the target dataset ($f(\boldsymbol{w}', X) = f(\boldsymbol{w}, X)$, where $X \nsubseteq X_{target}$) [6].

We assume that the attacker can freely control m clients. The number of controlled clients should be less than 30% of the total ($m/n \leq 30\%$). On the one hand, the attacker has full control over m client's training dataset, training process, and uploaded local update. On the other hand, the attacker cannot break the simulated aggregate on the server or alter local updates uploaded by other uncontrolled clients.

III. The Proposed Method

In this section, we first describe the differences between a poisoned update and a clean update. Second, we show two strategies for VSRFL. Finally, we summarize the whole process of VSRFL in federated learning. Furthermore, we demonstrate that the gap between the aggregated results of VSRFL and the optimal solution is bounded.

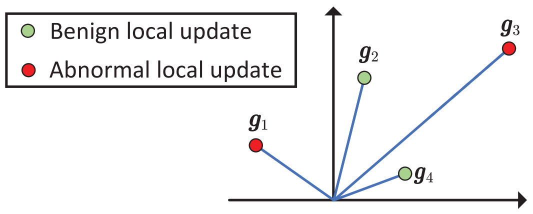


Fig. 2: Abstract two-dimensional representation of benign and abnormal local updates.

A. Problem statement

Client-uploaded local update abstraction. Since local updates are usually saved as high-latitude matrices, it is difficult to observe the difference between local updates. Therefore, we abstract the serialized local updates as vectors in two-dimensional space. Figure 2 shows local updates for different identities. We can see that there are two kinds of malicious updates.

One is a local update with an abnormal direction. Compared to benign local updates, such malicious updates have a large angle deviation from other updates (cf., $\boldsymbol{g}_1$ in Fig. 2). Attackers often obtain such malicious updates by operating the training process to increase the loss. Such malicious updates cause the global model to be updated in the opposite direction during model aggregation. Therefore, the accuracy of the global model on the test dataset will be reduced.

The other is local updates with an abnormal magnitude. Such malicious updates are less angularly biased than benign local updates. But their magnitudes are much larger than those of benign local updates (cf., $\boldsymbol{g}_3$ in Fig. 2). Attackers obtain such updates by amplifying the local updates of the controlled client. Such malicious updates can dominate the model aggregation process. Therefore, the attacker can make the global model achieve high accuracy on the backdoor task.

B. Problem solving method

To eliminate the threat of the above two kinds of malicious local updates to the global model, we propose two strategies as follows:

Voting mechanism: The local updates with large angular biases are dynamically filtered out through a voting mech-

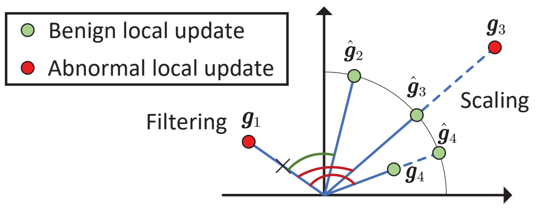
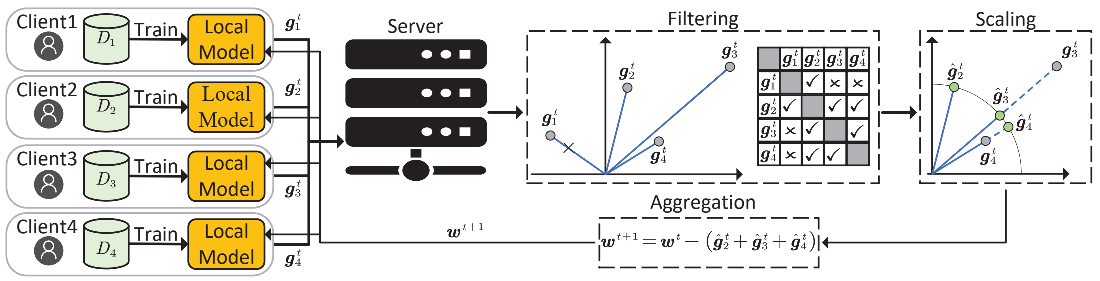


Fig. 3: Two strategies for dealing with different kinds of malicious updates. For local updates with abnormal directions, we filter them through the voting mechanism. For local updates with abnormal magnitudes, we adjust them through the scaling mechanism.



	g_1^t	g_2^t	g_3^t	g_4^t
g_1^t		✓	×	×
g_2^t	✓		✓	✓
g_3^t	×	✓		✓
g_4^t	×	✓	✓	

Fig. 4: Proposed structure for robust federated learning based on voting selection. The client performs local training after receiving the global model parameter $\boldsymbol{w}^t$. First, the server receives local updates uploaded by the clients. Second, the collected local updates are filtered and scaled to eliminate ones with anomalous direction and magnitude, respectively. Finally, the processed local updates are used to maintain the global model $\boldsymbol{w}^{t+1}$.

anism. As described in the previous subsection, there is a large vector angle between a well-prepared poisoned update and a clean local update. We propose a voting mechanism to dynamically filter local updates collected by the server at each epoch (illustrates in Fig. 3). Specifically, the server first treats a local update as a target. Second, the server computes the vector angle between the target and other local updates. Angles greater than 90° are considered negative votes, otherwise are considered positive votes. Finally, if the number of negative votes exceeds half of the total, the target is considered abnormal and cannot participate in model aggregation. Existing defense methods usually fix the number of filtered updates. This requires the server to obtain the maximum number of abnormal clients beforehand. However, our method is dynamic in the number of filtered updates at each epoch. That is, the server only needs to filter out local updates that have more than half of the total number of negative votes. We calculate the angle between two updates $\boldsymbol{g}_i$ and $\boldsymbol{g}_j$ based on the following formula:

$$v_{ij} = \frac{\sum_{k=1}^{p} \boldsymbol{g}_i^k \boldsymbol{g}_j^k}{\sqrt{\sum_{k=1}^{p} (\boldsymbol{g}_i^k)^2} \sqrt{\sum_{k=1}^{p} (\boldsymbol{g}_j^k)^2}}, \tag{1}$$

where k is the dimension of the serialization update. If $v_{ij} < 0$, the angle between $\boldsymbol{g}_i$ and $\boldsymbol{g}_j$ is greater than 90°, otherwise less than or equal to 90°.

Scaling mechanism: The local updates are scaled to average their impact on the global model. The attacker can also dominate the iterative direction of the global model by amplifying local updates. We propose a scaling mechanism to limit the impact of this anomaly. Specifically, the server first computes the filtered update median. Second, updates above the median will be scaled down, otherwise they will be scaled up. Finally, scaled local updates are used for server model aggregation. The scaling effect is shown in Fig. 3. The reason we choose to scale by the median is because the median is more representative than the mean. If mean scaling is used, $\boldsymbol{g}_2$ and $\boldsymbol{g}_4$ will be greatly enlarged due to their smaller magnitudes. The $\boldsymbol{g}_3$ will be slightly reduced due to the larger magnitude of $\boldsymbol{g}_3$. The local update $\boldsymbol{g}_i$ scaling formula is as follows:

$$\hat{\boldsymbol{g}}_i = \frac{\|\boldsymbol{g}_{median}\|}{\|\boldsymbol{g}_i\|} \times \boldsymbol{g}_i, \tag{2}$$

where i belongs to the filtered local update set $\mathcal{S}$.

Algorithm 1: VSRFL

```
Input: $n$, $T$
Output: $w^T$
Initialize $w^0$;
for $t \in \{1, \dots, T\}$ do
    Initialize list $\mathcal{B}$, $\mathcal{S}$;
    for $i \in \{0, \dots, n-1\}$ do
        $g_i^{t-1}$ = ClientLocalUpdate $(w^{t-1}, D_i)$;
    end
    //Step 1. Filter local updates;
    for $i \in \{0, \dots, n-1\}$ do
        Initialize the number of votes $\mathcal{O}, \mathcal{A} = 0$;
        for $j \in \{0, \dots, n-1\} \setminus \{i\}$ do
            Calculate $v_{ij}$ based on Eq. (1);
            if $v_{ij} < 0$ then
                $\mathcal{O}++$;
            else
                $\mathcal{A}++$;
            end
        end
        if $\mathcal{O} < \mathcal{A}$ then
            Add $g_i^{t-1}$ to list $\mathcal{B}$;
        end
    end
    //Step 2. Scaling local updates;
    Calculate the median value $g_{median}$ in list $\mathcal{B}$;
    for $g_i^{t-1} \in \mathcal{B}$ do
        Calculate $\hat{g}_i^{t-1}$ based on Eq. (2);
        Add $\hat{g}_i^{t-1}$ to list $\mathcal{S}$;
    end
    //Step 3. Update the global model;
    Calcute $w^t$ based on Eq. (3);
end
```

C. The framework of VSRFL

The VSRFL method is based on the above two strategies. Specifically, these two strategies can be summarized as: filtering and scaling. Figure 4 shows these components and the workflow of VSRFL during epoch t. First, the server receives local updates uploaded by clients. Second, the collected local updates are processed through filtering and scaling, respectively. Finally, the processed updates are used to maintain the global model as follows:

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^{t} - \frac{\alpha}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} \hat{\boldsymbol{g}}_i^t, \tag{3}$$

where α is the learning rate, $\mathcal{S}$ is the processed local update set, and $|\mathcal{S}|$ is the number of elements of $\mathcal{S}$. Algorithm 1 describes the entire process of VSRFL in one epoch.

IV. Experiments

In this section, we evaluate the VSRFL against existing FL poisoned attacks.

A. Datasets

We use the following five datasets for evaluation. **MNIST** [25] and **Fashion-MNIST (F-MNIST)** [26] are 10-class classification datasets with 60,000 training samples and 10,000 test samples, respectively. Each sample is a 28×28 picture. **CIFAR-10** [27] is a 10-class dataset with 50,000 training samples and 10,000 test samples, each of which is a 32×32 color image. **Breast Cancer Wisconsin (BCW)** [28] is a small sample binary classification dataset. The dataset has a total of 683 samples, and each sample has 9-dimensional feature values. We randomly selected 80% of the samples as training samples and the rest as test samples. **Human Activity Recognition (HAR)** [29] is a 6-class dataset and the task is to predict user behavior in 6 possible activities. The dataset has a total of 10,299 samples, and each sample has 561-dimensional feature values.

B. Machine learning classifiers

To demonstrate the applicability of our method, we use different models for classification tasks. Specifically, we use a convolutional neural network (CNN) to identify the MNIST and F-MNIST datasets. For the HAR and BCW datasets, we use a logistic regression (LR) classifier as the global model. Due to the complexity of the CIFAR-10 dataset, we chose the ResNet20 network as the classifier.

C. Evaluated poisoned attacks

We evaluate FL poisoned attacks based on different strategies. For data poisoned attacks [23], [30] that directly modify client datasets, we evaluate Label flipping and Scaling attacks. For model poisoned attacks [8] that directly modifying local updates uploaded by the client, we evaluate Gaussian, Krum, Trimmed mean, and Median attacks.

Gaussian attack. This method directly modifies local updates uploaded by the client. First, the attacker builds a gaussian distribution of locally updated parameters. Second, the attacker selects m values from the gaussian distribution as the j-th update parameter for m controlled clients. Repeat the above process until all parameters of the local update are fetched.

Label flipping attack (LF). This method modifies the labels of all local training samples. In detail, the attacker modifies the label I to label $(I+1)\%L$, where L is the total number of labels in the dataset and $I \in Y = \{1, 2, \ldots, L\}$.

Krum attack [8]. This method is an untargeted attack against Krum aggregation rule, and its purpose is to reduce the accuracy of the global model on the test dataset. The attacker tries to change the global model update by the maximum magnitude in the opposite direction of $\boldsymbol{g}$. However, excessive magnitude is usually discarded by the server. Therefore, the attacker needs to simulate model aggregation to choose an appropriate magnitude.

Trimmed mean attack (Trim) [8]. This is a local model poisoned attack against Trim-mean aggregation rule, with the goal of lowering the global model's accuracy on the test dataset. The attacker first computes the mean μ_j and standard deviation ξ_j of the j-th parameter in the local update. When s_j is estimated to be -1, the attacker samples m numbers from the interval $[\mu_j + 3\xi_j, \mu_j + 4\xi_j]$ as the j-th parameter of the m controlled client. When s_j is estimated to be 1, the attacker samples m numbers from the interval $[\mu_j - 4\xi_j, \mu_j - 3\xi_j]$ as the j-th parameter of the m controlled client. s_j is the change direction of the j-th global model parameter for the clean local update.

Median attack [8]. This method is a local model poisoned attack against Median aggregation rule, which aims to reduce the accuracy of the global model on the test dataset. When s_j is estimated to be 1, the attacker samples m numbers from the interval $[\boldsymbol{g}_{min,j}/b, \boldsymbol{g}_{min,j}]$ as the j-th parameter of m controlled clients. When s_j is estimated to be -1, the attacker samples m numbers from the interval $[\boldsymbol{g}_{max,j}, b \times \boldsymbol{g}_{max,j}]$ as the j-th parameter of m controlled clients. $\boldsymbol{g}_{max,j} = \max(\boldsymbol{g}_{1j}, \boldsymbol{g}_{2j}, \ldots, \boldsymbol{g}_{mj})$ and $\boldsymbol{g}_{min,j} = \min(\boldsymbol{g}_{1j}, \boldsymbol{g}_{2j}, \ldots, \boldsymbol{g}_{mj})$, where $\boldsymbol{g}_{ij}$ is the parameter of the j-th clean local update of the i-th controlled client, s_j is the change direction of the j-th global model parameter for the clean local update, and $b = 2$.

Scaling attack [31]. This method is a local data poisoned attack against FL. Its purpose is to misclassify target samples into target labels, rather than reducing the accuracy of the global model on the test dataset.

D. Evaluation metrics

For untargeted poisoned attacks such as Gaussian, LF, Krum, Trim, Trim and Median attacks, we take the accuracy rate of the global model on the test dataset as the evaluation metric (call this metric as the main task rate), because these attacks aim to indiscriminately reduce the accuracy of the global model. If the main task rate of the FL method in the presence of an attack remains close to the accuracy without an attack, then the FL method is robust. For scaling attack, we not only focus on the accuracy of the global model, but also the probability of non-target label samples being misclassified as the target label (call this metric as the poisoned task rate). This is because targeted attacks usually maintain the accuracy of the global model to avoid the attack being detected. Targeted attacks typically cause the global model to predict the attacker-chosen target labels for the attacker-chosen target test samples. If the poisoned task rate of the FL method in the presence of an attack is still close to that without an attack, it means that the FL method is resistant to backdoor attacks.

E. Federated learning settings

We set different degrees of non-IID training data distributions to simulate a real federated learning environment. Specifically, we assume that the dataset is an L classification problem, e.g., $L = 10$ for the MNIST and F-MNIST datasets. First, we divide the clients into L groups equally. Then, training samples with label I are assigned to group I with probability $q > 0$ or other groups with probability $(1-q)/(L-1)$. Finally, these training samples from each group are divided equally among the clients within the group. The larger the value of q, the higher the degree of non-IID distribution. We set $q = 0.5$ in our experiments.

TABLE I: The main task rates of different FL methods.

(a) CNN global model, MNIST

	FedAvg	Krum	Trim-mean	Median	FlTrust	VSRFL
No attack	0.96	0.91	0.90	0.90	0.82	0.87
Gaussian attack	0.91	0.82	0.89	0.91	0.86	0.87
LF attack	0.57	0.76	0.86	0.86	0.64	0.83
Krum attack	0.88	0.82	0.87	0.85	0.77	0.85
Trim attack	0.29	0.85	0.83	0.85	0.81	0.85
Median attack	0.83	0.86	0.83	0.85	0.86	0.85
Scaling attack	0.88/0.05	0.85/0.01	0.89/0.03	0.89/0.03	0.87/0.01	0.88/0.01

(b) CNN global model, F-MNIST

	FedAvg	Krum	Trim-mean	Median	FlTrust	VSRFL
No attack	0.90	0.82	0.90	0.89	0.87	0.89
Gaussian attack	0.89	0.83	0.89	0.89	0.85	0.89
LF attack	0.59	0.85	0.85	0.87	0.77	0.83
Krum attack	0.87	0.80	0.87	0.84	0.77	0.86
Trim attack	0.89	0.81	0.83	0.86	0.85	0.87
Median attack	0.83	0.83	0.83	0.86	0.81	0.86
Scaling attack	0.89/0.04	0.81/0.01	0.90/0.01	0.90/0.01	0.87/0.01	0.88/0.01

(c) ResNet20 global model, CIFAR-10

	FedAvg	Krum	Trim-mean	Median	FlTrust	VSRFL
No attack	0.67	0.23	0.68	0.68	0.56	0.62
Gaussian attack	0.67	0.51	0.68	0.67	0.52	0.61
LF attack	0.61	0.19	0.62	0.59	0.51	0.56
Krum attack	0.48	0.10	0.30	0.37	0.39	0.58
Trim attack	0.17	0.20	0.36	0.41	0.54	0.61
Median attack	0.20	0.35	0.38	0.43	0.52	0.62
Scaling attack	0.56/0.15	0.26/0.21	0.59/0.08	0.59/0.08	0.53/0.10	0.56/0.08

(d) LR global model, BCW

	FedAvg	Krum	Trim-mean	Median	FlTrust	VSRFL
No attack	0.97	0.96	0.95	0.93	0.93	0.99
Gaussian attack	0.95	0.95	0.97	0.95	0.90	0.96
LF attack	0.97	0.94	0.96	0.96	0.93	0.97
Krum attack	0.98	0.95	0.95	0.97	0.97	0.97
Trim attack	0.97	0.93	0.98	0.93	0.97	0.99
Median attack	0.97	0.92	0.95	0.97	0.96	0.94
Scaling attack	0.93/0.11	0.88/0.20	0.96/0.03	0.96/0.03	0.97/0.08	0.97/0.06

(e) LR global model, HAR

	FedAvg	Krum	Trim-mean	Median	FlTrust	VSRFL
No attack	0.85	0.77	0.84	0.84	0.82	0.84
Gaussian attack	0.86	0.83	0.87	0.86	0.82	0.84
LF attack	0.12	0.58	0.87	0.75	0.81	0.77
Krum attack	0.86	0.77	0.87	0.85	0.83	0.82
Trim attack	0.55	0.55	0.79	0.84	0.79	0.77
Median attack	0.83	0.44	0.85	0.85	0.79	0.82
Scaling attack	0.76/0.17	0.69/0.27	0.82/0.04	0.86/0.05	0.80/0.09	0.82/0.07

TABLE II: The aggregate unit time (s) under different FL methods and different datasets. The results represent the aggregation time of different FL methods under different dataset units epoch.

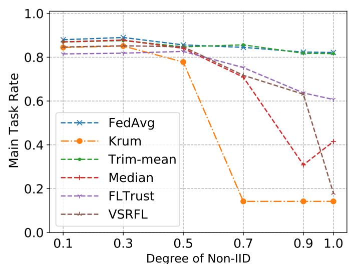
	FedAvg	Krum	Trim-mean	Median	FlTrust	VSRFL
MNIST	0.2898	0.7946	18.085	14.998	0.9925	0.1350
F-MNIST	0.0776	0.3504	19.862	18.237	1.4431	0.1694
CIFAR-10	0.5591	4.7390	300.50	339.95	2.1696	2.2957
BCW	0.0015	0.0045	3.8210	4.0460	0.1534	0.0039
HAR	0.0019	0.0174	4.0813	4.2324	0.1719	0.0124

F. Experimental results

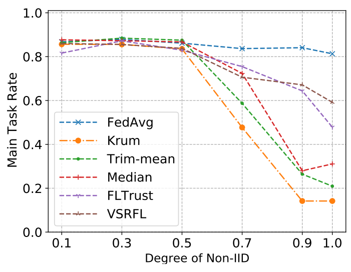
Our aim is to design a FL aggregation method that aims to enable the server to train an accurate global model when there are abnormal clients in FL. Specifically, our method achieves the following two defense goals:

Robustness. FL methods should not sacrifice the accuracy of the global model in different environments. Table I shows the main task rate of different FL methods on different datasets, where the result of Scaling attack is “main task rate/poisoned task rate”. On the one hand, in the absence of abnormal clients, our method should be able to learn a model with the same accuracy as the non-adversarial FedAvg. For instance, on F-MNIST, the main task rates of VSRFL and Median are only 0.01 different from FedAvg, while Krum’s main task rate is as high as 0.08 compared to FedAvg. The reason is that our method considers all filtered local updates, while Krum only selects the local updates with the highest confidence to maintain the global model. On the other hand, our method should keep the accuracy of the global model from fluctuating too much when there are abnormal clients. Specifically, the difference between the main task rate of VSRFL and the FedAvg under the no-attack environment is at most 0.09. However, other methods have higher main task rate differences compared to FedAvg. In addition, for backdoor Scaling attacks, VSRFL can also reduce its success rate.

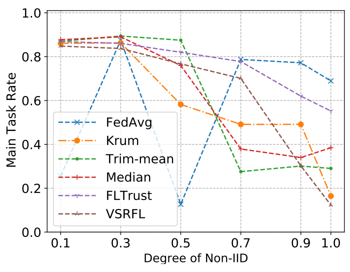
Efficiency. The FL method should not incur additional computational overhead. The clients of FL are usually served by edge devices with limited computing power. If the server requires a lot of computation to aggregate the model, this requires the server and client to maintain long-term communication. This is obviously not feasible. Therefore, FL methods should redu ce the computational overhead of the server aggregation model. Table II shows the aggregated unit time of different FL methods on different datasets. Results show that our method is similar to FedAvg in server aggregation time per epoch. This means that our method does not impose additional computational overhead on the server compared to the unattacked FedAvg. In addition, we also find that the computational overhead of Mean and Median raises with the



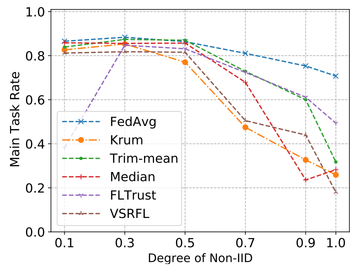
(a) No attack



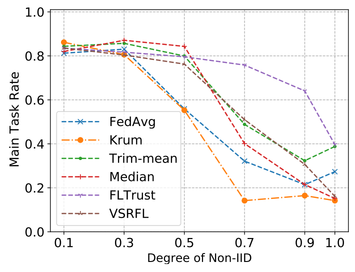
(b) Gaussian attack



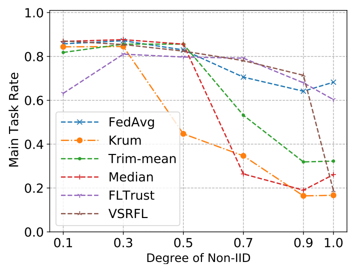
(c) LF attack



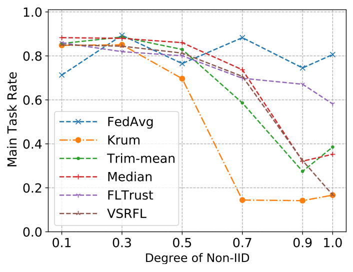
(d) Krum attack



(e) Trim attack



(f) Median attack



(g) Scaling attack

Fig. 5: Main task rates under various degrees of non-IID on HAR.

increase of model parameters. The reason is because Mean and Median consider each parameter of the model.

We also conducted experiments on the degree of non-IID and the total number of clients. The experimental results are as follows:

Impacts of the degree of non-IID: Figure 5 shows the main task rates of different FL methods under different attacks when the non-IID degree increases from 0.1 to 1.0. When there are no abnormal client in the environment, FedAvg and Trim-mean can make the global model converge at different non-IID degrees. The global model based on the remaining four FL methods will be difficult to converge with the increase of the Non-IID degree, especially the Krum method. When there are abnormal clients in the environment, the global model based on VSRFL will be difficult to converge with the increase of Non-IID degree. Specifically, the global model aggregated by VSRFL is difficult to converge when each client's local dataset has only one class. The reason is that VSRFL is based on a voting mechanism to filter local updates with abnormal directions. However, the direction difference between local updates trained on datasets with high Non-IID degree is large. This makes VSRFL unable to effectively filter local updates with abnormal directions.

Impacts of total number of clients: Figure 6 shows the main task rate of different FL methods under different attacks when the total number of clients increases from 40 to 200. When there are no abnormal clients in the environment, the impact of the number of clients on the global model of VSRFL aggregates can be ignored. When there are abnormal clients in the environment, our method can basically eliminate the impact of abnormal updates on the global model. In addition, Krum, Trim and Median attacks have a greater impact on the global model aggregated by Krum, Trim-mean and Median methods, respectively. The reason is that these three model poisoned methods are specially proposed for these three FL methods.

V. Conclusion

In this paper, we propose a robust federated learning based on voting selection. Malicious local updates are usually divided into abnormal directions and abnormal magnitudes. The voting mechanism is designed to filter local updates with abnormal directions. The scaling mechanism is desigened to adjust local updates with abnormal magnitudes. Our proposed method can adaptively filter abnormal updates without requiring the server to obtain the maximum number of outlier clients.

In addition, we also found such a phenomenon in the experiment. If local datasets have widely different sample distributions, then the updates trained on these datasets will make it difficult for the global model to converge. In the future, we will explore how to reduce the impact of this difference on the global model.

VI. Acknowledgements

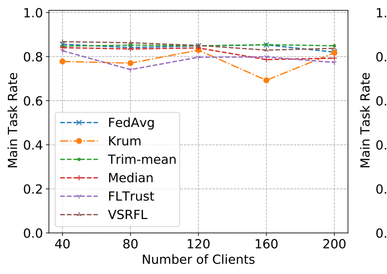
This work is supported by the National Natural Science Foundation of China (61902328), the Applied Basic Research Project of Science and Technology Bureau of Nanchong City (SXHZ040), and Central Government Funds of Guiding Local Scientific and Technological Development (2021ZYD0003).

References

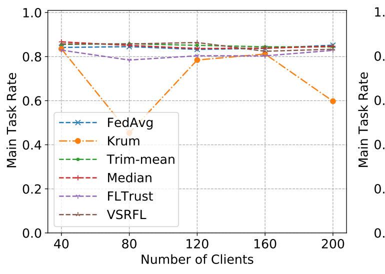
[1] Z. Zheng, Y. Zhou, Y. Sun, Z. Wang, B. Liu, and K. Li, "Applications of federated learning in smart cities: recent advances, taxonomy, and open challenges," *Connection Science*, 2022.

[2] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, 2016.

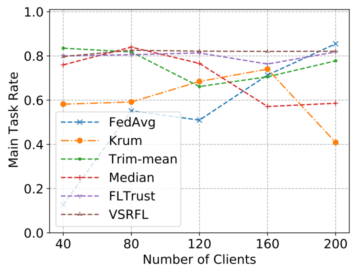
[3] M. Hao, H. Li, G. Xu, H. Chen, and T. Zhang, "Efficient, private and robust federated learning," in *Annual Computer Security Applications Conference*, 2021.



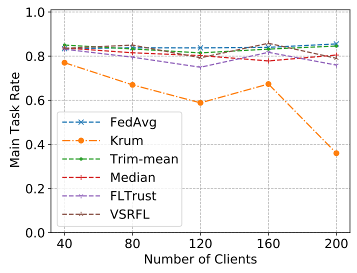
(a) No attack



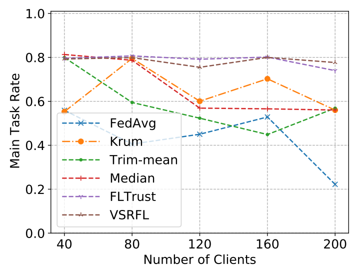
(b) Gaussian attack



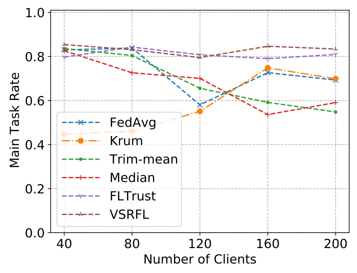
(c) LF attack



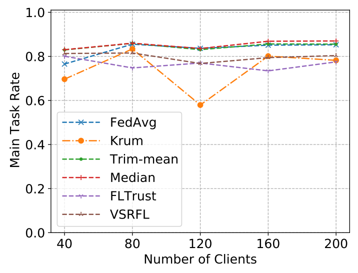
(d) Krum attack



(e) Trim attack



(f) Median attack



(g) Scaling attack

Fig. 6: Main task rates under different numbers of client on HAR.

[4] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," in *ISOC Network and Distributed System Security Symposium*, 2021.

[5] B. Ghimire and D. B. Rawat, "Recent advances on federated learning for cybersecurity and cybersecurity for federated learning for internet of things," *IEEE Internet of Things Journal*, 2022.

[6] T. D. Nguyen, P. Rieger, H. Chen, H. Yalame, H. Möllering, H. Fereidooni, S. Marchal, M. Miettinen, A. Mirhoseini, S. Zeitouni *et al.*, "Flame: Taming backdoors in federated learning," in *USENIX Security Symposium*, 2022.

[7] C. Xie, K. Huang, P.-Y. Chen, and B. Li, "Dba: Distributed backdoor attacks against federated learning," in *International Conference on Learning Representations*, 2019.

[8] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to byzantine-robust federated learning," in *USENIX Security Symposium*, 2020.

[9] Y. Khazbak, T. Tan, and G. Cao, "Mlguard: Mitigating poisoning attacks in privacy preserving distributed collaborative learning," in *International Conference on Computer Communications and Networks*, 2020.

[10] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "Verifynet: Secure and verifiable federated learning," *IEEE Transactions on Information Forensics and Security*, 2019.

[11] S. Andreina, G. A. Marson, H. Möllering, and G. Karame, "Baffle: Backdoor detection via feedback-based federated learning," in *IEEE International Conference on Distributed Computing Systems*, 2021.

[12] C. Fung, C. J. Yoon, and I. Beschastnikh, "The limitations of federated learning in sybil settings," in *International Symposium on Research in Attacks, Intrusions and Defenses*, 2020.

[13] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, B. McMahan, T. Van Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards federated learning at scale: System design," in *Proceedings of Machine Learning and Systems*, 2019.

[14] G. Liu, X. Ma, Y. Yang, C. Wang, and J. Liu, "Federaser: Enabling efficient client-level data removal from federated learning models," in *IEEE/ACM International Symposium on Quality of Service*, 2021.

[15] C. P. Wan and Q. Chen, "Robust federated learning with attack-adaptive aggregation," *arXiv preprint arXiv:2102.05257*, 2021.

[16] S. Li, Y. Cheng, W. Wang, Y. Liu, and T. Chen, "Learning to detect malicious clients for robust federated learning," *arXiv preprint arXiv:2002.00211*, 2020.

[17] C. Wu, S. Zhu, and P. Mitra, "Federated unlearning with knowledge distillation," *arXiv preprint arXiv:2201.09441*, 2022.

[18] J. Gao, B. Zhang, X. Guo, T. Baker, M. Li, and Z. Liu, "Secure partial aggregation: Making federatedlearning more robust for industry 4.0 applications," *IEEE Transactions on Industrial Informatics*, 2022.

[19] Y. Mi, J. Guan, and S. Zhou, "Ariba: Towards accurate and robust identification of backdoor attacks in federated learning," *arXiv preprint arXiv:2202.04311*, 2022.

[20] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2017.

[21] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, 2017.

[22] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proceedings of the International Conference on Machine Learning*, 2018.

[23] T. D. Nguyen, P. Rieger, M. Miettinen, and A.-R. Sadeghi, "Poisoning attacks on federated learning-based iot intrusion detection system," in *NDSS Workshop on Decentralized IoT Systems and Security*, 2020.

[24] H. Wang, K. Sreenivasan, S. Rajput, H. Vishwakarma, S. Agarwal, J.-y. Sohn, K. Lee, and D. Papailiopoulos, "Attack of the tails: Yes, you really can backdoor federated learning," *Advances in Neural Information Processing Systems*, 2020.

[25] L. Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," *IEEE signal processing magazine*, 2012.

[26] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[27] A. Krizhevsky *et al.*, "Learning multiple layers of features from tiny images," *Master's Thesis, University of Tront*, 2009.

[28] A. Asuncion and D. Newman, "Uci machine learning repository," 2017.

[29] D. Anguita, A. Ghio, L. Oneto, X. Parra Perez, and J. L. Reyes Ortiz, "A public domain dataset for human activity recognition using smartphones," in *Proceedings of the International European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2013.

[30] S. Shen, S. Tople, and P. Saxena, "Auror: Defending against poisoning attacks in collaborative deep learning systems," in *Proceedings of the Annual Conference on Computer Security Applications*, 2016.

[31] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2020.