Hindawi
Security and Communication Networks
Volume 2021, Article ID 9965157, 14 pages
https://doi.org/10.1155/2021/9965157

Research Article

DAM-SE: A Blockchain-Based Optimized Solution for the Counterattacks in the Internet of Federated Learning Systems

Shichang Xuan, Ming Jin, Xin Li, Zhaoyuan Yao, Wu Yang, and Dapeng Man

Information Security Research Centre, Harbin Engineering University, Harbin 150001, China

Correspondence should be addressed to Dapeng Man; mandapeng@hrbeu.edu.cn

Received 26 March 2021; Accepted 21 June 2021; Published 1 July 2021

Academic Editor: Muhammad Shafiq

The rapid development in network technology has resulted in the proliferation of Internet of Things (IoT). This trend has led to a widespread utilization of decentralized data and distributed computing power. While machine learning can benefit from the massive amount of IoT data, privacy concerns and communication costs have caused data silos. Although the adoption of blockchain and federated learning technologies addresses the security issues related to collusion attacks and privacy leakage in data sharing, the "free-rider attacks" and "model poisoning attacks" in the federated learning process require auditing of the training models one by one. However, that increases the communication cost of the entire training process. Hence, to address the problem of increased communication cost due to node security verification in the blockchain-based federated learning process, we propose a communication cost optimization method based on security evaluation. By studying the verification mechanism for useless or malicious nodes, we also introduce a double-layer aggregation model into the federated learning process by combining the competing voting verification methods and aggregation algorithms. The experimental comparisons verify that the proposed model effectively reduces the communication cost of the node security verification in the blockchain-based federated learning process.

1. Introduction

With the continuous expansion of the scale of Internet of Things and the rapid development of artificial intelligence, the amount of data generated is growing at an unprecedented speed. If these resources distributed in the edge of the Internet of Things are effectively used, this will greatly improve the efficiency of machine learning and reduce the cost of machine learning. Therefore, how to use the data generated by Internet of Things devices efficiently becomes a hot topic. However, with the deepening of research, how to transfer the value of data under the premise of ensuring data security has become a key problem to be solved. On the one hand, the centralized management of the existing Internet of Things may lead to data centralization, which leads to the problem of single point failure and affects the smooth operation of the whole network. On the other hand, in the process of information transmission, due to the openness of wireless network, wireless signals between transmission devices are easily eavesdropping, interfering and shielding, and being attacked by DDoS and Sybil, which will affect the security of privacy data. In view of the above problems, the combination of blockchain and federated learning advantages and application in the field of Internet of Things can effectively improve the security of the Internet of Things.

Blockchain is an emerging and rapidly developing technology. Due to its wide commercial application, many research directions have been formed. Through the combination of various technologies, we have obtained unique advantages and capabilities and greatly expanded some application fields and functions. Blockchain can effectively integrate resources to form a perfect architecture and promote the development of machine learning technology [1]. Some examples include helping doctors make more accurate diagnosis [2] and providing more humanized and intelligent services for the Internet of Things [3]. At present, researches on federated learning based on blockchain are increasing. The main research direction in this field is to use blockchain to replace the central node in federated learning and use the decentralization of blockchain to overcome the privacy leakage and single point of failure problems caused by the node.

In traditional scenarios, enterprises or institution training models need to collect, store, and process a large amount of data. This means higher network, storage, and computing capabilities, as well as training costs. In the process of data transmission and sharing, the following challenges are encountered: (i) "Data island" phenomenon, (ii) stricter security regulations, and (iii) the data storage capacity that cannot meet the practical application requirements. To overcome these challenges, federated learning (FL) came into being. In FL process, the training node uses the local data set to train a local model. Then, the training node sends the parameters of the local model to the central coordinator, which aggregates multiple local models to get a global model. Finally, the training node downloads the parameters of the global model to update the local model and then trains on the local data set again, so that n rounds of iterations are carried out until the global model converges. FL can cooperate with or learn in depth on the Internet of Things edge network to ensure data security because the data set does not need to be migrated throughout the learning process. However, due to the complexity of the Internet of Things, different edge devices show different quality and stability in the learning process. This leads to different requirements for communication cost, privacy protection, and resource allocation, which increases a certain bottleneck for the wide spread of federated learning. Some researchers have proposed some schemes, such as verification process and algorithms, to overcome the "free-rider" and "model poisoning" attacks in the federated learning process. However, the current solution needs to verify and audit all local models one by one, which not only results in the waste of computing power but also increases the communication cost of the whole training process.

The main contribution of this paper is to compare existing verifiable federated learning frameworks based on blockchain and propose a double-layer aggregation model based on security evaluation to address its high communication cost. At the same time, the proposed incentive mechanism ensures that rational workers can gain the maximum benefit by remaining honest. The combination with the incentive model allows the proposed model to reduce the communication cost of training without relying on any heavy encryption and special hardware and to defend against poisoning attacks and free-rider attacks.

The rest of the paper is organized as follows. We discuss the existing works in more detail and compare them with our work in Section 2. The double-layer polymerization model based on safety evaluation is described in Section 3. Section 4 gives the conclusion.

2. Related Works

Federated learning can train the local model and aggregate the global model on the premise that the private data does not leave the local, to protect the security of the private data. However, there are still some defects in the model aggregation and the benign incentive of participating nodes in federated learning, which leads to a series of researches.

In terms of security of model aggregation, Fu et al. [4] proposed a privacy-protected verified federated learning (VFL) algorithm. This algorithm used Lagrange interpolation to carefully set the interpolation points verifying the correctness of the polymerization gradient, which preserved the safety of the aggregation model. If no more than $n-2$ of n participants collude with the aggregation server, VFL could guarantee the encrypted gradients of other participants not being inverted.

In decentralized federated learning, Kim et al. [5] proposed a blockchain federated learning (BlockFL) architecture. Since the local training results included a verification process, BlockFL overcame the single point of failure and expanded its federated scope to untrusted devices in public networks. Besides, it promoted the combination of more equipment with more training samples by providing rewards proportional to the number of training samples. Bao et al. [6] proposed a centered, public-audited, and healthy federated learning ecosystem called FLchain in trust and incentive. In FLchain, blockchain is used to replace the traditional federated learning central coordinator. With the tamper-resistant nature of blockchain, the behavior of participating nodes can be trusted, so that benign and malicious participating nodes can be identified and incentivized and punished accordingly. Majeed and Seon [7] proposed a new blockchain architecture. The concept of channels is used to learn multiple global models of this architecture. The local model parameters for each global iteration are stored as blocks in a channel-specific ledger. Zhang et al. [8] proposed a decentralized, collaborative privacy protection training method based on a multizone blockchain system for medical image analysis. This method allowed researchers from different medical institutions to collaborate when training machine learning models without sharing sensitive patient data.

In the area of security collaboration, Yin et al. [9] developed a federated learning-based security data collaborative (FDC) mechanism for handling the safety collaboration of multiparty data in the IoT environment. The blockchain was used to record the multipartial interaction content addressing the privacy and security issues of the IoT data. Federated learning was used to solve the problem of large-scale multiparty security collaboration in the IoT. Kim and Hong [10] et al. proposed a local learning weighting method based on node identification, and the experimental results show that the method proposed in this paper performs better in terms of learning speed and stability compared to the traditional federated learning. Martinez et al. [11] addresses the issues of data privacy, security, and fair rewards in distributed machine learning using blockchain and federated learning. An in-depth workflow, off-chain record database that can be used in conjunction with blockchain and an architecture for scalable recording and rewarding of gradients are proposed. Zhang et al. [12] proposed a blockchain-based federated learning method to detect equipment failures in the IoT. This method enables the authentication integrity of client data. To solve the data heterogeneity problem in failure detection, a novel centroid weighted joint average algorithm called CDWFEDAVG is proposed. Finally, to motivate customers to participate in the federated learning process, an incentive mechanism is designed based on the customer data used in local model training.

In terms of privacy protection, Lu et al. [13] design a secure data-sharing architecture supporting zone chains. By adopting federated learning, data sharing was expressed as a machine learning problem. They also showed that the data privacy could be protected by sharing data models rather than the actual data. Zhao et al. [14] designed a federated learning system using credit mechanisms. This system allows household appliance manufacturers to predict customer needs and consumption behaviors based on machine learning models trained using customer data. They also proposed a new standardization technology, which achieved a higher test accuracy than the bulk standardization while retaining the extraction characteristics privacy of each participant data. Besides, by using differential privacy, the opponent could be prevented from inferring the customer's sensitive information. Yu et al. [15] proposed a new privacy-preserving federated learning scheme. Based on the trusted execution environment (TEE), the training Integrity Protocol of the scheme was designed. The protocol can detect causal attacks and ensure the integrity of the deep learning process. Kim et al. [16] proposed a blockchain federated learning (BlockFL) scheme. This scheme uses the method of combining blockchain technology and federated learning to solve the problem that the central coordinator which the centralized federated learning system depends on is vulnerable to attack. Wang et al. [17] proposed a new secure decentralized multiparty learning system based on blockchain technology. The authors design two types of Byzantine attacks in the system and design secure off-chain sample mining and on-chain sample mining schemes to resist the attacks.

At present, there are not particularly many blockchain-based federated learning technologies and platforms, and most of the research in this field uses blockchain instead of central servers to ensure the security and accuracy of federated learning aggregation. Meanwhile, the incentive mechanism of blockchain can attract more nodes to participate in training. However, there is less research related to node security verification, which deserves in-depth study.

3. The Double-Layer Aggregation Model Based on Safety Evaluation

In this section, we introduce the two main models of the current blockchain-based verifiable federated learning framework and subsequently introduce the double-layer aggregation model proposed in this paper with its security evaluation index and finally present our comparative experiments and the analysis of the results.

3.1. Comparison of Verified Federated Learning Frameworks. There are currently two main models of blockchain-based verifiable federated learning frameworks: (i) the centralized verification model of the verification set and (ii) the distributed verification model of all submodels.

An example of the first model is the EOS blockchain of Martinez et al. [11], which proposed a new concept based on the data segmentation of the edge node customizing the verification data set called class sample verification error schemes (CSVES), as shown in Figure 1. This method is recommended to be used before the start of training. The collection of all available classes is defined as $C = \{C_1, C_2, \ldots, C_p\}$. For edge node $D \in D$, the article puts the collection C_D as a collection of all categories of data. Then, $C_D \subseteq C$, since there might be a class $C_i \in C$, making all local data points $d \in d$, where $d \notin C_i$. D sends a data set C_D to O and receives a validation set, C_D. The data set of the verification set C_D is only selected from the chain data set of O. Once the verification set is received, D starts with the local training data set d and the received verification set training model T_k. During the training, the model applies the verification set to the training model and records the verification errors. If the number of verification errors is reduced, the model is considered to be improved. At the end of the training, D sends the authentication error information and other parameters of the call function *UploadGradient ()*. Reference [11] improved on this feature to observe the overall trend of verification errors during model training. If the verification error is reduced, δ is a valuable and valid gradient update, and the model is rewarded based on its data cost n.

Toyoda et al. [18] proposed another competitive model update method called IABFL, as shown in Figure 2. IABFL was a low-cost approach achieving the expected goals in the event of reasonable action of participants. The main focus was on federated learning to introduce duplicate competition, which allowed the rational workers to follow the agreement and maximize their profits. Each worker selected in a particular round picked the top update model submitted by the last round of worker and used it to update their model.

EOS and IABFL are able to validate the child model updated by the edge node participating in the training to prevent useless or malicious models from joining to the global model. In the EOS, each training round requires participants to claim the validation data set from the training process on the blockchain based on the local data set.

The size of the validation data set is not as large as the local data set, but, in the overall framework, it requires the blockchain to send different validation sets to each edge node participating in the training. The number of edge nodes in a typical federation learning framework is measured in tens of thousands and multiplied by the total number of rounds in the whole training process, the communication consumption of the scheme will be very large, and the communication cost of training a completed model will be tens or even hundreds of times higher than that of unverified federation learning.

In the IABFL, the submodel of each node is transmitted between blockchains as the communication data during the verification process. The amount of submodel data is not voluminous compared to the verification data set; hence, the communication cost of the program will be much lower than that of the EOS.

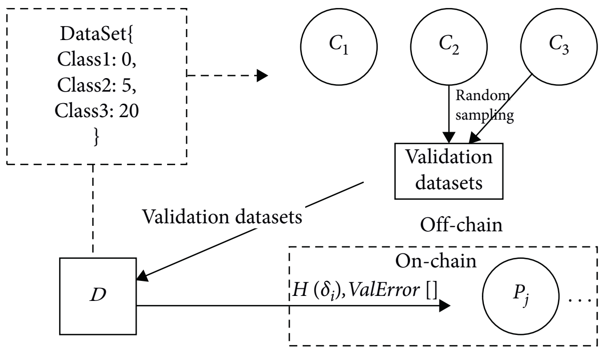

FIGURE 1: Class sample verification error schemes.

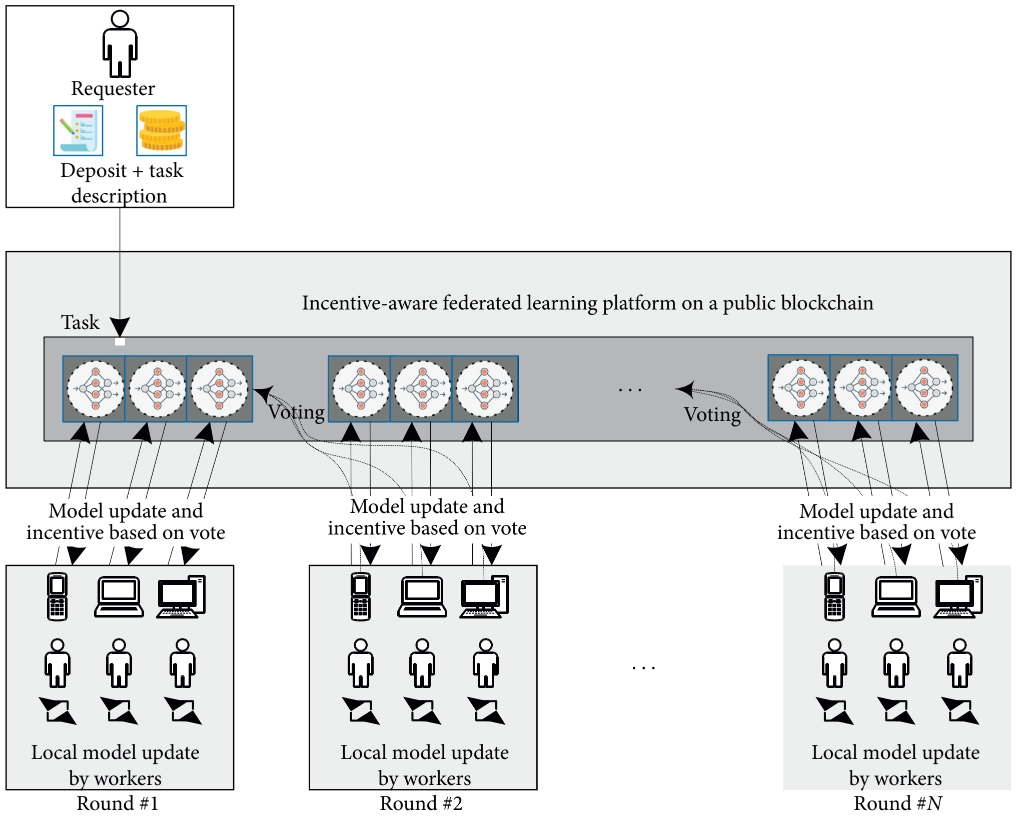
Requester
Deposit + task description
Incentive-aware federated learning platform on a public blockchain
Task
Voting
Voting
...
Model update and incentive based on vote
Model update and incentive based on vote
Model update and incentive based on vote
Local model update by workers
Round #1
Local model update by workers
Round #2
...
Local model update by workers
Round #N

FIGURE 2: Class sample verification error schemes.

In this model, it is assumed that n worker members are involved in training each round, and the traffic created by transmitting a single model parameter in the network is t. First, n worker members submit a local model to the block and synchronize the child model of all other nodes on the chain. This process requires a data traffic of C_1, which can be expressed as

$$C_1 = n \times (n-1) \times t. \tag{1}$$

Assuming that the amount of data consumed by a single vote is 1, n worker participating in training completes voting on the chain and synchronizes the data traffic that needs C_2; that is,

$$C_2 = n \times (n-1). \tag{2}$$

The data traffic created by n worker in the IABFL is denoted as C_{pre}, which can be calculated as

$$C_{\text{pre}} = C_1 + C_2 = n \times (n-1) \times t + n \times (n-1). \tag{3}$$

Through the analysis of communication cost, it can be found that the current work has solved the problem of security verification of nodes relatively well, but it brings a significant increase in communication cost, and the effect is doubtful in practical application, so this chapter wants to propose a corresponding solution to the problem of high communication cost.

3.2. The Double-Layer Polymerization Model. In the traditional blockchain-based federated learning process, training is vulnerable to model poisoning attacks or free-riding attacks. The current verifiable federated learning will add a large amount of data communication on the basis of the original training to verify the quality of nodes or data in various ways to solve the above problems. However, the sharp increase in communication costs will reduce the willingness of each node to participate, so we propose a step-by-step federated learning platform based on blockchain with a verification mechanism design.

The key idea behind our platform is to introduce repeated models to update the competition design, and through incentive mechanisms any rational worker can work hard and abide by the agreement to maximize their profits. The proposed design will naturally enable rational workers to act honestly without any heavy encryption and special hardware. As shown in Figure 3, in a specific round, all the nodes participating in the training are divided into multiple small clusters. The submodels of several nodes in a small cluster are first partially aggregated into a step-by-step model, and each child node will select the upper A round of the best k model updates submitted by a small cluster and update its own model based on these updates. The reason is that the reward for workers in the previous round depends on the results of the voting. The motivation for choosing the model with the best k models is that its model updates will have more chances to be voted in the next round, which means that they will get more return. The workers in the next round still cannot be destroyed, because their models are also competed and voted on by the workers in the next round. In the following, we will discuss the system model and the mechanism process in detail.

The symbols used later are described in Table 1.

There are four roles in the system: administrator, requester, worker, and consensus node. Table 2 lists the role information of each participant. The role of the administrator is to deploy a series of smart contracts [19] on a public blockchain, such as Ethereum [20], and to register requesters and workers to the platform upon request. It is assumed that the participants know how to access the location of the smart contracts through a forum or website. The requester may have neither the data for training nor the equipment for training deep learning models. The worker, on the other hand, needs to have both data for training and equipment for training the deep learning model. Any type of data can be processed on this platform, such as images, text, and audio. Enter the data set of worker i for task t. Subscript t is omitted because the next are for a specific task t. It is assumed that the data sets owned by workers for a specific task are independent and homogeneously distributed. This assumption is natural because a requester submitting a model for a specific task, such as a deep learning model for identifying cats in pictures, would require that only workers with data sets specific to that task can join.

The platform consists of seven procedures: user registration, task release, task joining, task start, model update, reward assignment, and task completion. In this article, Ethereum is used as the blockchain, which is one of the most popular cryptocurrencies that support intelligent contracts. However, the proposed model is applicable to any other technology with intelligent contract support.

3.2.1. User Registration. Administrators need to register all participating users on the platform based on their requests. Each user must share their Ethereum address with the administrator for receiving tasks and rewards, besides declaring whether to register as a requester or worker. After the registration is completed, the requester can release the FL training task on the blockchain, and the worker can join the task to update the model and get the corresponding reward.

3.2.2. Task Release. Any user registered as a requestor can issue federated learning training tasks through a smart contract. To do that, the requester must specify the following:

(1) Model description, for example, loss functions, data formats, learning rates, layers, unit numbers, and activation functions
(2) The parameters, for example, the training period, safety evaluation index, start time, the number of workers, and the total rewards
(3) Deposits of the total rewards, D, which are equal to $r \times N$

3.2.3. Task Joining. After the requester posts the task, event notifications will be sent to all registered workers via the Ethereum's event processing function. Each worker will then decide whether to participate in that task. If the worker decides to join, the intelligent contract should be called before the task begins. Based on the requirements, the intelligent contract can only be called when the caller is registered as a worker; otherwise, the abort code is executed. From the perspective of code implementation, the EMI address of a worker is stored in an array.

3.2.4. Task Starts. After the task application period, the requester enrolls in the group of workers W_t joining the task t. Then, they select the number of the model updates N and the number of worker members participating in each round. However, the requester should not disclose in advance to the worker the number of rounds N to be used for model updates. It is necessary to show the worker at the end of the N-wheels. The cause of this will be detailed later. Besides, the requester needs to introduce the federated learning model parameters into ω_0 and submits them into the blockchain. The requester can use any algorithm to initialize the model [21].

3.2.5. Model Update. After model training, each round of worker k is randomly selected among all members registered with the intelligent contract. Then, the number of individuals in each cluster is calculated according to the safety evaluation algorithm, which will be detailed in the next sections. Each worker gets the local aggregation model parameters of the previous round of each cluster from the blockchain and verifies and votes them. Then, they calculate the global model for the model update based on the selected top model. Finally, each worker is trained based on the local

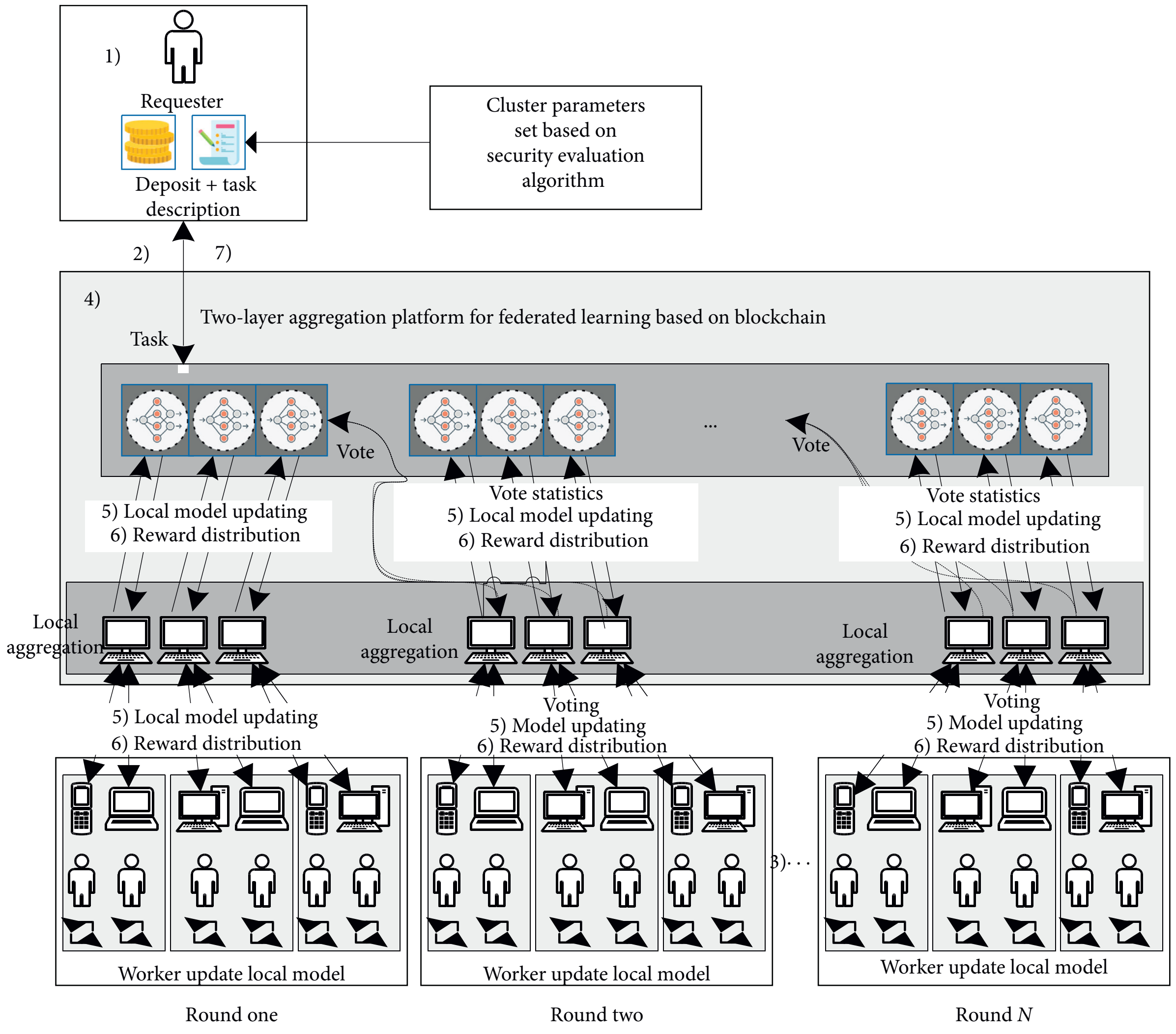

FIGURE 3: The double-layer FL aggregation platform based on blockchains.

TABLE 1: Symbols table.

Symbol	Paraphrase
W	The entire collection of workers
i	Worker's index
e	Index of each round
a	The number of top model updates selected by the staff in the next round
K	Total number of workers participating in task t
C	Percentage of workers per round
k	Number of workers selected in each round, $k = \max(K \cdot C, 1)$
d_i	Data set owned by worker i in round e
B	Batch size in deep learning
η	Learning rate in deep learning
s	The security value
g	Number of clusters in federated learning
$\tau(\cdot)$	Loss function, such as the mean square error in deep learning
$\text{split}(d, B)$	Function to randomly divide data set d into batch B

data set to derive the submodel for this round and submits it to the blockchain consensus node of the cluster. The consensus node locally aggregates the submodels of all workers of the cluster according to the average aggregation algorithm to obtain the local model parameters of the cluster and submits them to the chain together with the voting results of each node. The algorithm for model update is shown as Algorithm 1.

In lines 1–6 of the algorithm, the main task is to select the best top models, a, for voting and provide them for use in subsequent model aggregations. This is done as follows: if it is not the first round at the beginning of the training task, each worker uses the local data set to validate the local aggregation models of the g clusters from the previous round and selects a model that they consider the best for voting; otherwise, this step is skipped. In lines 7–11 of the algorithm, the role is to compute the underlying global model for this training round. This is done by using the initialization parameter ω_0 provided by the requester as the

Table 2: The role of each participant.

Participant	Task	Availability of data	Availability of equipment for model training
Administrator	Deploy smart contracts on the public blockchain and register requesters and workers to smart contracts upon request	—	—
Requester	Submit a training task to obtain a trained model	Unnecessary	Unnecessary
Worker	Train the task model submitted by the requester for the reward	Yes	Yes
Consensus node	Local aggregation, synchronization of blockchain information, and distribution of rewards to submodels in the cluster	Unnecessary	Unnecessary

```
Input: model updates submitted by all clusters in round $e-1$, $\{\omega_{i,e-1}\}_{i\in g}$ (when $e\geq 2$) or $\omega_0$ (when $e=1$)
Output: trained model parameters $\omega_i$, Voting results $M_{i,e}$
/*1. Use local data to select the best $a$ model to update (except the first round)*/
(1) if $e\geq 2$ then
(2)     for $m\in g$ do
(3)         $l_m = 1/|d_i|\sum_{j\in d_i}\tau(X_j, y_j; \omega_{m,e-1})$
(4)     end
(5)     $M_{i,e} \longleftarrow$ chose $a$ models whose $l_m$ is the smallest.
(6) end
/*2. Aggregation with $a$ models value */
(7) if $e == 1$ then
(8)     $\omega'_{i,e} \longleftarrow \omega_0$
(9) else
(10)    $\omega'_{i,e} \longleftarrow 1/a\sum_{m\in M_{i,e}}\omega_{m,e-1}$
(11) end
/* Update the model with local data */
(12) $\beta \longleftarrow$ split$(d_i, B)$
(13) $\omega_{i,e} \longleftarrow \omega'_{i,e}$
(14) for each local epochs $E$ do
(15)     for each batch $b$ in $\beta$ do
(16)         $\omega_{i,e} \longleftarrow \omega_{i,e} - \eta\nabla\tau(\omega_{i,e}, b)$
(17)     end
(18) end
(19) return $\omega_{i,e}$, $M_{i,e}$
```

Algorithm 1: Model update of worker i in round e.

parameter of the global model if it is the first round at the beginning of the training task; otherwise the a local models selected in lines 1–6 of the algorithm are averaged and aggregated to obtain the global model. In lines 12–18 of the algorithm, each worker performs local model training using the local data set based on the global model computed in lines 7–11 of the algorithm to train the submodel parameters for that round. It is worth noting that there are E rounds of local training, and the data set in each round is not the entire local data set, but the data set is first randomly divided into b batches before training, and each round of local training uses one of these batches.

3.2.6. Reward Assignment.

3.2.6. Reward Assignment. As shown in the algorithm in the model update, in the submission phase of the model update, each worker in round e votes for the first g local models (only one vote can be cast for a model). Based on the combined votes, the smart contract calculates the number of votes received by each cluster in round $e-1$. Based on the result of the number of votes, the reward is assigned to each cluster by $r_1 \geq r_2 \geq \cdots \geq r_k \geq 0$. Thus, the cluster with the most votes receives a reward of r_1, the cluster with the second most votes receives a reward of r_2, and so on. Each cluster distributes the profit according to the amount of data involved in training by the child nodes in each cluster based on the respective rewards obtained.

The total reward of each round is set to r, and the profit relationship between each cluster is given as

$$\sum_{j\in[1,k]} r_j = r. \quad (4)$$

3.2.7. Task Completion. After model update and reward assignments are repeated $N-1$ times, since there is no training task as well as workers in the next round, it is not possible to vote on the model updates completed by the workers in the last training round N. Therefore, the rewards from the last round of tasks are equally distributed to all workers involved in the training. However, this may negatively affect the working of the worker, because if the worker knows that they are selected to participate in the last round, they can get a reward only if they send one of the previous model updates. If that happens, the motivation of the first few workers will reduce, since their model updates may not receive the correct vote in the next round. Therefore, to ensure effective model training, the worker must not know whether they are in the last round of the training. So, the requester needs to reveal N to all workers at the end of the Nth round. [22]. In order not to let the worker guess N value from the remaining deposits, the requester needs to provide a deposit D larger than the actual total reward $N \times r$ before task initialization. Furthermore, after the worker completes all tasks, the requester asks them to return their excess deposits.

3.3. Safety Evaluation Index. Assuming that there are n training workers and the consensus nodes g connect to the blockchain in each round, the amount of communication consumed to transmit a single model parameter in the network is t. Firstly, the training worker members obtain the model parameters from the blockchain for the local aggregation of each cluster in the previous round. This process requires C_1 data traffic, which is calculated as

$$C_1 = n \times g \times t. \tag{5}$$

Each worker then updates the local model and submits the trained model parameters and its own voting results to the consensus node of the cluster it belongs to, and this process needs C_2 data traffic, which is calculated as follows:

$$C_2 = n \times (t + 1). \tag{6}$$

Finally, the consensus node g aggregates the child node model in the cluster into a local model. The local model and voting results of the cluster submitted to the block synchronize the local model of all other consensus nodes on the chain. This process requires C_3 data traffic [23] as follows:

$$C_3 = g \times (g - 1) \times (t + 1). \tag{7}$$

In this article, the data traffic consumed by n training workers in the model architecture C_{step} is calculated as

$$C_{\text{step}} = \sum_{j=1}^{3} C_j = n \times g \times t + n \times (t + 1) + g \times (g - 1) \times (t + 1). \tag{8}$$

According to the previous communication cost analysis, it is known that the amount of data traffic in the IABFL scheme is C_{pre}. Here are some definitions:

C_{save} is the part where the data traffic of single-round training of the proposed algorithm is less than that of IABFL, which is shown as

$$\begin{aligned} C_{\text{save}} = C_{\text{pre}} - C_{\text{mod}} &= n^2 t - nt + n^2 - n - bnt \\ &- b^2 t + bt + n - b^2 + b. \end{aligned} \tag{9}$$

R_{save} is the saving coefficient, that is, the ratio of the saved data traffic to the traffic consumed when transmitting a single model:

$$R_{\text{save}} = \frac{C_{\text{save}}}{t} \cong n^2 - b^2 + (1 - n)b - 2n. \tag{10}$$

Because the saving factor only indicates that the data traffic of a single-round training of the improved model is better than that of the original model if it is greater than 0, so $R_{\text{save}} > 0$ and the minimum number of consensus nodes g equals two; this leads to the following equation:

$$g \in \left[2, \frac{\sqrt{5n^2 - 10n + 1} - n + 1}{2}\right), \tag{11}$$

where n is an integer $(n \geq 5)$, so

$$\frac{\sqrt{5n^2 - 10n + 1} - n + 1}{2} \geq 2. \tag{12}$$

According to the calculation, the algorithm has an impact only when the number of participants in the federated learning is greater than or equal to five. The next derivation is discussed based on this result.

The security value of the improved model, C_{safe}, has two components: (i) the security of the blockchain and (ii) the security of the edge node. The principle of the blockchain states that the more the consensus nodes, the stronger the various attacks and the more secure the entire blockchain. The security value of the blockchain is denoted as g. There are more edge node workers in a single cluster for larger n/g values, which would "dilute" the polymerization influence degree of a single node of the local model. Besides, the safety value of the model is negatively correlated with n/g. Hence, g/n is taken as the security value of the edge node, which can be calculated as

$$C_{\text{safe}} = (s + 1) \times \left(g^m + (g/n)\right), \tag{13}$$

where $s \in [0, 10]$ is the security value.

The safety factor of the model proposed in this article is denoted as R_{safe}. According to the principle of blockchain, the more consensus nodes, the stronger the ability to resist various attacks, such as the Byzantine [24], and the more secure the whole blockchain; therefore, the more consensus nodes the model has, the lower the proportion of security value will be got. Hence, R_{safe} is negatively correlated with the size of g; that is,

$$R_{\text{safe}} = \frac{C_{\text{save}}}{g} \cong (s + 1) \times g^{m-1}. \tag{14}$$

Next, the value of m and its optimal value in the actual application are analyzed by modeling. The overall equalization coefficient of this model ϑ is defined as follows:

$$\vartheta = \frac{R_{\text{save}}}{R_{\text{safe}}}. \tag{15}$$

For $m = 2$, ϑ can be calculated as

$$\vartheta = \frac{R_{\text{save}}}{R_{\text{safe}}} = \frac{n^2 - 2n}{(s + 1)g} - \frac{g}{s + 1} + \frac{1 - n}{s + 1}. \tag{16}$$

When s is constant, the values of ϑ and $1/\vartheta$ are shown in Figures 4 and 5, respectively.

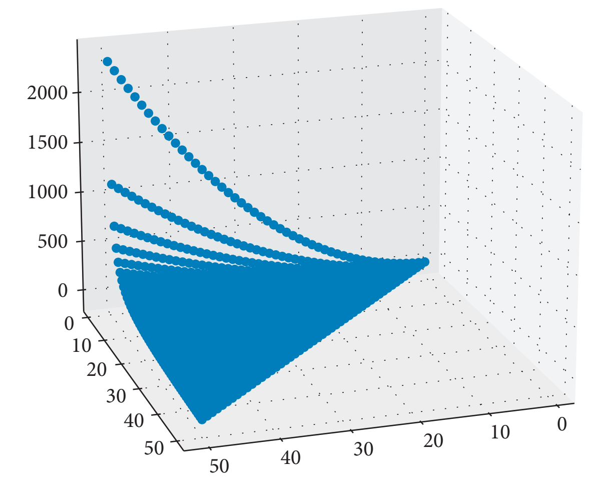

Figure 4: Value of ϑ for n and g, when $m = 2$.

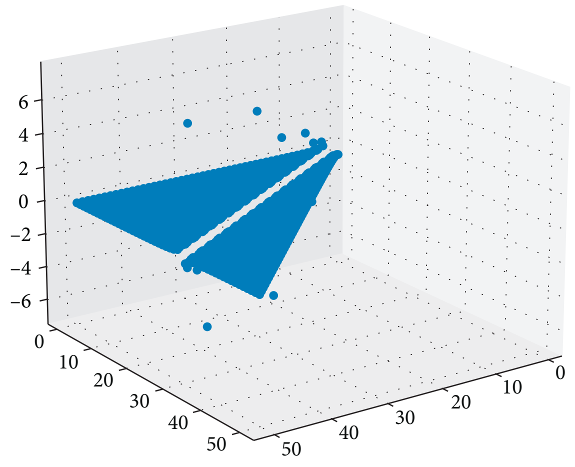

Figure 5: Value of $1/\vartheta$ for n and g, when $m = 2$.

When $m = 3$, ϑ is calculated as

$$\vartheta = \frac{R_{\text{save}}}{R_{\text{safe}}} = \frac{n^2 - g^2 + (1-n)g - 2n}{(s+1)g^2}. \tag{17}$$

When s is constant, the values of ϑ and $1/\vartheta$ are shown in Figures 6 and 7, respectively.

When $m=4$, $1/\vartheta$ is calculated as

$$\frac{1}{\vartheta} = \frac{R_{\text{safe}}}{R_{\text{save}}} = \frac{(s+1)g^3}{n^2 - g^2 + (1-n)g - 2n}. \tag{18}$$

When s is constant, the values of ϑ and $1/\vartheta$ are shown in Figures 8 and 9, respectively.

When $m = 2$, Figures 4 and 5 show that the proportion of R_{save} in the balance coefficient is super linear growth with n, which does not meet the actual situation and expectations. When $m = 4$, Figures 8 and 9 show that the proportion of R_{save} in the balance coefficient is super linear growth with g. Similarly, when $m \geq 4$, it is more incompatible. When $m = 3$, Figures 6 and 7 show that the balance coefficient trend is relatively stable with the relationship between n and g, which conforms to the actual situation and expectation and then further verification.

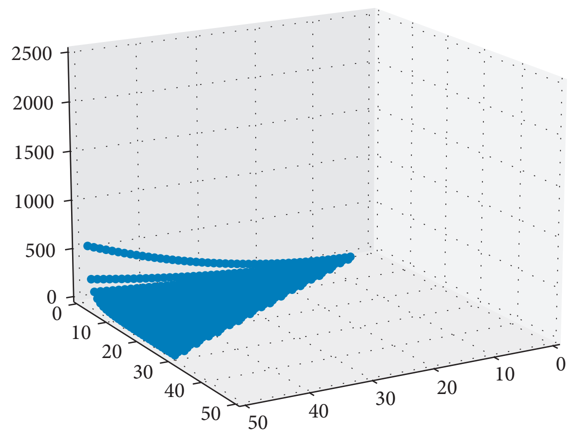

Figure 6: Value of ϑ for n and g, when $m = 3$.

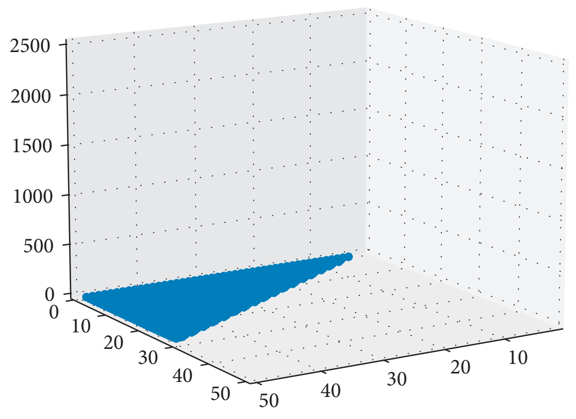

Figure 7: Value of $1/\vartheta$ for n and g, when $m = 3$.

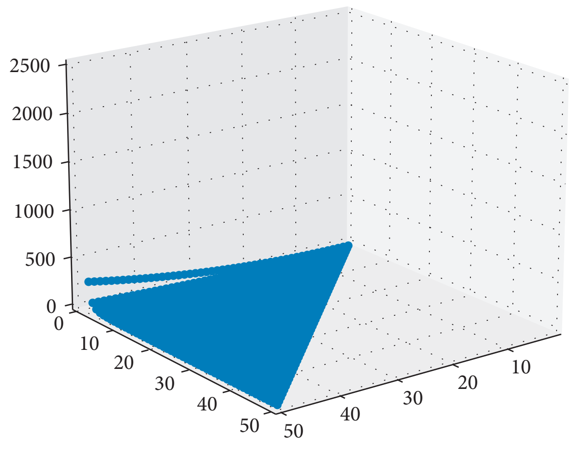

Figure 8: Value of ϑ for n and g, when $m = 4$.

When $m=3$, the number of the edge nodes in a single cluster, n/g, is taken as a variable, and its relationship with the balance coefficient is shown in Figure 10.

Figure 10 shows that the proportion of R_{save} in the balance coefficient increases linearly with the number of edge nodes in a single cluster, n/g. When $g > 5$, the impact of g on the overall trend is almost invisible. Also, it is intuitively clear from the formula that the equilibrium factor ϑ is

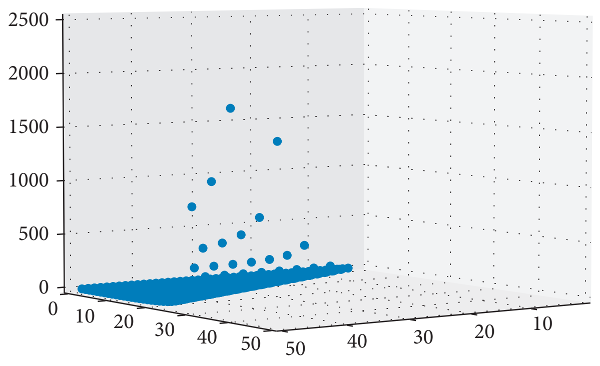

Figure 9: Value of $1/\vartheta$ for n and g, when $m = 4$.

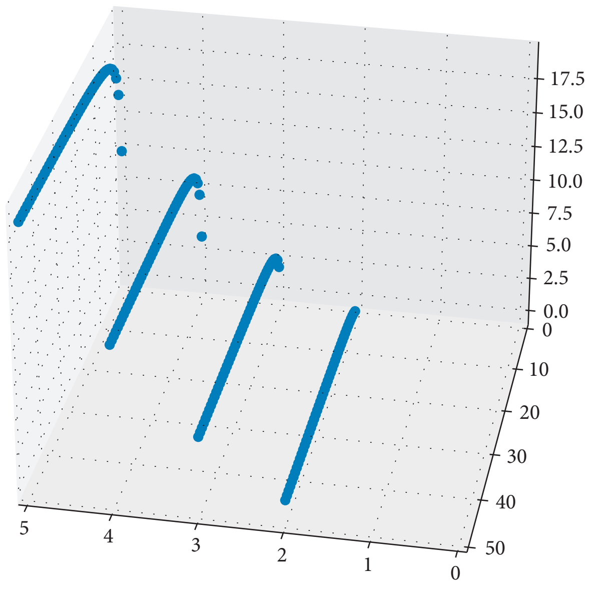

Figure 10: Value of ϑ for n/g and g when $m = 3$.

inversely proportional to the safety value s. It is concluded that when $m = 3$, the balance coefficient ϑ complies with the actual situation and the expectation of the model. Thus, the balance coefficient ϑ is determined as

$$\vartheta = \frac{R_{\text{save}}}{R_{\text{safe}}} = \frac{n^2 - g^2 + (1-n)g - 2n}{(s+1)g^2}. \tag{19}$$

As defined, when $\vartheta = 1$. The model focuses on saving the communication cost of the whole federated learning architecture when $\vartheta = 1$, the model is in a relative balance between saving communication consumption and model security. The model focuses on the safety of the entire federated learning architecture when $\vartheta < 1$. The specific value can be personalized by the requester during the training task, and this paper is recommended to take the balance state $\vartheta = 1$.

When $\vartheta = 1$, $R_{\text{save}} = R_{\text{safe}}$ shown as follows:

$$R_{\text{save}} - R_{\text{safe}} \cong n^2 - (s+2) \times g^2 - n \times g = 0, \tag{20}$$

where g is

$$g = \frac{\sqrt{4 \times s + 9} - 1}{2 \times s + 4} \times n. \tag{21}$$

Using (21), the relational Table 3 is generated.

The table shows that even if the value of security is minimized, it can save approximately 25% of the communication costs. When the security value $s \geq 2.79$, it can save about 50% of the communication costs.

3.4. Feasibility Analysis of Double-Layer Aggregation Model.

The research indicates that the core step in the FedAvg algorithm is the parameter aggregation of each submodel, which can be formulated as

$$\omega_{t+1} \longleftarrow \sum_{k=1}^{K} \frac{n_k}{n} \omega_{t+1}^{k}. \tag{22}$$

In (22), the global parameter is obtained by accumulating k subparameters based on the weight of the sample data size. The total information of each subparameter consists of two parts: (i) ω_{t+1}^{k}, the value of the submodel's current parameter, and (ii) n_k, the number of samples trained with the model parameter (in federated learning, it often refers to an edge node participating in training the amount of data). From another perspective, if a subparameter contains the value of the submodel's current parameter and the number of samples from which the model parameter is trained, then the subparameter can be regarded as produced by the same training sample.

Therefore, according to the combined rate of addition, it can be initially obtained that the result of the global parameter can be obtained by weighting the parameters of the two submodels, as shown in the following equation:

$$\omega \longleftarrow \frac{n_1}{n_1 + n_2} \omega^1 + \frac{n_2}{n_1 + n_2} \omega^2. \tag{23}$$

Now, let us expand the added submodel into i $(1 < i < k)$ parts. First, the k_i training samples are aggregated according to FedAvg to obtain the global parameter ω_i as

$$\omega \longleftarrow \sum_{k=1}^{K} \frac{n_k}{n} \omega_k, \tag{24}$$

where n_k are samples contained in each training sample. Then, the k training samples are randomly divided into i $(1 < i < k)$ parts. The i part has a total of k_i training samples, and each training sample is aggregated according to FedAvg to obtain the local parameter ω_i. At this time, ω_i is equivalent to a training sample of $i \times n_k$ samples. Then, i training samples ω_i are aggregated according to FedAvg to obtain the global parameter ω_α as follows:

$$\omega_\alpha \longleftarrow \sum_{i=1}^{I} \frac{n_i}{n_\alpha} \omega_i, \tag{25}$$

where n_α is

$$n_\alpha = \sum_{i=1}^{I} n_i. \tag{26}$$

Table 3: Communication volume reduction ratio table.

Value of security value, s	The average number of edge nodes in each cluster, n/g	The proportion of reduction in communication volume (%), $C_{\text{save}}/C_{\text{pre}}$
0	2.00	25
1	2.30	37
2	2.56	45
3	2.79	51
4	3.00	55
5	3.19	59
6	3.37	62
7	3.54	64
8	3.70	66
9	3.85	67
10	4.00	68

Finally, the following equation is gathered:

$$\begin{aligned}\omega &\longleftarrow \sum_{k=1}^{K} \frac{n_\alpha}{n}\omega_\alpha \\ &\longleftarrow \sum_{k=1}^{K} \frac{n_\alpha}{n} \times \frac{n_1 \times \omega_1 + \cdots + n_i \times \omega_i}{n_\alpha} \\ &\longleftarrow \sum_{k=1}^{K} \frac{n_k}{n}\omega_k.\end{aligned} \tag{27}$$

The final results of ω and ω_α demonstrate that they are equivalent. Thus, it can be concluded that the double-layer aggregation does not affect the final result of the global model parameters. The same applies to the IABFL, where the following equation expresses the average aggregation based on the number of nodes:

$$\omega_{i,e}' \longleftarrow \frac{1}{k} \sum_{m \in M_{i,e}} \omega_{m,e-1}. \tag{28}$$

3.5. Experiments and Result Analysis

3.5.1. Experiments. The federated learning process of the IABFL and the double-layer aggregation model is simulated through multiple virtual nodes. The aggregation effect, aggregation speed, and the training communication volume between the two models are compared. The results revealed the optimization effect in the availability and communication cost of the double-layer aggregation model.

The experiments adopt the controlling variables method. Experiment 1 is to compare the accuracy of the two models, while Experiment 2 compares them in terms of aggregation time. Experiment 3 evaluates the required network/data traffic, that is, communication cost, when the two models are aggregated. The experiments use the MINIST training set as the local data set, and 20% of the same validation set is used as the test set.

In the experiments, we simulated a small-scale federated learning environment, where the number of participating nodes is set to 50, the data set size of each node to 1000, and the number of local iterations of single-round training to 10. As mentioned, each iteration used 20% of the local data. We recorded the accuracy, aggregation time, and network traffic after each round of the training and averaged the values after ten times of running.

The experimental environment is given in Table 4.

Table 4: Experimental testbed specifications.

Environment	Model/version
System	64-bit Windows 10 Professional operating system
CPU	Intel® Core™ i7-7700 HQ@2.80GHz
GPU	NVIDIA GeForce GTX 1050
Software	Anaconda3
Python	Python = 3.7
Dependency	PyTorch, PySyft

3.5.2. Analysis of the Results

Experiment 1. Comparison of the aggregation effects

The two aggregation models are tested under the same conditions. Figure 11 illustrates that although the accuracy of the IABFL in the first three rounds is not as good as the double-layer aggregation, it becomes slightly better in the following rounds. Hence, the final difference in accuracy between the two models is very small. Considering the amount of data used, the accuracy that the models achieved is about 84%, which is quite good. This experiment validates that the double-layer aggregation model using the safety evaluation method proposed in this paper is applicable.

Experiment 2. Comparison of the polymerization speeds

Figure 12 depicts that the double-layer aggregation model consumes more time than the IABFL. According to the seven rounds repeated ten times, the average time of each round increases by 0.206 seconds. Analyzing the system process shows that the increased time is mainly used for the local aggregation of consensus nodes and the time consumed by one more segment of network communication. However, the increased time only accounts for 2.53% of the total model training, and it does not affect the training timeliness. This experiment, therefore, validates that the polymerization aging of the double-layer aggregation model proposed in this paper is applicable.

Experiment 3. Comparison of communication costs

Figure 13 shows that the communication cost of the double-layer aggregation model gradually decreases with the increase of the safety value. Hence, the security value s should be set in [0, 4] if there is no special requirement. It can also be seen that although the traffic trend of the double-layer aggregation model is the same as the previous theoretical analysis, there is always a small gap between the two, which is constant. This difference might be due to several factors, such as the network protocol header, various verification packets (e.g., the three-way handshake packet), and network fluctuations. However, the model is very close to the theoretical value, and the difference does not affect the actual use. Therefore, the model proposed in this paper can still reduce a large amount of data traffic and hence the

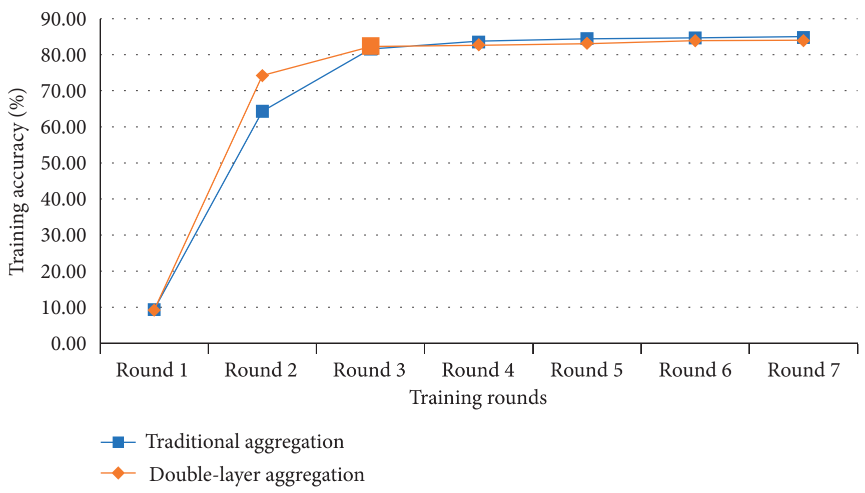

Figure 11: Training round versus accuracy of the two aggregation models.

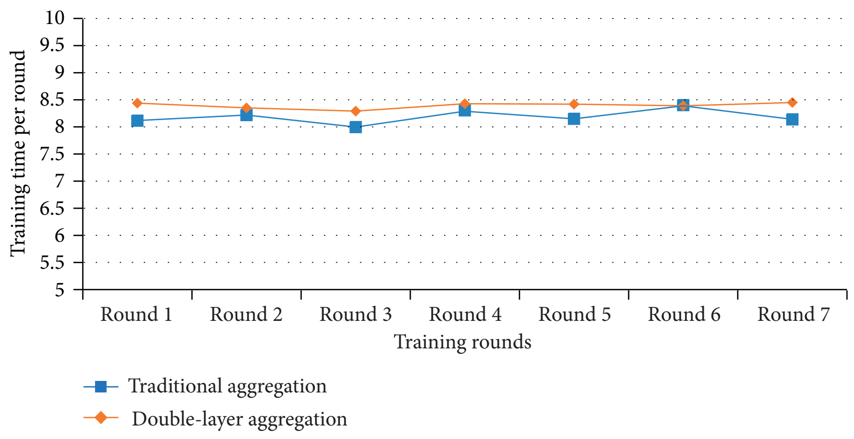

Figure 12: Training round versus time of the two aggregation models.

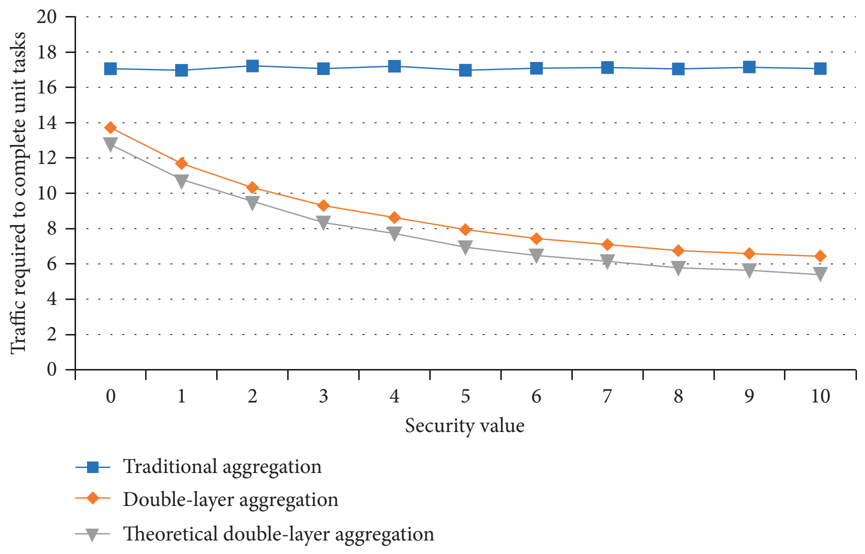

Figure 13: Security level versus communication volume.

communication cost during model training, which meets the design goals.

The above analysis of the experimental results concludes that the blockchain-based communication cost optimization and safety evaluation methods proposed can reduce the communication cost of the learning process while ensuring the security and availability of the nodes.

4. Conclusions

For the existing protection methods of “free-riding attacks” and “model poisoning attack” in the federated learning process, the training models need to be audited one by one, causing the problem of high communication costs throughout the training process. A step-by-step aggregation federated learning platform based on blockchain is proposed, and the architecture and algorithm of the platform are designed in detail. In this article, we first studied the verification mechanism of useless or malicious nodes. Then, using a competitive voting verification algorithm, a blockchain federation learning communication cost optimization method based on security evaluation is proposed, and the security of the model is analyzed in combination with game theory methods. Finally, through experimental comparison and analysis, we verify that the proposed method can reduce the communication cost of the learning process while ensuring the security and availability of IoT nodes.

Data Availability

The data used to support the findings of this study are included within this article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (grant nos. 61802086 and U20B2048), Defence Industrial Technology Development Program (grant no. JCKY2020604B004), and the Fundamental Research Funds for the Central Universities (grant nos. 3072021CF0608 and 3072020CF0601).

References

[1] H. Sun, Z. Liu, G. Wang, W. Lian, and J. Ma, “Intelligent analysis of medical big data based on deep learning,” *IEEE Access*, vol. 7, pp. 142022–142037, 2019.

[2] S. Santiago, G. Boris, R. Eduardo, M. T. Paul, A. Andre, and L. Marco, “Federated learning in distributed medical databases: meta-analysis of large-scale subcortical brain data,” in *Proceedings of the 16th IEEE International Symposium on Biomedical Imaging*, pp. 270–274, IEEE, Venice, Italy, April 2019.

[3] Q. Wu, K. He, and X. Chen, “Personalized federated learning for intelligent IoT applications: a cloud-edge based framework,” *IEEE Open Journal of the Computer Society*, vol. 1, pp. 35–44, 2020.

[4] A.-M Fu, X.-L. Zhang, N.-X. Xiong, Y. Gao, H. Wang, and J. Zhang, “VFL: a verifiable federated learning with privacy-preserving for big data in industrial IoT,” *IEEE Transactions on Industrial Informatics*, 2020.

[5] H. Kim, P. J. Hong, B. Mehdi, and K. Seong-Lyun, “Blockchained on-device federated learning,” *IEEE Communications Letters*, vol. 24, no. 6, pp. 1279–1283, 2020.

[6] X.-L. Bao, C. Su, Y. Xiong, W. Huang, and Y. Hu, “FLChain: a blockchain for auditable federated learning with trust and incentive,” in *Proceedings of the 5th International Conference on Big Data Computing and Communications (Bigcom)*, pp. 151–159, IEEE, Qing Dao, China, August 2019.

[7] U. Majeed and H. C. Seon, “FLchain: federated learning via MEC-enabled blockchain network,” in *Proceedings of the 20th Asia-Pacific Network Operations And Management Symposium (APNOMS)*, pp. 1–4, IEEE, Matsue, Japan, September 2019.

[8] W. Zhang, Q. Wang, and M. Li, “Medical image collaborative training based on multi-blockchain,” in *Proceedings 2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 590–597, IEEE, San Diego, CA, USA, November 2019.

[9] B. Yin, H. Yin, Y. Wu, and Z. Jiang, “FDC: a secure federated deep learning mechanism for data collaborations in the internet of things,” *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6348–6359, 2020.

[10] Y. J. Kim and C. S. Hong, “Blockchain-based node-aware dynamic weighting methods for improving federated learning performance,” in *Proceedings of the 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pp. 1–4, IEEE, Matsue, Japan, November 2019.

[11] I. Martinez, S. Francis, and A. S. Hafid, “Record and reward federated learning contributions with blockchain,” in *Proceedings of the 2019 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pp. 50–57, IEEE, Guilin, China, October 2019.

[12] W. Zhang, Q. Lu, Q. Yu et al., “Blockchain-based federated learning for device failure detection in industrial IoT,” *IEEE Internet of Things Journal*, vol. 8, 2020.

[13] Y.-L. Lu, X.-H. Huang, Y.-Y. Dai, S. Maharjan, and Y. Zhang, “Blockchain and federated learning for privacy-preserved data sharing in industrial IoT,” *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4177–4186, 2020.

[14] Y. Zhao, J. Zhao, L. Jiang et al., “Privacy-preserving blockchain-based federated learning for IoT devices,” *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1817–1829, 2021.

[15] C. Yu, L. Fang, L. Tong, X. Tao, L. Zheli, and L. Jin, “A training-integrity privacy-preserving federated learning scheme with trusted execution environment—science direct,” *Information Sciences*, vol. 522, pp. 69–79, 2020.

[16] H. Kim, J. Park, M. Bennis, and S.-L. Kim, “Blockchained on-device federated learning,” *IEEE Communications Letters*, vol. 24, no. 6, pp. 1279–1283, 2020.

[17] X. Wang, S. Garg, H. Lin, G. Kaddoum, J. Hu, and M. S. Hossain, “A secure data aggregation strategy in edge computing and blockchain empowered Internet of things,” *IEEE Internet of Things Journal*, vol. 99, p. 1, 2020.

[18] K. Toyoda and A. N. Zhang, “Mechanism design for an incentive-aware blockchain-enabled federated learning platform,” in *Proceedings of the 2019 IEEE International Conference on Big Data (Big Data)*, pp. 395–403, Los Angeles, CA, USA, December 2019.

[19] S. Wang, L. Ouyang, Y. Yuan, X. Ni, X. Han, and F.-Y. Wang, "Blockchain-Enabled smart contracts: architecture, applications, and future trends," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 11, pp. 2266–2277, 2019.

[20] Y. Huang, B. Wang, and Y. Wang, "MResearch on Ethereum private blockchain multi-nodes platform," in *Proceedings of the 2020 International Conference on Big Data, artificial intelligence and Internet of things engineering (ICBAIE)*, pp. 369–372, IEEE, Fuzhou, China, June 2020.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: surpassing human-level performance on imagenet classification," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, IEEE, Santiago, Chile, December 2015.

[22] Y. Lu, Q. Tang, and G. Wang, "On enabling machine learning tasks atop public blockchains: a crowdsourcing approach," in *Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 81–88, IEEE, Singapore, November 2018.

[23] D. Pietro, K. A. Ellersgaard, S. Čedomir, and P. Petar, "Analysis of the communication traffic for blockchain synchronization of IoT devices," in *Proceedings of the IEEE International Conference on Communications (ICC)*, pp. 1–7, IEEE, Kansas, MO, USA, May 2018.

[24] R. Kashyap, K. Aror, M. Sharma, and A. Aazam, "Security-Aware ga based practical byzantine fault tolerance for permissioned blockchain," in *Proceedings of the 4th International Conference on Control, Robotics and Cybernetics*, pp. 162–168, IEEE, Tokyo, Japan, September 2019.