Fed-Credit: Robust Federated Learning with Credibility Management

JIAYAN CHEN[*], Beijing Normal University at Zhuhai, China

ZHIRONG QIAN[*], Beijing Normal University at Zhuhai, China

TIANHUI MENG[†], Beijing Normal University at Zhuhai, China

XITONG GAO, Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, China

TIAN WANG, Beijing Normal University at Zhuhai and Guangdong Key Lab of AI and Multi-Modal Data Processing, BNU-HKBU United International College, China

WEIJIA JIA, Beijing Normal University at Zhuhai and Guangdong Key Lab of AI and Multi-Modal Data Processing, BNU-HKBU United International College, China

Additional Key Words and Phrases: Privacy preservation, Federated learning, Credibility management, Poisoning attacks

Abstract

Aiming at privacy preservation, Federated Learning (FL) is an emerging machine learning approach enabling model training on decentralized devices or data sources. The learning mechanism of FL relies on aggregating parameter updates from individual clients. However, this process may pose a potential security risk due to the presence of malicious devices. Existing solutions are either costly due to the use of compute-intensive technology, or restrictive for reasons of strong assumptions such as the prior knowledge of the number of attackers and how they attack. Few methods consider both privacy constraints and uncertain attack scenarios. In this paper, we propose a robust FL approach based on the credibility management scheme, called Fed-Credit. Unlike previous studies, our approach does not require prior knowledge of the nodes and the data distribution. It maintains and employs a credibility set, which weighs the historical clients' contributions based on the similarity between the local models and global model, to adjust the global model update. The subtlety of Fed-Credit is that the time decay and attitudinal value factor are incorporated into the dynamic adjustment of the reputation weights and it boasts a computational complexity of $O(n)$ (n is the number of the clients). We conducted extensive experiments on the MNIST and CIFAR-10 datasets under 5 types of attacks. The results exhibit superior accuracy and resilience against adversarial attacks, all while maintaining comparatively low computational complexity. Among these, on the Non-IID CIFAR-10 dataset, our algorithm exhibited performance enhancements of 19.5% and 14.5%, respectively, in comparison to the state-of-the-art algorithm when dealing with two types of data poisoning attacks.

1 INTRODUCTION

The constantly increasing amount of data and the intricacy of machine learning models have given rise to an augmented requirement for computational resources. The current machine learning paradigm necessitates data collection in a central server, which may be unfeasible or undesirable from the perspectives of privacy, security, regulation, or economics. Recently, federated learning (FL) [1] and other decentralized machine learning methods [2, 3] have been

[*]Both authors contributed equally to this research.
[†]Corresponding author.

Authors' addresses: Jiayan Chen, ryan-utopia@outlook.com, Beijing Normal University at Zhuhai, No.18, Jinfeng Road, Zhuhai, Guangdong, China, 519087; Zhirong Qian, zr.qian@outlook.com, Beijing Normal University at Zhuhai, No.18, Jinfeng Road, Zhuhai, Guangdong, China, 519087; Tianhui Meng, tmeng@bnu.edu.cn, Beijing Normal University at Zhuhai, No.18, Jinfeng Road, Zhuhai, Guangdong, China, 519087; Xitong Gao, xt.gao@siat.ac.cn, Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, Guangdong, China; Tian Wang, @bnu.edu.cn, Beijing Normal University at Zhuhai and Guangdong Key Lab of AI and Multi-Modal Data Processing, BNU-HKBU United International College, No.18, Jinfeng Road, Zhuhai, Guangdong, China, 519087; Weijia Jia, @bnu.edu.cn, Beijing Normal University at Zhuhai and Guangdong Key Lab of AI and Multi-Modal Data Processing, BNU-HKBU United International College, No.18, Jinfeng Road, Zhuhai, Guangdong, China, 519087.

proposed as potential solutions to address the above issues. In a federated learning framework, the server broadcasts the shared global model to clients, clients then perform training with their private data sets and upload the updated models to the server. Then the server updates the global model by aggregating the locally updated models and begins the next round of training. FedAvg [4], which takes the average of local parameters as global parameters, is a typical aggregation algorithm.

However, due to its special framework, the typical FL algorithm faces some serious security threats to the model if some clients are malicious. Byzantine failure is one most important security threats, where some clients are malicious and take measures to attack the global model. For example, malicious clients could upload modified parameters while genuine clients upload local parameters. Then, the global model performance would be degraded even though only one single malicious client in FedAvg algorithm [5]. The untargeted attack [6, 7] is a model poisoning attacks, where malicious clients directly manipulate their local updates to compromise the global model performance. Shafah *et al.* [8] introduced a form of clean-label poisoning attack, which adds carefully crafted perturbations to a subset of the training data, to contaminate an image classification model. This contamination aims to disrupt the model training process, leading to training failure. [9–13] conducted investigations into a backdoor attack strategy named label flipping. In this attack, the attacker's model update is engineered to deliberately induce the local model to learn an incorrect mapping for a small subset of the data. As an example, during training on MNIST, the attackers may aim to make the model to classify all images originally labeled "7" as "1", thereby obstructing the convergence of the global model.

To address the above issue, previous research has proposed some robust aggregation rules to defend against such attacks. Krum and Multi-Krum [14], Median and Trimmed-mean [15], GeoMed [16], Bulyan [17], MAB-RFL [18] aim to drop potential malicious updates by comparing local updates. While [19–22] distinguish malicious clients by comparing the local updates and server update, which is trained using clean data stored on the server. Hsieh *et al.* [23] discovered that the Non-IID (not identically and independently distributed) distribution can notably stymie model training convergence in various distributed algorithms, including FedAvg, yet previous work ignores the effects of such heterogeneous data distributions. In the real-world FL system implementation, we neither know how the attacker is attacking, nor how many attackers. And for privacy reasons, it could be difficult to curate a global dataset sampled from the underlying data distribution. An urgent need thus arises to propose an aggregation rule that does not require such prior knowledge and can still effectively defend against multiple attacks.

In this paper, we propose a new Robust FL method called Fed-Credit. The server maintains a credibility set of clients based on cosine similarity, which makes the server consider the historical contribution of clients when assigning weights. We performed a comprehensive series of experiments on the MNIST and CIFAR-10 datasets, aiming to compare the performance of our Fed-Credit with other existing algorithms. Our assessment encompassed multiple attack types, varying fractions of malicious clients and dataset distributions. The empirical findings unequivocally showcase that our algorithm not only maintains high test accuracies but also demonstrates exceptional robustness against adversarial attacks. Importantly, these achievements are coupled with the benefit of maintaining a comparatively low computational complexity when contrasted with alternative algorithms.

The main contributions of this work can be summarized as follows:

- **Fed-Credit Scheme:** In general, this paper proposes a new framework for robust federated learning based on credibility values called Fed-Credit to address the challenges faced above. It maintains and employs a credibility set, which weighs the historical clients' contributions based on the similarity between the local models and global model, to adjust the global model update. The elegance of Fed-Credit lies in its inherent ability to dynamically adjust the weights of credibility values, all while maintaining a commendable computational complexity of $O(n)$. Notably, this operation is achieved without requiring the server to possess a training dataset.

- **Credibility Management for FL:** In this work, we intricately devised a management approach for credibility values within Fed-Credit. This is achieved through initialization and during the training process by comparing the similarity between the weights of global and local models and incorporating a temporal decay function to update the reputation values of each client. We designed a method that is characterized by its low computational complexity, while also being capable of efficiently training a global model without the necessity of prior knowledge regarding the credibility of individual clients.
- **Detailed Evaluation:** We conducted extensive comparative experiments between Fed-Credit and various algorithms proposed in prior research. These experiments encompassed diverse datasets, disparate distributions, varying attack methods, and differing numbers of attackers. The results of these experiments demonstrate that our algorithm attains or surpasses the state-of-the-art algorithms in almost all scenarios. Notably, on the Non-IID CIFAR-10 dataset, our algorithm exhibited performance enhancements of 19.5% and 14.5%, respectively, in comparison to the state-of-the-art algorithm when dealing with two types of data poisoning attacks as evaluated in our experiments involving four attackers. We also tracked the dynamic changes in credibility values among different clients during the training process. The outcomes indicate that Fed-Credit effectively distinguishes attackers within the client population.

The remainder of this paper is structured as follows. The system model, threat model and defense model are presented in Section 2, while Section 3 provides details of our proposed solution. Experimental results are showcased in Section 4. The related work and motivation are introduced in Section 5. Finally, Section 6 concludes the paper.

2 FL SYSTEM

In this section, we present the system model, threat model, and defense model of the Federated Learning (FL) system considered in this work. Figure 1 provides an overview of the system.

2.1 System Model

We consider a Federated Learning (FL) system that consists of multiple clients and one central server for collaborative model training. The server forms a global model by aggregating model parameters uploaded by clients. However, some of these clients may be compromised and controlled by malicious attackers, turning them into adversarial clients Figure 1. Therefore, our objective is to find an effective method to aggregate the model parameters provided by both the potentially adversarial and the regular clients, while maintaining the efficacy of the global model. Further details are provided below.

Client: In an FL system, the clients refer to the individual participants or devices that contribute to the collaborative model training. Each client has its own local dataset and model, which it uses to train the model. Before each training session, clients will receive the latest global model from the server. They then use their own local datasets to train respective models based on this global model. Finally, the trained models will be uploaded to the server.

Server: The server in this system maintains the credibility values of each client and the global model. At the start of each training round, the server distributes the current global model to all clients. Once the clients complete the predetermined number of training epochs, the server collects the models trained by each client and aggregates them

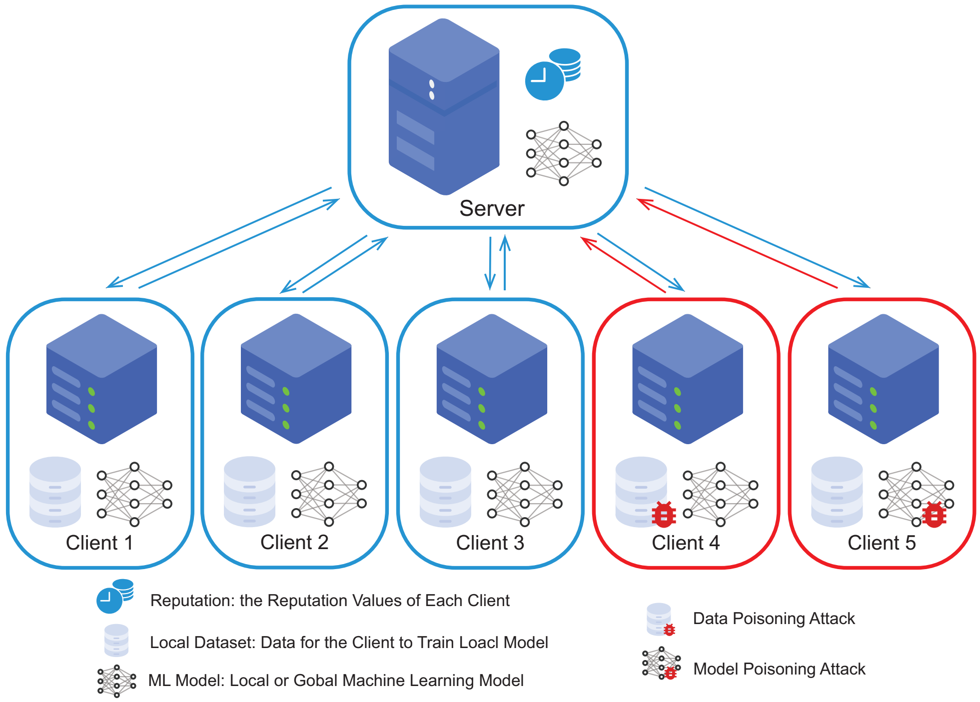

Fig. 1. FL system framework

into a new global model using a credibility value-based algorithm. Subsequently, the server updates the credibility values of different clients based on the new global model and the local models submitted by the clients. Finally, the server sends the updated global model back to the clients for the subsequent training round. This process repeats until reaching the desired accuracy or the maximum number of training rounds.

2.2 Threat Model

In this section, we present a comprehensive threat model for poisoning attacks in Federated Learning (FL) system (Figure 1). This model encompasses various aspects such as the objectives of poisoning attacks, the types of poisoning attacks, the knowledge possessed by the poisoning attacker, and the assumptions made for poisoning attacks.

Objective of Poisoning Attacker: The attacker typically compromises the learning system to cause failure on specific inputs intentionally chosen by the attacker. This process can even construct backdoors through which they can control the output of a deployed model in the future. Similar to numerous prior studies on poisoning attacks [24–26], we assume that the primary objective of poisoning attacker in FL system is to deliberately manipulate the local training process and compromise the aggregation process of the global model. The aim is to cause a significant increase in error rates for the testing data, thereby undermining the integrity and reliability of the model.

Types of Poisoning Attacks: Poisoning attacks in FL systems can be categorized based on their attack methods, mainly including data poisoning attacks [27] and model poisoning attacks [25]. Data poisoning attacks involve injecting poisoned data samples into the training dataset, such as label-flipping attacks [27]. In this paper, we apply label-flipping attacks using the pairwise [11] and symmetric [10] matrix methods. Model poisoning attacks aim to target the model parameters directly, manipulating the aggregation process by sending error or noisy parameters. Model poisoning attacks [24, 28, 29] aim to thwart the FL process by uploading to the server either constant model weights, or weights sampled from a certain distribution, or parameters that are opposite to the training results.

Poisoning Attacker's Knowledge: As poisoning attackers are part of the FL system, they possess a certain level of knowledge about the FL system and its components. This knowledge includes an understanding of the training data distribution, the model architecture, the learning algorithm, and the global model parameters that are updated during the iterative process in communication rounds. However, conducting data and model poisoning attacks does not necessarily depend on this knowledge but rather on the attackers' ability to collaborate with each other. For instance, to execute a model poisoning attack, several attackers may need to provide identical constants or distributions while returning locally trained parameters.

Poisoning Attacks Assumptions: Poisoning attacks in FL system are built upon certain assumptions. *(1)* Multiple poisoning attackers can assume the ability to collaborate. This collaboration can involve the use of different strategies by different attackers, such as pairwise and symmetric matrix label flipping attacks, or utilize the same model poisoning attacks. *(2)* It is assumed that the number of malicious clients does not exceed half of the total [24]. *(3)* We finally assume that the communication between the server and client is reliable, which means that in our paper, we do not consider noise and errors caused during the transmission process.

2.3 Defense Model

Defender's knowledge: The defense is performed on the server. In practical applications, the server has several limitations. *(1)* The server does not possess any training data on the server side. *(2)* The server cannot access the data stored in clients. *(3)* The server does not know the number of malicious clients, and their attack strategy choices. *(4)* The server only has access to local model updates.

In this work, we consider the general FL framework, which consists of a server and n clients, and f (where $f < \frac{n}{2}$) of them are malicious clients. The server has R synchronous rounds and clients cluster C has E local epochs in each round. During each round, the server broadcasts global parameters g to the clients. The clients subsequently train the model using a mini-batch size of B with their respective local datasets for E epochs to obtain local updated model g_i. Among the n clients, f malicious clients perform attacks by poisoning datasets (data poisoning) or manipulating local models (model poisoning). The server collects local updates and computes a new global model by applying an aggregation rule. Detailed definitions of all symbols are given in Table 1.

3 FED-CREDIT

FL algorithms, for instance, FedAvg, typically aggregates the local model updates from all clients by averaging the local model updates to update the global model. The weight of each client is usually set to be equal to the size of its local dataset. In the presence of malicious clients, the global model is vulnerable to attacks; and the server faces challenges in accurately determining whether the clients' data distributions exhibit natural heterogeneity or are intentionally manipulated. This difficulty arises from the server's inability to sample directly from the clients' datasets. It is thus preferred to evaluate the credibility of clients as each client's weight, thus reducing the weights of malicious clients to protect global model training.

To address this issue, we propose Fed-Credit, a novel defense framework based on credibility management. It aims to resist various types of attacks without the knowledge of the number of malicious clients. The server considers the

Table 1. Table of Notations.

Parameter	Description
$\alpha_1, \alpha_2, \beta$	Hyperparameters of Fed-Credit
α	The equilibrium factor
n	Number of clients
f	Number of malicious clients ($0 \leq f < n/2$)
η	Learning rate
B	Batch Size
C_i	The i^{th} client ($1 \leq i \leq n$)
$\boldsymbol{\theta}$	The global model
$\boldsymbol{\theta}_i$	The local model of C_i
$\boldsymbol{\tau}$	The credibility set of clients
τ_i	The credibility of C_i
S_i	The credibility score between local model of C_i and the global model
w_i	The weight of C_i
R	Number of global training rounds
E	Number of local training epochs

cosine similarity of each client's local update with the global update and the credibility value of the previous rounds to update the client's credibility. Our approach envisions the server dynamically managing a set of credibility values for each client. These values assess historical contributions by considering the similarity between local models and the global model. Incorporating temporal decay and credibility values, the server judiciously adjusts the weights assigned to each client's local updates for global model aggregation. The overall algorithm of our proposed Fed-Credit method can be found in Algorithm 1, and it is summarized in the following five steps:

- **Step 1:** The server initializes the global model $\boldsymbol{\theta}$ and assigns an initial credibility value of 1 to each client. Next, the server iterates Steps 2 to 5 until either the global model $\boldsymbol{\theta}$ achieves the desired performance or reaches the maximum allowable number of global training epochs R.
- **Step 2:** The server sends the global model $\boldsymbol{\theta}$ to all clients.
- **Step 3:** The clients C_i independently train models using their own local datasets. There is no communication between benign clients, ensuring that they cannot exchange the datasets or trained models with other clients. At the end of the training process, the clients upload the model parameters $\boldsymbol{\theta}_i$ to the server.
- **Step 4:** The server incorporates an equilibrium factor α to dynamically regulate the impact of the credibility value τ_i on the client weights w_i. α is obtained in Algorithm 3. As training progresses, the influence of the credibility value τ_i on weights w_i gradually increases. The server then aggregates the local updates using the client weights w_i.
- **Step 5:** The server assesses the cosine similarity between individual layers of each local model $\boldsymbol{\theta}_i$ and the global models to obtain the credibility score S_i of $localmodel$. Fed-Credit takes the decaying effect over time into consideration, thus an exponential decay factor is utilized to average historical credibility values. Finally, the server normalizes the credibility values of each client.

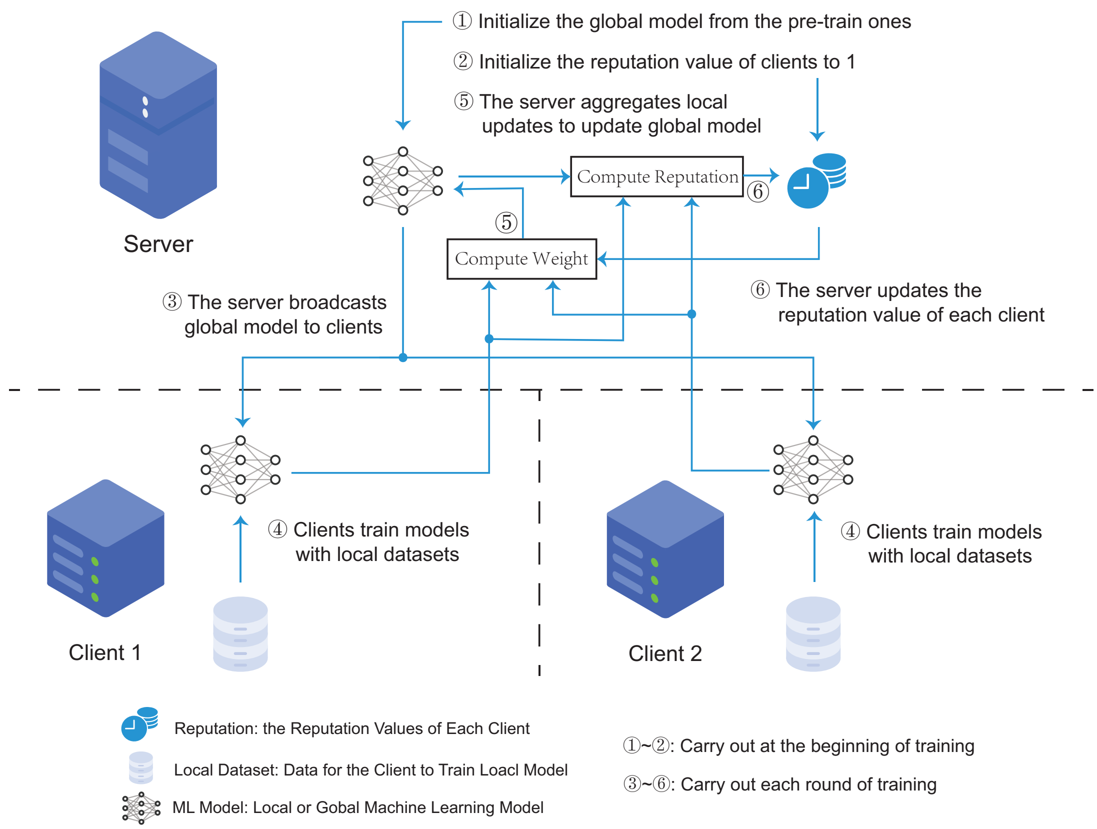

Fig. 2. A high-level overview of Fed-Credit.

3.1 Credibility Management Mechanism

In a FL system, users with higher credibility and more stable network connections contribute more to the training process. On the FL system initialization, it assigns each client a credibility value $\tau_i = 1$. After each round of training, it aggregates the client model parameters $\boldsymbol{\theta}_i$ into the global model, where the aggregation weight w_i of each client is determined by its credibility value τ_i. We then evaluate the credibility score S_i, which is computed by averaging the cosine similarity of each layer between local and global models. The algorithm for credibility value assignment is outlined in Algorithm 2.

The credibility value τ_i is then updated with the credibility score S_i using an exponential decay function to take the decaying effect over time into consideration. Thus an decay factor β is utilized to average historical credibility values. A larger β value signifies that the past credibility value holds lesser significance, thereby highlighting the augmented significance of the current credibility score S_i. Following this, to mitigate the attacks from malicious clients, we normalize credibility values by subtracting the minimum credibility value from all values:

$$\boldsymbol{\tau} = \boldsymbol{\tau} - \min(\boldsymbol{\tau}) \tag{1}$$

Note that in Algorithm 2, the bias parameters in each layer are also used to evaluate the S_i of the model.

Algorithm 1 The Fed-Credit algorithm.

Input: Clients with local training datasets, $\{C_1, C_2, C_3, \ldots, C_n\}$; learning rate η; batch size B; number of local training iterations E; number of communications R; hyperparameters $\alpha_1, \alpha_2, \beta$.
Output: Global model $\boldsymbol{\theta}$.

// **Step 1:** The server initiates the global model and the credibility values.
Initialize $\boldsymbol{\theta}$
$\boldsymbol{\tau} \leftarrow \mathbf{1}$. ▷ $\boldsymbol{\tau} = [\tau_1, \tau_2, \tau_3, \ldots, \tau_n]$ contains the credibility value of clients.
for $r \in R$ **do**
    // **Step 2:** The server broadcasts global model $\boldsymbol{\theta}$.
    The server sends global model $\boldsymbol{g}$ to all clients $\{C_1, C_2, C_3, \ldots, C_n\}$.
    // **Step 3:** Clients train models with local datasets.
    for $i = 1$ to n **do** ▷ do in parallel
        $\boldsymbol{g_i} \leftarrow$ **getLocalModel**$(\boldsymbol{\theta}, C_i, \eta, E, B)$
        Return $\boldsymbol{g_i}$ to server.
    end for
    // **Step 4:** The server aggregates local updates to update the global model.
    for $i = 1$ to n **do**
        $w_i \leftarrow$ **getWeight**$(R, \tau_i, \alpha_1, \alpha_2)$ ▷ τ_i is the credibility value of the i^{th} client
    end for
    $\boldsymbol{\theta} \leftarrow \sum_{i=1}^{n} w_i \cdot \boldsymbol{\theta}_i$ ▷ Combine local gradients
    // **Step 5:** The server updates the credibility value of each client.
    for $i = 1$ to n **do** ▷ do in parallel
        $\tau_i \leftarrow$ **getCredibility**$(\tau_i, \boldsymbol{\theta}_i, \boldsymbol{\theta}, \beta)$
    end for
    $\boldsymbol{\tau} \leftarrow \boldsymbol{\tau} - min(\boldsymbol{\tau})$
end for
Return the global model $\boldsymbol{\theta}$

Algorithm 2 The getCredibility function.

Input: Credibility value τ_i; local model $\boldsymbol{\theta}_i$; global model $\boldsymbol{\theta}$, hyperparameters β
Output: Updated credibility value τ_i

// Compute the credibility score S_i.
$S_i \leftarrow 0$ ▷ Initialize the credibility score
for *layer*① in $\boldsymbol{\theta}$ **do**
    $S_i += \langle \boldsymbol{\theta}_i[layer], \boldsymbol{\theta}[layer] \rangle / \|\boldsymbol{\theta}_i[layer]\| \|\boldsymbol{\theta}[layer]\|$ ▷ The sum of cosine similarity of each layer
end for
// Utilize the S_i and historical credibility value to update the new credibility value τ_i.
$\tau_i = \beta \cdot S_i + (1 - \beta) \cdot \tau_i$
Return τ_i

3.2 Updating Weight

Initially, the preference of the proposed scheme is to assign uniform weights w_i to all clients in order to prevent inadvertent misclassification of malicious clients. Subsequent to this initial phase, our aim is to identify malicious clients through their credibility values τ_i and significantly reduce their weights w_i during training to protect the global model $\boldsymbol{\theta}$. In pursuit of this objective, we introduced the equilibrium factor α, which is calculated based on a variant sigmoid function (2). In our weight formula (3), we utilized it to dynamically modulate the significance of both the credibility value τ_i and the average value within the weights w_i. The comprehensive weight calculation procedure is illustrated in Algorithm 3.

$$\alpha = \left(1 + e^{(-(R+\alpha_1)/\alpha_2)}\right)^{-1} \tag{2}$$

$$w_i = \frac{1}{n} \cdot (1 - \alpha) + \frac{\tau_i}{\sum_{j=1}^{n} \tau_j} \cdot \alpha \tag{3}$$

Algorithm 3 getWeight

Input: Credibility value τ_i; number of training rounds R; hyperparameters α_1, α_2.
Output: The weight of i^{th} client w_i.

```
1: // Compute the equilibrium factor α by the variant sigmoid function.
2: α = (1 + e^(−(R+α1)/α2))^(−1)
3: // Update the weight w_i
4: w_i = (1/n) · (1 − α) + (τ_i / Σ_{j=1}^{n} τ_j) · α
5: Return w_i
```

4 EXPERIMENT RESULT

4.1 Experiment Setup

Our experimental platform comprises the AMD EPYC 7742 64-Core Processor and the NVIDIA Tesla A100 40G computing accelerator. We conducted a comparative analysis of our approach, Fed-Credit, with several existing methods including FedAvg [4], GeoMed [16], Krum [14], Median [15], Multi-Krum [14], Trimmed [15], and FLTrust [19]. This evaluation was carried out on the MNIST and CIFAR-10 datasets, considering varying numbers of attackers as well as both iid and Non-iid data distribution settings. For MNIST, we choose a Multi-Layer Perceptron (MLP) network with two hidden layers and one output layer to train the global model. For CIFAR-10, We opt for a lightweight model called Compact Convolutional Transformers (CCT) [30], as its small size and effectiveness offer better potential in addressing the resource constraints of onboard FL devices. We utilized Dirichlet distribution to model Non-iid distribution [31]. The hyperparameter settings of this work are shown in Table 2.

MNIST: The MNIST dataset is a well-known collection of handwritten digits widely used in the field of machine learning. It consists of 60,000 training examples and 10,000 testing examples. Each image is a 28x28 grayscale image of a digit, ranging from 0 to 9. The MNIST dataset serves as a benchmark for evaluating image classification algorithms and has played a crucial role in advancing the field of deep learning.

CIFAR-10: The CIFAR-10 dataset is a popular benchmark dataset in the field of computer vision. It consists of 60,000 color images, each of size 32x32 pixels, divided into 10 different classes. The dataset serves as a standard evaluation tool for image classification algorithms and has played a significant role in advancing the field of deep learning.

Attack types: In our experiments, we mainly use two attack methods, data poisoning attacks and model poisoning attacks. Among the data poisoning attacks, we select the label flipping attack based on pairwise (PW) and symmetric (SM) matrices. As for model poisoning attacks, we have chosen three different implementations. Specifically, Constant Parameter (CP), where all model parameters are identical; Normal Parameter (NP), where returned model parameters follow a normal distribution; and Sign-Flip Parameter (SF), which returns a model with parameters opposite to those obtained during training.

<table>
<tr><th>Parameter</th><th>Description</th><th colspan="2">Value</th></tr>
<tr><td>n</td><td>Number of clients</td><td colspan="2">10</td></tr>
<tr><td rowspan="2">f</td><td rowspan="2">Number of malicious clients</td><td>Model Poison</td><td>1, 2, 3</td></tr>
<tr><td>Data Poison</td><td>1, 2, 4</td></tr>
<tr><td>η</td><td>Learning rate</td><td colspan="2">0.01</td></tr>
<tr><td rowspan="2">B</td><td rowspan="2">Batch size</td><td>MNIST</td><td>64</td></tr>
<tr><td>CIFAR-10</td><td>32</td></tr>
<tr><td>R</td><td>Number of global training epochs</td><td colspan="2">100</td></tr>
<tr><td rowspan="2">E</td><td rowspan="2">Number of local training epochs</td><td>MNIST</td><td>5</td></tr>
<tr><td>CIFAR-10</td><td>2</td></tr>
<tr><td>α_1</td><td rowspan="3">Hyperparameters of Fed-Credit</td><td colspan="2">1</td></tr>
<tr><td>α_2</td><td colspan="2">0.8</td></tr>
<tr><td>β</td><td colspan="2">0.1</td></tr>
</table>

Table 2. Hyperparameters settings

Evaluation: To evaluate the multiple defense model, as many other works [32] [29] [33], we adopted the accuracy as a criterion. The accuracy is employed to judge which represents the proportion of correctly classified samples to the total number of samples in the test dataset and is defined in Eq. (4).

$$\text{accuracy} = \frac{TP + TN}{TP + FN + FP + FN} \tag{4}$$

4.2 Numerical Analysis

Table 3 shows the accuracy of different robust aggregation rules under various attacks. We can find that Fed-Credit performs well in many situations.

4.2.1 Impact of Number of Malicious Clients.

First of all, our results demonstrated that with the absence of attacks, Fed-Credit, GeoMed, Median, Trimmed and FedAvg achieve relatively higher accuracy while FLTrust, Krum and Multi-Krum get lower accuracy. Especially, the disparity is more obvious when the dataset distribution is Non-iid. For instance, from the results shown in Figure 3, for MNIST with Non-iid distribution, the accuracy of lower three aggregation rules (FLTrust, Krum, Multi-Krum) are **72.26%, 85.30%, 93.08%** which are significantly lower than other methods that are around **96%**. This might be because Krum and Multi-Krum tend to use one or few local updates to update the global model, which makes the global model cannot fit the overall dataset well. Another finding is that the FLTrust converges slower than other methods, which is consistent with Cao *et al.* [19].

As indicated by the data presented in Figure 4 Figure 5, a clear pattern emerges where an increase in the number of malicious clients corresponds to a noticeable decline in accuracy. Additionally, it is worth noting that the FedAvg and Trimmed algorithms appear to be sensitive to the growing proportion of malicious attackers. This sensitivity can be attributed to the fact that both of these algorithms primarily rely on averaging methods to update the global model. When a malicious client is involved in the computation, it holds the same weight as a benign client, thereby contributing to a degradation in the overall model performance.

Dataset	Distribution	Attack type		f	Fed-credit	FedAvg	GeoMed	Krum	Multi-Krum	Median	Trimmed	FLTrust
MNIST	iid	No attack	-	0	**97.61**	97.63	**97.81**	95.64	96.33	97.64	97.59	86.85
		Model poison	CP	1	**97.55**	90.18	97.62	95.53	96.37	**97.66**	97.49	86.53
				2	**97.36**	83.09	97.34	95.56	96.32	**97.50**	88.34	86.58
				3	**97.07**	67.38	**97.17**	95.17	96.29	97.04	71.97	85.79
			NP	1	**97.55**	91.82	97.61	94.87	96.33	**97.63**	97.61	85.76
				2	**97.42**	88.39	97.39	94.93	96.43	**97.49**	90.55	86.37
				3	**97.22**	84.48	**97.31**	95.20	96.44	97.27	85.34	86.96
			SF	1	**97.55**	88.18	97.48	95.37	96.49	97.49	**97.55**	86.78
				2	**97.28**	76.42	97.47	94.53	96.45	**97.53**	85.27	85.22
				3	**97.23**	11.35	97.19	95.48	96.36	**97.44**	72.01	87.67
		Data poison	PW	1	**97.47**	97.51	97.59	94.88	96.39	**97.59**	97.52	87.64
				2	**97.47**	96.45	**97.48**	94.73	96.31	97.38	97.19	85.98
				4	**97.17**	81.22	96.81	95.22	96.46	96.47	85.33	85.54
			SM	1	**97.60**	97.56	**97.80**	94.98	96.33	97.58	97.62	86.12
				2	**97.46**	97.11	**97.51**	94.83	96.50	97.56	97.14	86.02
				4	**97.17**	93.33	**97.25**	95.19	96.34	96.95	94.65	86.19
	Non-iid	No attack	-	0	**96.18**	**96.43**	96.23	85.30	93.08	95.52	96.12	72.26
		Model poison	CP	1	**95.91**	70.87	95.49	69.13	93.24	94.71	94.98	70.45
				2	**95.76**	61.95	94.79	81.06	93.23	95.29	66.76	68.84
				3	**95.19**	27.31	92.71	83.88	93.16	94.15	38.47	61.70
			NP	1	**95.71**	78.47	95.53	74.57	92.89	94.41	95.27	67.44
				2	**96.29**	70.93	95.49	81.07	92.16	95.24	70.84	64.10
				3	**95.84**	57.87	93.57	81.51	92.94	95.00	54.09	67.38
			SF	1	**95.75**	67.26	**95.85**	76.50	93.18	95.02	95.26	67.86
				2	**95.90**	52.29	95.57	81.41	93.26	95.84	64.41	71.58
				3	**95.48**	21.08	93.61	82.65	92.95	94.78	27.18	70.09
		Data poison	PW	1	**95.97**	95.77	96.10	72.99	93.24	94.66	**96.21**	63.02
				2	**95.89**	95.60	96.33	73.79	91.90	95.23	95.87	70.08
				4	**94.96**	74.56	94.43	75.62	93.22	93.14	79.50	59.46
			SM	1	**96.02**	95.53	**96.17**	71.73	92.88	93.76	95.42	62.27
				2	**96.23**	95.60	**96.30**	81.09	92.88	95.12	96.14	68.88
				4	**93.78**	92.04	**95.14**	70.16	85.48	93.53	93.83	52.79
CIFAR-10	iid	No attack	-	0	**68.34**	68.99	**69.41**	58.20	61.09	68.08	68.72	46.91
		Model poison	CP	1	**69.55**	44.68	**69.72**	56.68	60.57	68.50	69.13	46.74
				2	**68.16**	15.04	68.02	57.72	61.62	68.25	18.50	46.77
				3	**65.97**	13.78	61.97	55.79	61.82	66.43	13.90	47.59
			NP	1	**69.24**	58.62	**68.95**	56.45	62.87	67.52	69.07	46.32
				2	**68.62**	37.45	**69.02**	58.17	62.90	66.70	54.96	46.87
				3	**68.60**	10.00	67.93	57.11	62.12	66.02	24.88	46.04
			SF	1	**69.67**	37.94	68.99	57.62	62.27	67.16	69.34	46.97
				2	**69.13**	10.00	68.24	58.49	62.22	65.59	24.65	46.24
				3	**69.40**	10.00	67.42	57.67	61.50	65.39	10.00	46.91
		Data poison	PW	1	**68.69**	67.13	68.21	54.24	62.26	67.08	68.25	45.99
				2	**68.01**	64.50	66.81	57.74	62.93	64.47	65.21	47.14
				4	**66.10**	48.09	50.43	55.85	61.07	48.95	48.06	45.28
			SM	1	**68.63**	67.85	68.01	56.40	62.73	66.00	68.27	46.29
				2	**68.57**	64.36	67.07	56.92	62.32	61.28	65.87	45.87
				4	**66.70**	56.98	52.39	55.41	61.96	51.28	55.92	45.82
	Non-iid	No attack	-	0	**67.68**	68.64	**69.14**	59.09	62.45	67.84	68.87	46.84
		Model poison	CP	1	**69.07**	44.63	68.41	56.52	61.87	68.57	69.04	47.52
				2	**68.57**	15.22	67.84	57.05	60.97	**68.68**	17.42	46.58
				3	**66.42**	14.36	60.55	58.98	62.01	**67.37**	14.08	46.56
			NP	1	**68.86**	59.27	**69.30**	59.53	61.73	67.15	69.23	47.08
				2	**69.11**	37.71	68.49	54.37	62.52	67.18	55.22	47.77
				3	**68.65**	10.00	67.08	57.39	62.56	65.98	25.47	45.83
			SF	1	**69.93**	37.99	69.17	58.63	61.55	67.55	68.94	46.58
				2	**69.38**	10.00	68.94	48.99	61.84	67.43	25.64	47.54
				3	**69.57**	10.00	67.51	55.73	60.69	64.42	10.00	46.34
		Data poison	PW	1	**68.32**	67.33	68.13	56.98	61.39	67.31	68.27	47.11
				2	**67.82**	64.31	65.89	56.61	60.75	65.90	65.94	47.20
				4	**66.75**	47.88	49.91	54.43	62.26	49.45	48.38	48.57
			SM	1	**68.79**	67.92	68.35	54.76	61.11	66.11	68.47	46.97
				2	**67.95**	66.14	66.46	45.83	61.65	62.59	65.90	46.13
				4	**66.49**	55.10	52.59	55.21	62.30	52.10	55.88	47.48

Table 3. Experiment results overview
Constant Parameter (CP), Normal Parameter (NP), Sign Flipping (SF), Pairwise (PW), Symmetric (SM)

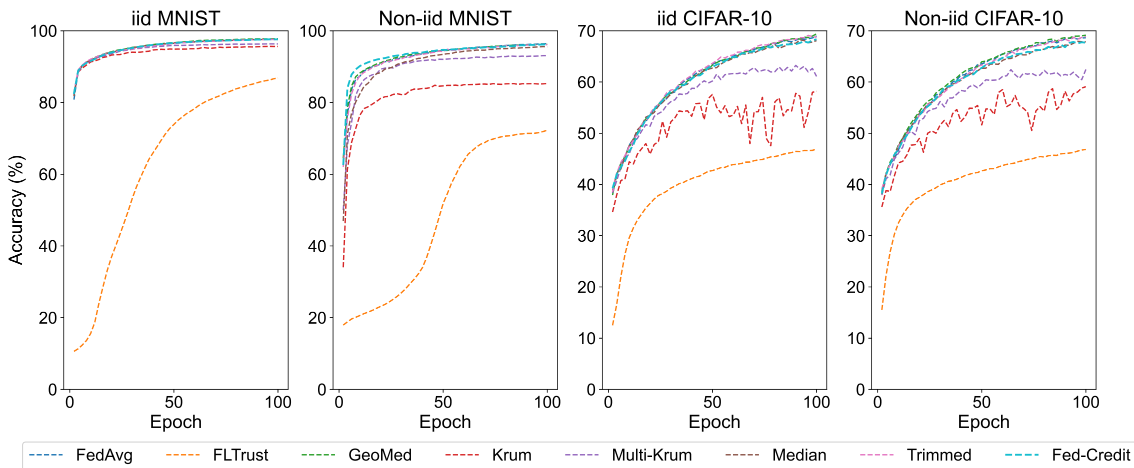

Fig. 3. Accuracy without attacks of iid and Non-iid datasets

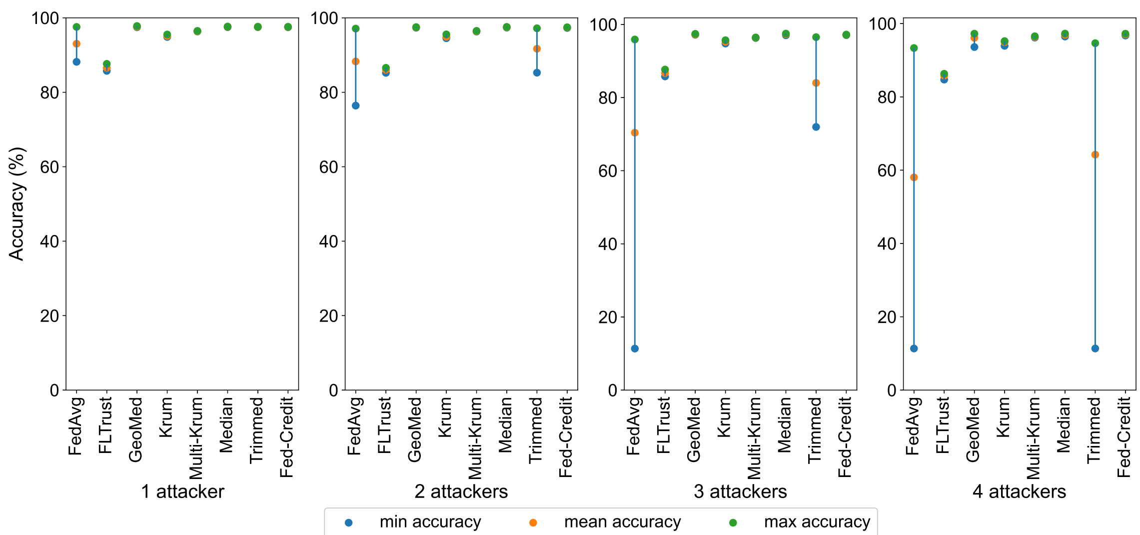

Fig. 4. The minimum, mean, maximum accuracy of various aggregation methods with 1,2,3,4 attacker(s) on iid MNIST. *Median* and *Fed-Credit* show high accuracy and narrow bias.

In contrast, the results also underscore the superior performance of the Fed-Credit algorithm. Notably, the Fed-Credit algorithm consistently maintains higher accuracy levels and demonstrates fewer instances of extreme variability. This fortifies the assertion that Fed-Credit adeptly preserves both accuracy and stability, even amidst the escalating presence of adversarial entities.

4.2.2 *Impact of Attack Types.*

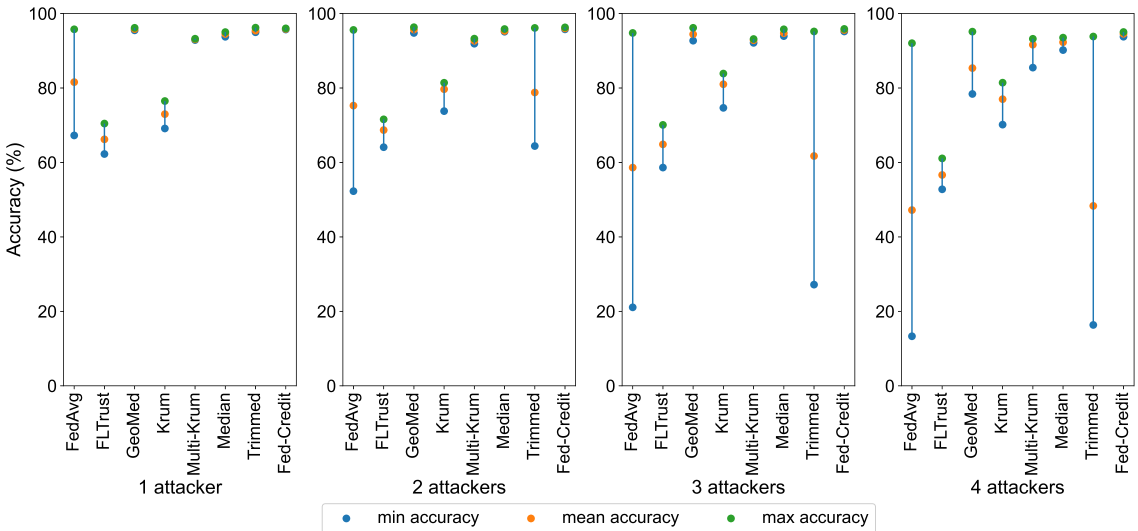

Fig. 5. The minimum, mean, maximum accuracy of various aggregation methods with 1,2,3,4 attacker(s) on Non-iid MNIST. *Fed-Credit* shows high accuracy and narrow bias.

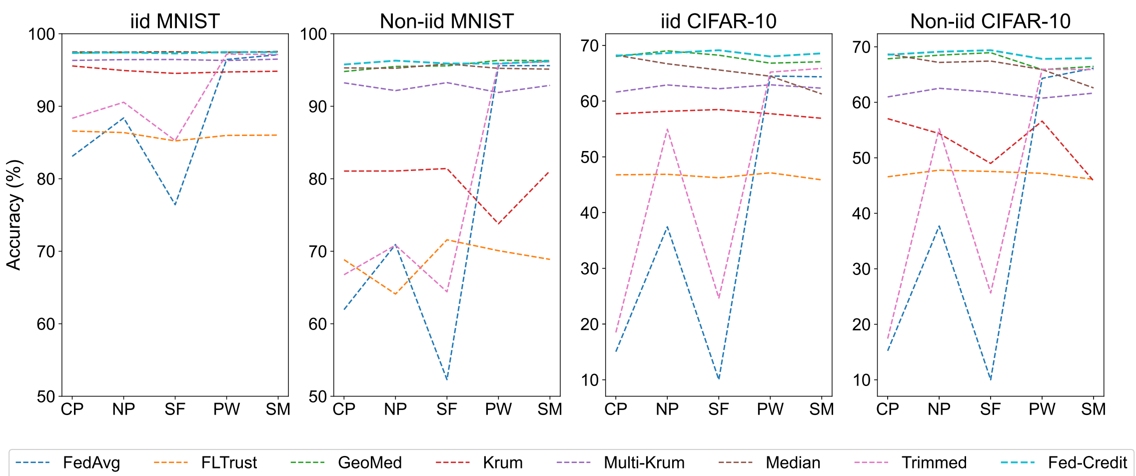

Fig. 6. Impact of different attack types on test accuracy for iid and Non-iid datasets. *Geomed, Trimmed* and *Fed-Credit* show higher tolerance than other methods.

It is worth discussing the impact of different attack types on the overall accuracy of the global model. From Figure 6, it is evident that various aggregation approaches show differing levels of effectiveness in countering a range of attack techniques across diverse datasets with distinct distributions. Specifically, when considering the scenario with two attackers, distinct patterns emerge.

For instance, both the FedAvg and Trimmed methods exhibit lower tolerance for Model Poison attacks (CP, NP, SF) compared to Data Poison attacks (PW, SM). On the contrary, algorithms like Krum, Multi-Krum, and FLTrust demonstrate a higher degree of tolerance for multiple attack types. Importantly, these algorithms exhibit sensitivity to only a limited number of attacks, with fluctuations that remain relatively contained compared to FedAvg and Trimmed. The Fed-Credit, Geomed, and Median algorithms consistently perform well, effectively mitigating all types of attacks with higher accuracy compared to alternative methods considered.

4.2.3 *Impact of Data Distribution.*

We conducted an assessment of the model's performance across distinct partitioned datasets. In scenarios where the data partition adheres to the iid principle, an equitable apportionment of each data category to every client was effected. Conversely, in instances characterized by Non-iid data distribution, the **Dirichlet Distribution** ($G \sim DP(\alpha, G_0)$) was employed as a means to characterize the prevailing data distribution dynamics.

Prior investigations [34] [35] have previously demonstrated the influence of Non-iid datasets on the convergence behavior of models. Our present study, Figure 4 Figure 5, confirmed this view. As the data distribution shifts from iid to Non-iid, the vast majority of methods show a downward trend in accuracy. In line with these antecedent findings, our own experimental endeavor reveals a supplementary facet: that adversarial attacks exhibit heightened efficacy in instances where the underlying dataset distribution is Non-iid. Notably, among the algorithms assessed, namely Fed-Credit, GeoMed, Krum, Multi-Krum, Median, and FLTrust, their predictive accuracy attains a comparable level to that observed under iid dataset conditions when confronted with Non-iid dataset configurations. However, it is noteworthy that both FedAvg and Trimmed algorithms manifest certain challenges in convergence within select scenarios. A case in point involves the application of 3 sign-flipping attackers on the Non-iid CIFAR-10 dataset, where these algorithms nearly regress to a state akin to random conjecture.

4.2.4 *Credibility Trend.*

Within our Fed-Credit algorithm, a key component that warrants attention is credibility management. This pivotal element profoundly influences the algorithm's operational framework. The graphical representation of credibility values across varying scenarios, employing the MNIST dataset, is concisely depicted in Figure 7. This illustration vividly showcases the algorithm's remarkable ability to withstand a diverse array of attacks.

A noteworthy discovery is the consistent trend of credibility values among benign clients, which converged to the same value that is significantly higher than the credibility values that malicious clients achieved. This observation serves as compelling evidence of the Fed-Credit algorithm's effectiveness.

For the group of malicious clients, aside from constant parameter attacks, their credibility values exhibit fluctuations as the number of attackers increases. This phenomenon becomes particularly pronounced when dealing with a Non-iid distributed dataset. It is essential to highlight, however, that this observed fluctuation remains well-contained within an acceptable and manageable scope. Overall, the Fed-Credit algorithm demonstrates a robust and reliable performance, showcasing its resilience across a wide spectrum of challenges.

5 RELATED WORK

In this section, we show the current research on poisoning attacks and aggregation rules for defending against attacks.

5.1 Poisoning Attacks

According to the poisoning attacks method, poisoning attacks can be classified into data poisoning attacks and model poisoning attacks [36].

In the data poisoning attack, the attackers can only inject poison into training data or labels. Therefore, we can divide the data poisoning attack into two categories, *clean label attack and dirty label attack.*

Clean label attack: The untargeted attack [6] [7] is a form of model poisoning attack. In this attack, malicious clients send arbitrary or counterfeit parameters to the central server with the aim of undermining the performance of the global model or causing it to deviate from its intended behavior. Ali *et al.* [8] proposed a method that optimizes

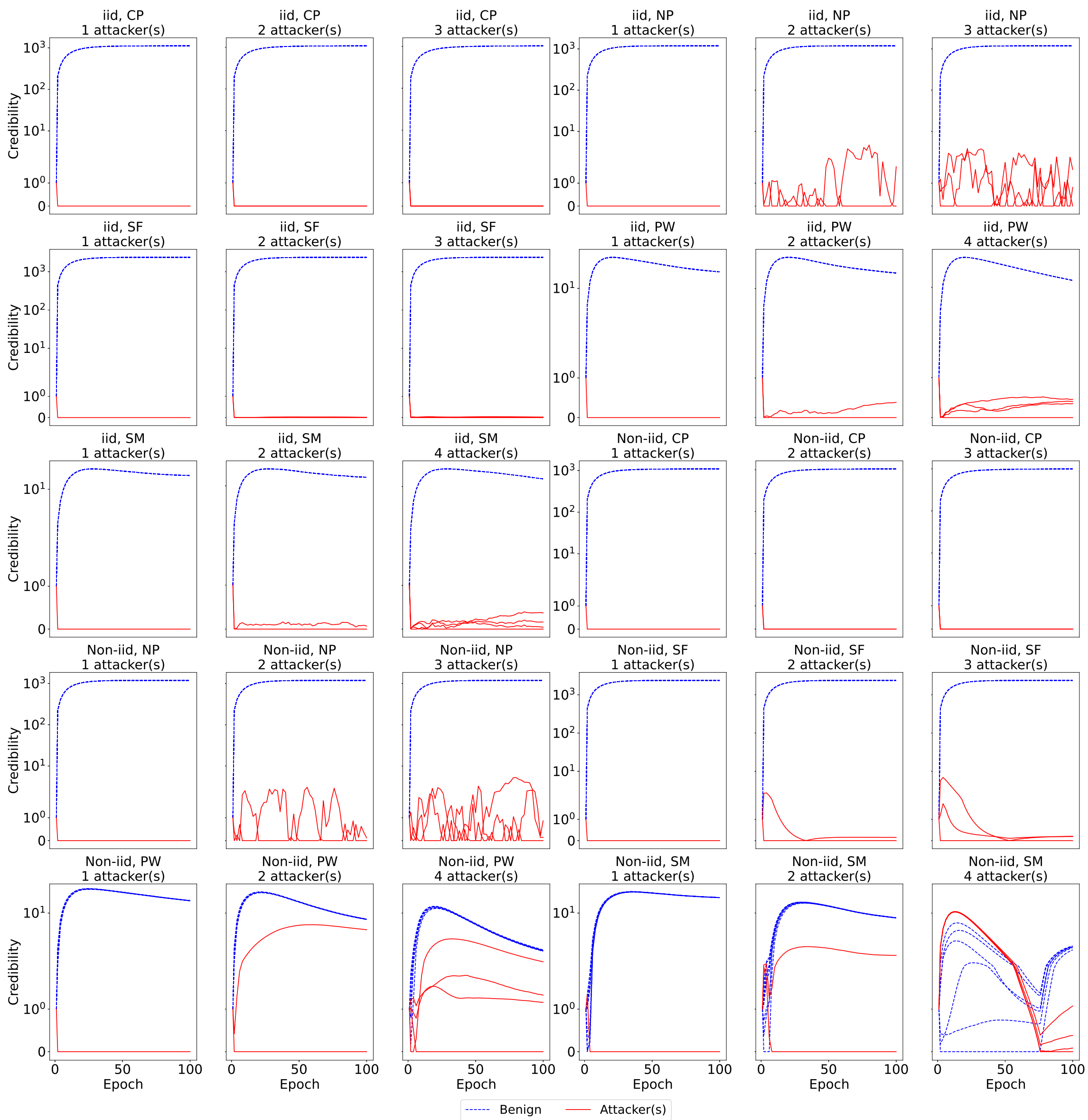

Fig. 7. Credibility values of benign clients and attackers with different attack types and varying number of attackers.

an equation to create a poison instance resembling a base class instance but embedded in the target class distribution. Dazhong *et al.* [37] designed FedRecAttack to employ public interactions for approximating the user's feature vector, which an attacker can exploit to train a malicious model. However, the above methods both assume the distribution of the dataset is iid, and if the distribution is Non-iid, the attackers cannot attack via these methods. To address this problem, Jiale *et al.* [38] utilize a generative adversarial network, called *PoisonGAN*, to generate data similar to other clients and execute attacks with these fake data, in which attackers could execute poisoning attacks without prior knowledge.

Dirty lable attack: Virat *et al.* [39] introduced that all label flipping can be divided into static label flipping (SLF) and dynamic label flipping (DLF). For instance, an attacker flips the label of "7" to "1" [12] [13] in SLF. This method has high requirement for prior knowledge which is not inefficient in practical application. To improve efficiency, symmetric flipping [10] and pairwise flipping [11] were introduced to flip each label to other labels. The attack distance-aware attack (ADA) was proposed by Yuwei *et al.* [40] to enhance poisoning attacks by discovering optimal target classes in the feature space.

Model poisoning aims to attack a global model by manipulating malicious clients' local model parameters directly. Li *et al.* [41] use the Same-value vector and Sign-flipping vector to attack the global model. Xie *et al.* [42] proposed Inner Product Manipulation (IPM) which aims to create a negative inner product between the genuine update mean and the aggregation schemes' output, thereby preventing any loss reduction. Wallach *et al.* designed ALIE to modify the local model parameters carefully based on the assumption that benign updates are expressed by a normal distribution. Xingchen et al. [43] proposed an optimization-based model poisoning attack, injecting malicious neurons into the neural network's redundant space using the regularization term. However, the primary issue with this approach was the computational complexity of malicious clients needing to compute the Hessian matrix during attack preparation.

5.2 Defense Rules

A variety of robust aggregation rules have been proposed. In general, they can be divided into the following three categories.

Distance-based rules aim to detect and reject abnormal local parameters which is uploaded by malicious clients. Blanchard *et al.* [14] proposed Krum and Multi-Krum. Krum chooses one update which is the most closest to its neighbors to update the global model, while Multi-Krum computes the mean of multiple updates to update global model. Cao *et al.* [44] presented Sniper, which constructs a graph based on Euclidean distances between local parameters, to ignore the updates from malicious clients. Wan *et al.* [18] designed MAB-RFL, which uses graph theory and principal components analysis (PCA) to distinguish honest and malicious in low-dimensional model space.

In performance-based rules, every update from clients will be evaluated with a clean dataset that is stored in server, then the server assigns weights for each update. Cao *et al.* [20] proposed a Byzantine-robust distributed gradient algorithm that filters out information from malicious clients by computing a noisy gradient with a small clean dataset and only accepting updates based on a pre-defined condition. Zeno [21] uses a small validation set to compute a score for each candidate gradient, considering the estimated loss function descendant and the update magnitude, indicating reliability and performance. Cao *et al.* [19] introduced FLTrust, which computes weights by ReLU-clipped cosine similarity between each local update and server update.

Statistics-based algorithms utilize statistical characteristics of updates to update the global parameters. Yin *et al.* [15] proposed Median and Trimmed to exploit the median of updates or the coordinate-wise trimmed mean of local parameters. Xie *et al.* [16] employed the geometric median, which requires more computational resources, to defend against the attacks. Mhamdi *et al.* [17] designed Bulyan, which combines malicious client detection algorithms, such as Multi-Krum, and Trimmed, to filter the updates from malicious clients.

Summary: *(1)* Although the current research has good results in defending against some kinds of attacks, few studies have discussed the effectiveness of aggregation rules against multiple attacks. *(2)* For the second category, it's impractical for the server to have a partially clean dataset due to privacy concerns. *(3)* some aggregation rules need to know in advance how many malicious clients there are, which cannot be put into practice. *(4)* high time complexity of one round of interaction for some aggregation rules.

6 CONCLUSION

In this paper, we first explored the practical use of the Federated Learning (FL) algorithm. We then proposed and evaluated a robust FL aggregation method named Fed-Credit. Through extensive experiments on MNIST and CIFAR-10 datasets, we compared Fed-Credit with several other algorithms. Results show that Fed-Credit maintains high accuracy while effectively countering a broad range of attacks. In our future work, we plan to integrate an outlier detection algorithm at the start of Fed-Credit to mitigate extreme local updates and preserve the credibility value system. Additionally, we aim to enhance the generality of Fed-Credit by providing clients with an initial credibility value from previous FL tasks.

ACKNOWLEDGMENTS

Acknowledgments goes here...

REFERENCES

[1] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *CoRR*, vol. abs/1610.02527, 2016.

[2] D. Pasquini, M. Raynal, and C. Troncoso, "On the (in)security of peer-to-peer decentralized machine learning," in *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, May 21-25, 2023*, pp. 418–436, IEEE, 2023.

[3] Z. Jiang, Y. Xu, H. Xu, L. Wang, C. Qiao, and L. Huang, "Joint model pruning and topology construction for accelerating decentralized machine learning," *IEEE Trans. Parallel Distributed Syst.*, vol. 34, no. 10, pp. 2827–2842, 2023.

[4] 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 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA* (A. Singh and X. J. Zhu, eds.), vol. 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282, PMLR, 2017.

[5] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 1, no. 2, pp. 44:1–44:25, 2017.

[6] M. Yang, H. Cheng, F. Chen, X. Liu, M. Wang, and X. Li, "Model poisoning attack in differential privacy-based federated learning," *Inf. Sci.*, vol. 630, pp. 158–172, 2023.

[7] V. Shejwalkar and A. Houmansadr, "Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning," in *28th Annual Network and Distributed System Security Symposium, NDSS 2021, virtually, February 21-25, 2021*, The Internet Society, 2021.

[8] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein, "Poison frogs! targeted clean-label poisoning attacks on neural networks," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada* (S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), pp. 6106–6116, 2018.

[9] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]* (S. Chiappa and R. Calandra, eds.), vol. 108 of *Proceedings of Machine Learning Research*, pp. 2938–2948, PMLR, 2020.

[10] B. van Rooyen, A. K. Menon, and R. C. Williamson, "Learning with symmetric label noise: The importance of being unhinged," in *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada* (C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, eds.), pp. 10–18, 2015.

[11] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. W. Tsang, and M. Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS*

2018, December 3-8, 2018, Montréal, Canada (S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), pp. 8536–8546, 2018.

[12] B. Biggio, B. Nelson, and P. 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.

[13] N. M. Jebreel, J. Domingo-Ferrer, D. Sánchez, and A. Blanco-Justicia, "Defending against the label-flipping attack in federated learning," *CoRR*, vol. abs/2207.01982, 2022.

[14] P. Blanchard, E. M. E. Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA* (I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, eds.), pp. 119–129, 2017.

[15] D. Yin, Y. Chen, K. Ramchandran, and P. L. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018* (J. G. Dy and A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, pp. 5636–5645, PMLR, 2018.

[16] C. Xie, O. Koyejo, and I. Gupta, "Generalized byzantine-tolerant SGD," *CoRR*, vol. abs/1802.10116, 2018.

[17] E. M. E. Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in byzantium," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018* (J. G. Dy and A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, pp. 3518–3527, PMLR, 2018.

[18] W. Wan, S. Hu, J. Lu, L. Y. Zhang, H. Jin, and Y. He, "Shielding federated learning: Robust aggregation with adaptive client selection," in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022* (L. D. Raedt, ed.), pp. 753–760, ijcai.org, 2022.

[19] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," in *28th Annual Network and Distributed System Security Symposium, NDSS 2021, virtually, February 21-25, 2021*, The Internet Society, 2021.

[20] X. Cao and L. Lai, "Distributed gradient descent algorithm robust to an arbitrary number of byzantine attackers," *IEEE Trans. Signal Process.*, vol. 67, no. 22, pp. 5850–5864, 2019.

[21] C. Xie, S. Koyejo, and I. Gupta, "Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance," in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA* (K. Chaudhuri and R. Salakhutdinov, eds.), vol. 97 of *Proceedings of Machine Learning Research*, pp. 6893–6901, PMLR, 2019.

[22] S. Li, Y. Cheng, Y. Liu, W. Wang, and T. Chen, "Abnormal client behavior detection in federated learning," *CoRR*, vol. abs/1910.09933, 2019.

[23] K. Hsieh, A. Phanishayee, O. Mutlu, and P. B. Gibbons, "The non-iid data quagmire of decentralized machine learning," in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, vol. 119 of *Proceedings of Machine Learning Research*, pp. 4387–4398, PMLR, 2020.

[24] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to byzantine-robust federated learning," in *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020* (S. Capkun and F. Roesner, eds.), pp. 1605–1622, USENIX Association, 2020.

[25] S. Sagar, C. Li, S. W. Loke, and J. Choi, "Poisoning attacks and defenses in federated learning: A survey," *CoRR*, vol. abs/2301.05795, 2023.

[26] C. Xie, O. Koyejo, and I. 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 2019, Tel Aviv, Israel, July 22-25, 2019* (A. Globerson and R. Silva, eds.), vol. 115 of *Proceedings of Machine Learning Research*, pp. 261–270, AUAI Press, 2019.

[27] B. Biggio, B. Nelson, and P. 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.

[28] J. Peng, Z. Wu, Q. Ling, and T. Chen, "Byzantine-robust variance-reduced federated learning over distributed non-i.i.d. data," *Inf. Sci.*, vol. 616, pp. 367–391, 2022.

[29] S. Li, E. C. H. Ngai, and T. Voigt, "An experimental study of byzantine-robust aggregation schemes in federated learning," *CoRR*, vol. abs/2302.07173, 2023.

[30] A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi, "Escaping the big data paradigm with compact transformers," *CoRR*, vol. abs/2104.05704, 2021.

[31] D. Zeng, S. Liang, X. Hu, H. Wang, and Z. Xu, "Fedlab: A flexible federated learning framework," *J. Mach. Learn. Res.*, vol. 24, pp. 100:1–100:7, 2023.

[32] Y. Zhao, J. Zhang, and Y. Cao, "Manipulating vulnerability: Poisoning attacks and countermeasures in federated cloud-edge-client learning for image classification," *Knowl. Based Syst.*, vol. 259, p. 110072, 2023.

[33] J. Shi, W. Wan, S. Hu, J. Lu, and L. Y. Zhang, "Challenges and approaches for mitigating byzantine attacks in federated learning," in *IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2022, Wuhan, China, December 9-11, 2022*, pp. 139–146, IEEE, 2022.

[34] T. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *CoRR*, vol. abs/1909.06335, 2019.

[35] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, OpenReview.net, 2020.

[36] G. Xia, J. Chen, C. Yu, and J. Ma, "Poisoning attacks in federated learning: A survey," *IEEE Access*, vol. 11, pp. 10708–10722, 2023.

[37] D. Rong, S. Ye, R. Zhao, H. N. Yuen, J. Chen, and Q. He, "Fedrecattack: Model poisoning attack to federated recommendation," in *38th IEEE International Conference on Data Engineering, ICDE 2022, Kuala Lumpur, Malaysia, May 9-12, 2022*, pp. 2643–2655, IEEE, 2022.

[38] J. Zhang, B. Chen, X. Cheng, H. T. T. Binh, and S. Yu, "Poisongan: Generative poisoning attacks against federated learning in edge computing systems," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3310–3322, 2021.

[39] V. Shejwalkar, A. Houmansadr, P. Kairouz, and D. Ramage, "Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning," in *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pp. 1354–1371, IEEE, 2022.

[40] Y. Sun, H. Ochiai, and J. Sakuma, "Semi-targeted model poisoning attack on federated learning via backward error analysis," in *International Joint Conference on Neural Networks, IJCNN 2022, Padua, Italy, July 18-23, 2022*, pp. 1–8, IEEE, 2022.

[41] L. Li, W. Xu, T. Chen, G. B. Giannakis, and Q. Ling, "RSA: byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets," in *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 1544–1551, AAAI Press, 2019.

[42] C. Xie, O. Koyejo, and I. 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 2019, Tel Aviv, Israel, July 22-25, 2019* (A. Globerson and R. Silva, eds.), vol. 115 of *Proceedings of Machine Learning Research*, pp. 261–270, AUAI Press, 2019.

[43] X. Zhou, M. Xu, Y. Wu, and N. Zheng, "Deep model poisoning attack on federated learning," *Future Internet*, vol. 13, no. 3, p. 73, 2021.

[44] D. Cao, S. Chang, Z. Lin, G. Liu, and D. Sun, "Understanding distributed poisoning attack in federated learning," in *25th IEEE International Conference on Parallel and Distributed Systems, ICPADS 2019, Tianjin, China, December 4-6, 2019*, pp. 233–239, IEEE, 2019.