Blockchain-Based Federated Learning with Malicious Attacks in Fog Computing Networks

Xiaoge Huang(✉), Wenjing Li, Yang Ren, and Qianbin Chen

School of Communications and Information Engineering,
Chongqing University of Posts and Telecommunications,
Chongqing 400065, China
huangxg@cqupt.edu.cn

Abstract. In fog computing networks, the discrete deployment of a large number of fog nodes (FNs) and the frequent information exchange pose significant risks to user privacy and network security. Federated Learning (FL), as an emerging distributed machine learning framework, provides a promising approach to privacy protection. However, non-independent and identically distributed (Non-IID) data and malicious attacks inevitably lead to a decrease in the performance of FL. In this paper, to ensure the security and effectiveness of FL, a Directed Acyclic Graph (DAG) blockchain-based FL (DAGFL) algorithm is proposed. Firstly, to reduce the involvement of malicious models in global aggregation, we propose a device selection algorithm based on reputation and user activity degree, along with an outlier-based malicious model detection algorithm. Besides, to mitigate the negative impact of Non-IID data, a weighted FL aggregation algorithm is introduced, which assigns weights to local models by evaluating their contribution and accuracy. Furthermore, DAG blockchain technology is applied in FL to bolster the security and efficiency of model sharing among FNs. Finally, simulation results show the effectiveness of the proposed algorithms.

Keywords: Federated learning · DAG blockchain · Malicious model identification · Fog computing network

1 Introduction

In fog computing networks, machine learning (ML) is commonly employed for data analysis, mining, and pattern recognition. However, traditional ML requires the collection of user data for model training, which poses a significant challenge to user privacy. To address privacy concerns, B. McMahan et al. proposed federated learning (FL) [6] trained on decentralized clients, which enables clients

Supported by the National Natural Science Foundation of China (61831002, 62371082, 62001076), Natural Science Foundation of Chongqing (CSTB2023NSCQ-MSX0726, cstc2020jcyj-msxmX0878).

F. Gao et al. (Eds.): Chinacom 2023, LNICST 590, pp. 50–65, 2024.
https://doi.org/10.1007/978-3-031-67162-3_4

to train their respective local models on their own local data and then upload model parameters, instead of raw data, to the central server for generating a global model.

However, when FL is applied to the real world, many challenges arise in comparison to centralized learning, including resource limitations for clients and the randomness and heterogeneity of a large number of devices. In particular, a lot of research has indicated that non-independent and identically distributed (Non-IID) or heterogeneous data almost inevitably leads to a degradation in the accuracy of federated learning [12]. In an effort to address this issue, [3] proposed a grouping mechanism to reduce the difference in data distribution between each group and the global. In addition, since model parameters are generated based on training local data, they can reflect the different degrees of the original data. It is difficult to evaluate the heterogeneity of model parameters with Euclidean distance. To address this, Huang et al. proposed to evaluate the distribution differences of model parameters based on cosine similarity [4].

Furthermore, in FL combined with fog computing network architecture, the update and storage of the global model completely rely on the central server and fog nodes (FNs), which may result in single points of failure (SPOF) or targeted attacks. Fortunately, due to the strong compatibility between blockchain and FL, FL could leverage the decentralized, immutable, and secure nature of blockchain to protect user data safety and privacy. After Kim et al. [5] first proposed that using blockchain to store model parameters could enhance the security of FL, extensive research has been carried out. [8] proposed that combining blockchain and FL could effectively resist attacks from malicious nodes. In [2], the authors applied FL and blockchain to 5G-enabled unmanned aerial vehicles (UAVs) for the purpose of verifying cross-domain UAV identities and eliminating malicious UAVs. However, current research primarily focuses on integrating FL with single-chain blockchains, which have limited throughput and cannot provide low-latency services to Internet of Things devices (IDs) in high-concurrency business scenarios.

To enhance both the training efficiency of FL and network security, we integrate Directed Acyclic Graph (DAG) blockchain technology into the FL algorithm in this paper. The main contributions of this paper are as follows:

- Firstly, a DAG blockchain-based fog computing system model is proposed for FL, which consists of the cloud application layer, DAG blockchain layer, edge layer, and ID layer.
- Secondly, to ensure the global model security, we propose a reputation and activity degree-based device selection (RADDS) algorithm to select participating IDs and an outlier-based malicious model identification (OMMI) algorithm to eliminate malicious local models.
- Thirdly, to reduce the negative impact of Non-IID data on FL performance, a weighted FL aggregation (WFLA) algorithm is proposed, which calculates the weights of local models by considering both the gradient similarity between the local and global models, as well as the accuracy of local models.
- Fourthly, a DAG blockchain-based FL (DAGFL) algorithm is proposed to enhance the security and efficiency of model sharing among FNs.

– Finally, the experiment results are provided to evaluate the effectiveness of the proposed algorithms in Non-IID data scenarios with malicious attacks.

The rest of this paper is organized as follows. The system structure is presented in Sect. 2. In Sect. 3, we propose the WFLA algorithm and the DAGFL algorithm to ensure secure and efficient services in Non-IID data scenarios. Simulation results are discussed in Sect. 4. Section 5 draws conclusions.

2 System Structure

2.1 Network Model

The DAG blockchain-based fog computing network is shown in Fig. 1, which consists of the cloud application layer, DAG blockchain layer, edge layer, and ID layer. In this network, the integration of FL and DAG blockchain technology realizes distributed learning, effectively mitigating the challenges of SPOF and malicious attacks.

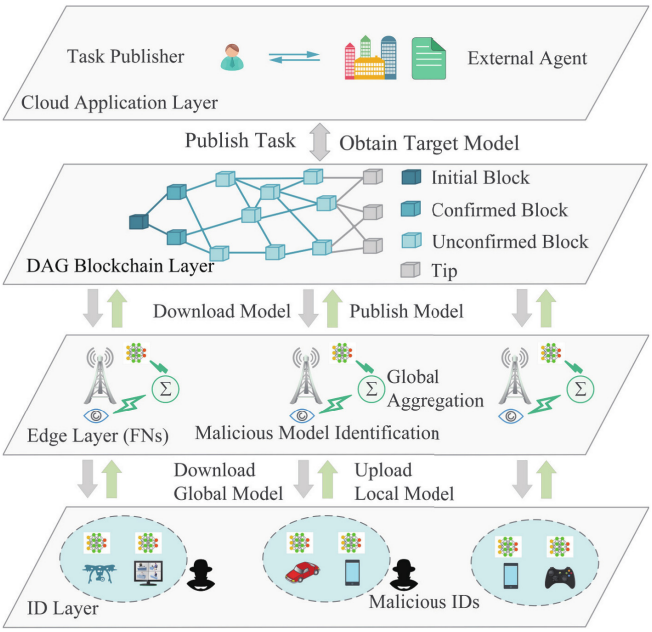

Fig. 1. DAG Blockchain-based Fog Computing Network.

ID layer: IDs download the global models from FNs, execute local training using their local data, and then transmit their local models to the edge layer. However, certain IDs may be vulnerable to malicious attacks, resulting in the upload of malicious local models.

Edge layer: It is composed of K FNs, denoted as $\mathcal{K} = \{1, \cdots, k, \cdots, K\}$, which provides abundant computational and communication resources for the

DAG blockchain layer and the ID layer. Each FN manages M IDs, denoted as $\mathcal{M} = \{1, \cdots, m, \cdots, M\}$, and the corresponding local dataset is denoted as $\mathcal{D} = \{D_1, \cdots, D_m, \cdots, D_M\}$. In FL, FNs serve as edge servers and are responsible for ID selection, malicious model identification, and global aggregation.

DAG blockchain layer: Blockchain nodes are deployed on FNs to ensure the secure storage of the global models generated by the edge layer.

Cloud application layer: It contains one central cloud of external agents, which coordinates FL task delivery and provides a global solution. Task publishers broadcast training tasks to FNs through external agents, including information such as the initial global model, data types, data sizes, training time, and desired model accuracy.

2.2 Federated Learning Model

To protect the data privacy of IDs, FL is adopted, which achieves global aggregation based on selected local models. Specifically, to obtain the $(t+1)$-th round global model w_g^{t+1}, ID l trains the $(t+1)$-th round local model w_l^{t+1} using its local dataset D_l and the t-th round global model w_g^t, then uploads its local model to the corresponding FN for the global aggregation. Additionally, the loss function of the local model w_l^{t+1} on a data sample $(x_j, y_j) \in \mathbf{D}_l$ is denoted as $f_j\left(w_g^t, x_j, y_j\right)$. Then, the local loss function $F_l(w_l^{t+1})$ and the global loss function $F(w_g^{t+1})$ could be defined as follows:

$$F_l(w_l^{t+1}) = \frac{1}{|D_l|} \sum_{j=\mathbf{D}_l} f_j(w_g^t, x_j, y_j) \tag{1}$$

$$F(w_g^{t+1}) = \sum_{l \in \mathbf{L}} \frac{|D_l|}{|D|} F_l(w_l^{t+1}) \tag{2}$$

Furthermore, the stochastic gradient descent (SGD) algorithm is applied to the local training to minimize the local loss function, as follows:

$$w_l^{t+1} = w_g^t - \eta \times \nabla F(w_l^t) \tag{3}$$

where η is the learning rate of the SGD algorithm.

Ultimately, the goal of FL is to minimize the global loss function, as follows:

$$Q(w) = arg \min F(w_g^{t+1}) \tag{4}$$

3 DAG Blockchain-Based Federated Learning

In this section, the WFLA algorithm is proposed to resist malicious attacks and address the challenges posed by Non-IID data. Then, we design a DAGFL algorithm that integrates the DAG blockchain and FL to ensure network security and training efficiency.

3.1 Reputation and Activity Degree-Based Device Selection Algorithm

In FL, after every T rounds of global training, FN $k \in \mathcal{K}$ uses the RADDS algorithm to reselect L IDs to participate in FL training. Specifically, the RADDS algorithm will choose IDs with high reputation values to reduce malicious local models. Additionally, the user activity degree is considered to address the issue of model overfitting caused by the excessive selection of high-reputation IDs. The algorithm mainly includes the following steps.

Firstly, FN k calculates the reputation values of M IDs. Taking ID m as an example, its reputation value is determined by three components: trust score b_{km}, distrust score d_{km}, and uncertainty score u_{km}.

$$\begin{cases} b_{km} = q_{km} \frac{\kappa \alpha_{km}}{\kappa \alpha_{km} + \eta \beta_{km}} \\ d_{km} = q_{km} \frac{\eta \beta_{km}}{\kappa \alpha_{km} + \eta \beta_{km}} \\ u_{km} = 1 - q_{km} \end{cases} \tag{5}$$

where q_{km} denotes the probability of successful data transmission. α_{km} and β_{km} respectively represent the number of normal and malicious local models uploaded by ID m during T global rounds, satisfying the condition $\alpha_{km} + \beta_{km} = T$. Additionally, κ and η respectively represent the weights assigned to normal and malicious local models.

Therefore, the reputation value of ID m in the $(t+1)$-th global round, denoted as $T_{km}(t+1)$, can be calculated as follows:

$$T_{km} = b_{km} + a u_{km} \tag{6}$$

$$T_{km}(t+1) = \lambda T_{km}(t) + r T_{km} \tag{7}$$

where $a \in [0, 1]$ denotes the impact on uncertainty, λ is the decay weight of the historical reputation value, and r is the weight of the current reputation value.

Secondly, set a reputation threshold $T_{\min}$ to select candidate IDs with reputation values higher than $T_{\min}$, which are denoted as $\mathcal{N} = \{1, \ldots, n, \ldots, N\}$, and $L \leq N \leq M$.

Thirdly, FN k calculates the activity degree of N candidate IDs. The activity degree is defined as the number of global rounds in which an ID participates in the current task.

Fourthly, considering both reputation values and activity degree, FN k calculates the utility values of the N IDs. Taking ID m as an example, its utility value can be obtained using the following formula:

$$U_{km} = \beta P T_{km}(t+1) - (1 - \beta) C_{km} \tag{8}$$

where C_{km} denotes the activity degree of ID m, the parameter β is used to balance the importance of the reputation value and activity degree, and the constant P is used to map the reputation value T_{km} to the same range as C_{km}.

Finally, FN k selects the top L IDs with the highest utility values to form the participating ID set denoted as $\mathcal{L} = \{1, \ldots, l, \ldots, L\}$, where $L \leq N$, maximizing the total utility value.

3.2 Outlier-Based Malicious Model Identification Algorithm

The data distributions of IDs could be significantly different since they belong to different individuals and enterprises, that is, the local data are Non-IID. In Non-IID data scenarios without malicious attacks, the parameter distributions of the local and global models have minimal differences. However, when malicious attacks occur, a significant disparity can arise in the parameter distribution between the local and global models. Based on this, the OMMI algorithm can identify malicious local models by detecting outliers among the parameter distances at each layer using the boxplot method.

In each global round, FN k distributes the global model to L participating IDs, and both the local and global models consist of I neural network layers. In the $(t+1)$-th global round, the OMMI algorithm executes the following specific steps.

Firstly, FN k collects local models from the L selected IDs, forming a set $\mathbf{W} = (w_1^{t+1}, \ldots, w_l^{t+1}, \ldots, w_L^{t+1})$. Then, it calculates the parameter distances between each layer of local models and the corresponding layer of the global model. For example, the parameter distance of the i-th layer between the local model w_l^{t+1} and the global model w_g^t can be calculated using the following formula:

$$d_{il} = ||w_{ig}^t - w_{il}^{t+1}|| \tag{9}$$

Secondly, sort the parameter distances of the i-th layer in ascending order, and then split them into quartiles to obtain the lower quartile Q_{i_1}, upper quartile Q_{i_3}, and inter-quartile range IQR_i. Subsequently, the minimum and maximum observed values of the parameter distances can be calculated using the following formulas:

$$lower_{i\_thr} = Q_{i\_1} - (1.5 \times IQR_i) \tag{10}$$

$$upper_{i\_thr} = Q_{i\_3} - (1.5 \times IQR_i) \tag{11}$$

Finally, parameter distances beyond the normal range $[lower_{i\_thr}, uupper_{i\_thr}]$ are identified as outliers, indicating a malicious local model. Conversely, if the parameter distances for each layer of the local model are not outliers, the model is considered normal. Based on this, the OMMI algorithm could effectively select normal local models denoted as $\mathbf{W}^{Prel} = (w_1^{t+1}, w_2^{t+1}, \ldots, w_{L'}^{t+1})$, where $L' \leq L$.

3.3 Weighted Federated Learning Aggregation Algorithm

Generally, the difference between the local models trained with the Non-IID data is greater than with the IID data, leading to a larger edge model difference, which will decrease the model accuracy and the global model convergence rate [10]. To address this issue, we propose the WFLA algorithm, which assigns larger weights to local models with greater contributions. Specifically, the proposed algorithm quantifies the contribution of each local model by calculating the angle

between the local model gradient and the global model gradient. Additionally, it incorporates the accuracy of each local model to determine the aggregation weight. In the $(t+1)$-th global round, the algorithm includes the following steps.

Firstly, calculate the angle $\theta_l(t+1)$ to quantify the contribution of the local model uploaded by ID l, as follows:

$$\theta_l(t+1) = \arccos \frac{\langle \nabla F(w_g^t), \nabla F(w_l^t) \rangle}{\left\| \nabla F(w_g^t) \right\| \left\| \nabla F(w_l^t) \right\|} \tag{12}$$

where $\nabla F(w_g^t)$ denotes the average gradient of the $(t+1)$-th round global model and can be obtained by:

$$\nabla F(w_g^t) = \sum_{l \in \mathbf{W}^{Prel}} \frac{|D_l|}{|D|} \nabla F(w_l^t) \tag{13}$$

When the angle $\theta_l(t+1)$ is smaller, the local model gradient $\nabla F(w_l^t)$ is more similar to the global model gradient $\nabla F(w_g^t)$, leading to a greater contribution to the global aggregation. Conversely, if $\theta_l(t+1)$ is large, exceeding $\pi/2$ for example, the local model gradient can negatively affect the global aggregation, as it points in the opposite direction to the global model gradient.

Secondly, in each global round, the randomness of the angle $\theta_l(t+1)$ will lead to an unstable quantification of the contributions of local models. To eliminate the impact of randomness, the angle $\theta_l(t+1)$ is replaced by the average angle $\tilde{\theta}_l(t+1)$, which is defined as follows:

$$\tilde{\theta}_l(t+1) = \begin{cases} \theta_l(t+1) & t = 0 \\ \frac{t}{t+1}\tilde{\theta}_l(t) + \frac{1}{t+1}\tilde{\theta}_l(t+1) & t > 0 \end{cases} \tag{14}$$

where the value of angle $\tilde{\theta}_l(t+1)$ depends on the proportion of Non-IID data in the local dataset D_l. The larger the proportion of Non-IID data, the greater the value of angle $\tilde{\theta}_l(t+1)$.

Thirdly, evaluate the accuracy of the local model uploaded by ID l on the test dataset, and define this accuracy as a_l. Then, by combining the average angle $\tilde{\theta}_l(t+1)$ with the local model accuracy a_l, the aggregation weight of the local model can be calculated as follows:

$$\tilde{\psi}_l(t+1) = \frac{p_l e^{f(\tilde{\theta}_l(t+1))}}{\sum_{l' \in \mathbf{W}^{Prel}} p_{l'} e^{f(\tilde{\theta}_{l'}(t+1))}} \tag{15}$$

where $f(\tilde{\theta}_l(t+1)) = \alpha(1 - e^{-e^{-\alpha(\tilde{\theta}_l(t+1)-1)}})$ is the Gompertz function, α is a constant, and p_l is given by:

$$p_l = \frac{a_l}{\sum_{l' \in \mathbf{W}^{Prel}} a_{l'}} \tag{16}$$

Finally, aggregate all normal local models according to their aggregation weights to obtain a new global model, as shown in the following formula:

$$F(w_g^{t+1}) = \sum_{l \in W^{Prel}} \tilde{\psi}_l(t+1) F(w_l^{t+1}) \tag{17}$$

To sum up, the details and processes of the WFLA algorithm are shown in Algorithm 1.

Algorithm 1. WFLA Algorithm

Require: Normal local model set $\mathbf{W}^{Prel}$, learning rate η, test dataset Im, global model w_g^t at global epoch t

Ensure: a new global model w_g^{t+1}

1: Calculate the gradient $\nabla F(w_l^t)$ for each normal local model $w_l^{t+1} \in \mathbf{W}^{Prel}$ using formula (3)
2: Calculate the average gradient $\nabla F(w_g^t)$ of the current global model w_g^t using formula (13)
3: **for** $w_l^{t+1} \in \mathbf{W}^{Prel}$ **do**
4: Calculate the angle $\theta_l(t+1)$ of the local model w_l^{t+1}
5: Calculate the average angle $\tilde{\theta}_l(t+1)$ of the local model w_l^{t+1}
6: FN k obtains the accuracy value a_l of the local model w_l^{t+1} based on the test dataset Im.
7: **end for**
8: Calculate the normalized accuracy value p_l for each normal local model $w_l^{t+1} \in \mathbf{W}^{Prel}$ using formula (16)
9: Calculate the aggregation weight $\tilde{\psi}_l(t+1)$ for each normal local model $w_l^{t+1} \in \mathbf{W}^{Prel}$ using formula (15)
10: Aggregate all normal local models into a new global model w_g^{t+1} using formula (17)

3.4 DAG Blockchain-Based Federated Learning Algorithm

Compared to traditional single-chain blockchains, DAG blockchains can execute transactions concurrently and add blocks asynchronously, leading to reduced block confirmation delays and increased network throughput. To ensure secure and efficient computation, the DAGFL algorithm is proposed, which combines DAG blockchain technology with the WFLA algorithm. In the DAGFL algorithm, the DAG blockchain stores the global models generated by FNs, and FNs also store the ledger information of the DAG blockchain network.

As shown in Fig. 2, in the first global round, FN k trains the initial global model broadcasted by the task publisher. In each subsequent global round, FN k generates a DAG blockchain-based global model and then trains it to obtain a new global model. The implementation follows the steps below:

– Step 1: Generate a DAG blockchain-based global model, denoted as w_g^t. Firstly, FN k uses the tip selection algorithm to randomly select α tips from the local DAG blockchain replica within the aging range, and then verifies the validity of their data. Secondly, FN k extracts global models from the selected tips and evaluates each global model's accuracy using the test dataset. Thirdly, FN k uses the FedAvg algorithm to aggregate the top c ($2 \leq c \leq \alpha$) global models with the highest accuracy into a DAG blockchain-based global model.

- Step 2: Device selection. After every T global rounds, FN k uses the RADDS algorithm to reselect IDs participating in FL training.
- Step 3: Local training. After FN k broadcasts the DAG blockchain-based global model w_g^t to the selected IDs, the IDs execute local training based on their local data.
- Step 4: Malicious model identification and global aggregation. FN k collects the local models uploaded by selected IDs, uses the OMMI algorithm to identify malicious models, and then aggregates all the normal local models into a new global model, denoted as w_g^{t+1}, using the WFLA algorithm.
- Step 5: DAG blockchain update. Firstly, FN k packages the new global model along with its digital signature into a new block and then broadcasts it to other FNs. Additionally, the verified parent blocks of the new block are the c selected tips from step 2. Finally, after receiving the new block, other FNs verify its validity and then add it to their local DAG blockchain replicas.

Repeat the above steps until the accuracy of the global model meets the requirement, and then return the final target model to the task publisher.

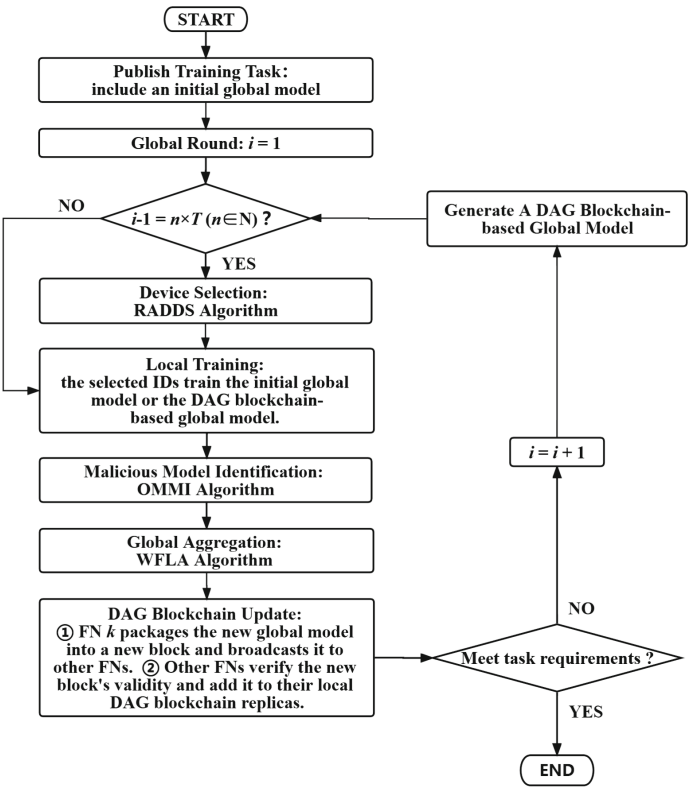

Fig. 2. Flowchart of the DAGFL Algorithm.

4 Simulation Results

4.1 Simulation Parameters

In the simulation, 5 FNs are randomly deployed, with each FN having 20 associated IDs. These FNs share their respective global models through the DAG blockchain network. Besides, a two-layer convolutional neural network (CNN) is trained on the MNIST dataset during the model training process.

Additionally, the local datasets of IDs are divided into two types: IID data and Non-IID data. In the IID scenario, the MNIST dataset is randomly and uniformly distributed among IDs, ensuring that each IID dataset contains labels for all categories. In the Non-IID scenario, each Non-IID dataset randomly includes 4 or 5 different categories of labels from the MNIST dataset.

Finally, the three attack methods employed by malicious IDs include the label flipping attack [7], the Byzantine attack [9], and the model compression attack. In the label flipping attack, malicious IDs flip the target labels to other categories and then use the contaminated dataset to train local models. In the Byzantine attack, malicious IDs generate local model parameters based on a standard normal distribution with a mean and a unit standard deviation. In the model compression attack, malicious IDs randomly prune the parameters of locally trained models and upload the pruned local models to FNs.

4.2 Baseline Algorithms

To evaluate the performance of the proposed algorithms, several baseline algorithms are also trained under the same malicious attacks.

- Median-based malicious model identification (MMMI) algorithm [11]: It reduces the negative impact of malicious models on the global model's performance by updating the global model with the median of all local model parameters.
- Isolated forest-based anomaly detection (IFAD) algorithm [1]: It detects anomalous local models based on isolation forests.
- All device selection (ADS) algorithm: In FL, it selects all IDs to participate in the training process.
- Random device selection (RDS) algorithm: In FL, it randomly selects a subset of IDs to participate in the training process.
- Full model selection aggregation (FMSA) algorithm: It treats all local models as trusted normal models, enabling global aggregation based on all local models.
- Single FN based FL (SFNFL) algorithm: It allows a single FN to independently select its associated IDs to conduct FL.
- Centralized FL (CFL) algorithm: It aggregates all local models trained by the IDs through a central server for FL.

4.3 Simulation Results and Analysis

By varying the proportion of malicious models, Fig. 3 demonstrates the effectiveness of the proposed OMMI algorithm in mitigating label flip poisoning attacks. As the number of malicious IDs increases, the accuracy of the global model gradually decreases. When 10% of IDs are malicious, the performance is similar to the scenario without malicious IDs. Furthermore, when 24% of IDs are malicious, the model accuracy and convergence speed are comparable to the scenario with 10% malicious IDs. However, when 25% of IDs are malicious, the accuracy decreases and the convergence speed slows down. Therefore, when the proportion of IDs uploading malicious local models is below 25%, the OMMI algorithm performs well. Subsequently, in the following simulation experiments, the proportion of malicious IDs is set to 24%.

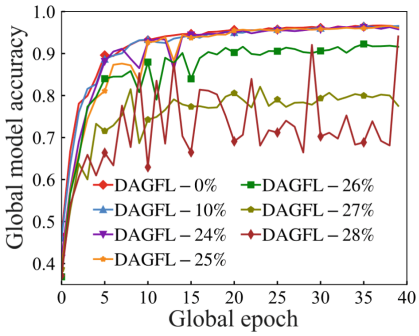

Fig. 3. Impact of malicious IDs with different proportions on the global model accuracy.

In Fig. 4, the impact of 24% malicious IDs on the global model accuracy is presented under three attack models, C1, C2, and C3. In C1 and C2, we consider the label flipping attack and the byzantine attack, respectively, and in C3, the model compression attack is involved. It can be seen from the figure that in both C1 and C2, the OMMI algorithm's global model achieves higher accuracy, almost matching the accuracy when there are no malicious IDs. However, in C3, the accuracy of the OMMI algorithm's model experiences some fluctuations. This is because the 24% proportion of malicious IDs is close to the defense upper bound of the OMMI algorithm. Moreover, in the model compression attack, the uploaded model parameters, apart from the pruned parameters, are highly similar to those trained normally. As a result, the malicious model recognition performance of the OMMI algorithm in C3 is inferior to that in C1 and C2. Conversely, in C1, C2, and C3, the accuracy of the FMSA algorithm's model experiences a significant decline because it lacks malicious model recognition, with the most severe impact occurring in C3. In conclusion, the proposed OMMI algorithm demonstrates the capability to recognize all three types of attacks, excelling notably in identifying Byzantine attacks.

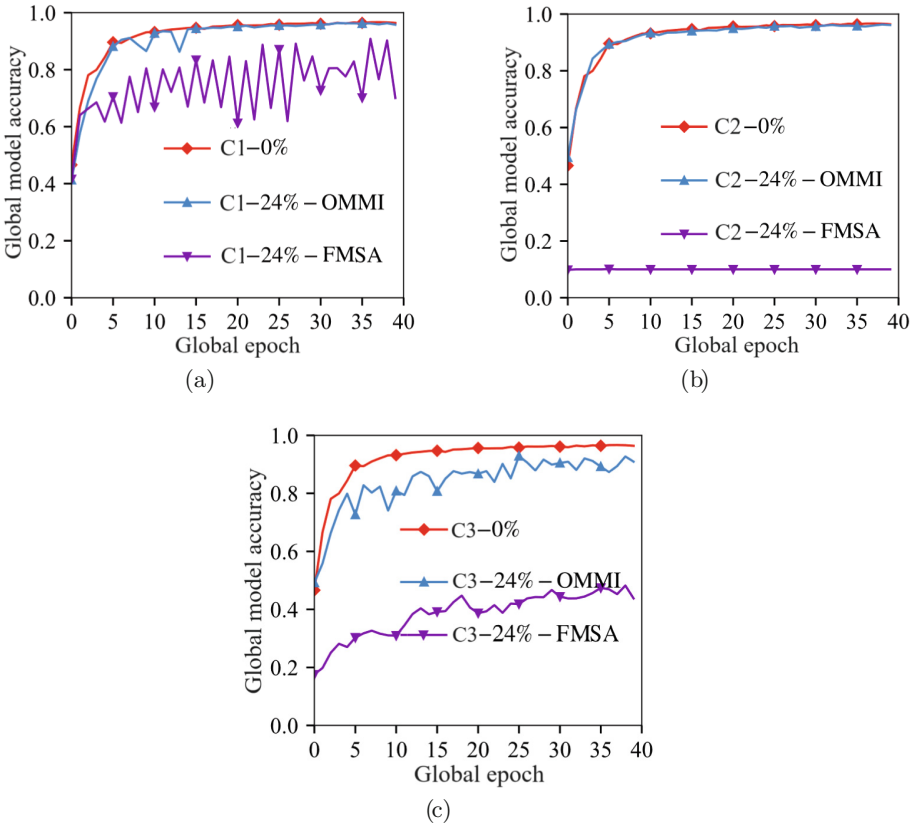

Fig. 4. Performance of the OMMI algorithm under three attack methods, where the proportion of malicious models is 24% or 0%. (a) C1: the label flipping attack; (b) C2: the byzantine attack; (c) C3: the model compression attack.

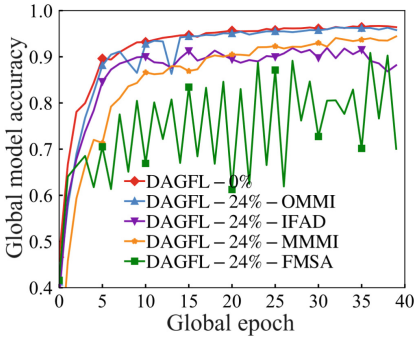

Fig. 5. Performance of different defense algorithms with 24% malicious IDs.

In Fig. 5, we illustrate the impact of different defense algorithms on the global model accuracy. The proposed OMMI algorithm outperforms others in Non-IID data scenarios, that is because it identifies malicious models by detecting outliers in the parameter distances. In contrast, due to substantial differences in the parameter distributions of normal local models, the MMMI algorithm performs worse. Moreover, the IFAD algorithm, a compromise method using the median of local model parameters, achieves higher global model accuracy compared to the FMSA algorithm.

The global model accuracy of the three different device selection algorithms with 24% malicious IDs is shown in Fig. 6. Specifically, the proposed RADDS algorithm achieves the highest accuracy by considering the reputation and activity degree of IDs, which effectively reduces the proportion of malicious models among the selected IDs. However, the ADS algorithm, which selects all malicious and normal IDs, leads to sharp fluctuations in accuracy. Overall, the OMMI algorithm and the RADDS algorithm can enhance the security and stability of FL.

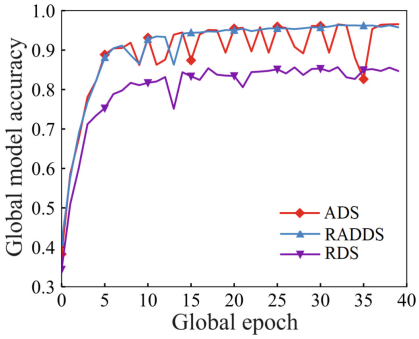

Fig. 6. Performance of different device selection algorithms with 24% malicious IDs.

To evaluate the performance of the proposed WFLA algorithm, we vary the proportion of IDs trained with Non-IID data and select the FedAvg algorithm for comparison, as shown in Fig. 7. Additionally, 10%, 50%, and 90% represent the proportions of IDs trained with Non-IID data. As this proportion gradually increases, the WFLA algorithm shows better global model convergence speed compared to the FedAvg algorithm. By evaluating the contribution of local model parameters to global aggregation and the accuracy of local models, the WFLA algorithm can effectively mitigate the negative impact of Non-IID data. Therefore, the proposed WFLA algorithm is more suitable for Non-IID data scenarios because it assigns larger aggregation weights to high-quality local models.

Figure 8 shows the performance of different FL algorithms. Among the compared algorithms, the proposed DAGFL algorithm demonstrates superior performance, as it combines the security and immutability of blockchain. However, the

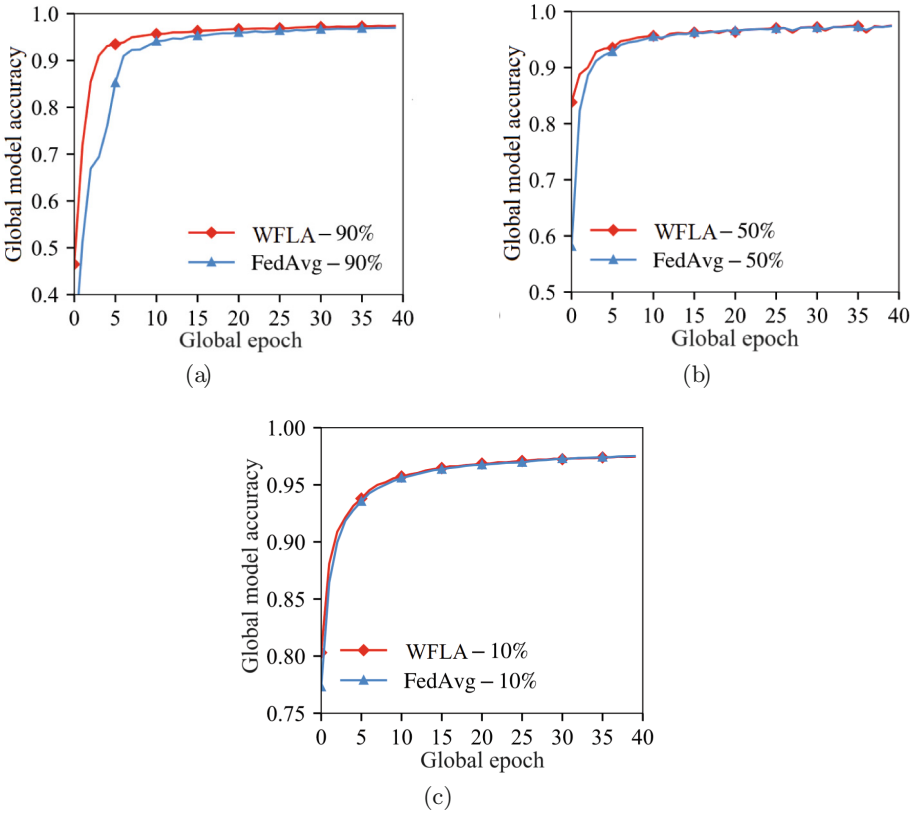

Fig. 7. Performance of different aggregation algorithms. (a) 90% of IDs are trained with Non-IID data; (b) 50% of IDs are trained with Non-IID data; (c) 10% of IDs are trained with Non-IID data.

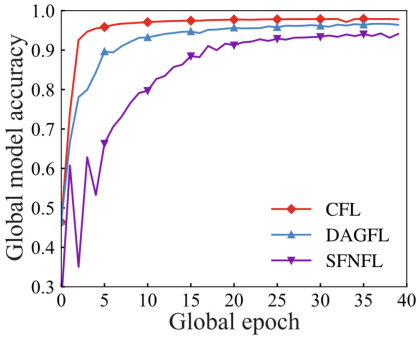

Fig. 8. Performance of different FL algorithms.

CFL algorithm, which collects all local models and centralizes global aggregation on a single server, faces the SPOF issue. In contrast, the DAGFL algorithm utilizes the DAG blockchain network to share the global model among FNs, enabling distributed learning and overcoming the SPOF issue.

5 Conclusion

In this paper, we focused on the DAG blockchain-enabled fog computing network for secure and effective FL. Firstly, we proposed a RADDS algorithm to reduce the participation of malicious IDs and introduced an OMMI algorithm to filter out malicious local models by detecting outliers in the parameter distances. Secondly, to mitigate the negative impact of Non-IID data, we proposed a WFLA algorithm that assigns weights to local models based on their contribution. Furthermore, we developed a DAGFL algorithm to enhance the security and training efficiency of the WFLA algorithm. Finally, the simulation results demonstrated the superiority of the proposed algorithms in Non-IID data scenarios.

References

1. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, pp. 93–104 (2000)
2. Feng, C., Liu, B., Yu, K., Goudos, S.K., Wan, S.: Blockchain-empowered decentralized horizontal federated learning for 5G-enabled UAVs. IEEE Trans. Ind. Inf. **18**(5), 3582–3592 (2022). https://doi.org/10.1109/TII.2021.3116132
3. He, Z., Yang, L., Lin, W., Wu, W.: Improving accuracy and convergence in group-based federated learning on Non-IID data. IEEE Trans. Netw. Sci. Eng. **10**(3), 1389–1404 (2023). https://doi.org/10.1109/TNSE.2022.3163279
4. Huang, Y., et al.: Personalized cross-silo federated learning on Non-IID data. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 7865–7873 (2021)
5. Kim, H., Park, J., Bennis, M., Kim, S.L.: On-device federated learning via blockchain and its latency analysis. arXiv preprint arXiv:1808.03949 (2018)
6. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics, pp. 1273–1282. PMLR (2017)
7. Połap, D., Srivastava, G., Yu, K.: Agent architecture of an intelligent medical system based on federated learning and blockchain technology. J. Inf. Secur. Appl. **58**, 102748 (2021)
8. Revathy, S., Priya, S.S.: Security enhanced federated learning approach using blockchain. In: 2022 International Conference on Computer, Power and Communications (ICCPC), pp. 50–55 (2022). https://doi.org/10.1109/ICCPC55978.2022.10072091
9. Tolpegin, V., Truex, S., Gursoy, M.E., Liu, L.: Data poisoning attacks against federated learning systems. In: Chen, L., Li, N., Liang, K., Schneider, S. (eds.) ESORICS 2020. LNCS, vol. 12308, pp. 480–501. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58951-6_24

10. Xiao, J., Du, C., Duan, Z., Guo, W.: A novel server-side aggregation strategy for federated learning in Non-IID situations. In: 2021 20th International Symposium on Parallel and Distributed Computing (ISPDC), pp. 17–24. IEEE (2021)
11. Yin, D., Chen, Y., Kannan, R., Bartlett, P.: Byzantine-robust distributed learning: towards optimal statistical rates. In: International Conference on Machine Learning, pp. 5650–5659. PMLR (2018)
12. Zhu, H., Xu, J., Liu, S., Jin, Y.: Federated learning on Non-IID data: a survey. Neurocomputing **465**, 371–390 (2021)