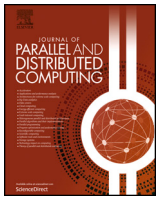
Journal of Parallel and Distributed Computing 196 (2025) 104995

Contents lists available at ScienceDirect

Journal of Parallel and Distributed Computing

journal homepage: www.elsevier.com/locate/jpdc



BRFL: A blockchain-based byzantine-robust federated learning model

Yang Li [a,b], Chunhe Xia [b,c], Chang Li [d], Tianbo Wang [e,*]

[a] School of Computer Science and Engineering, Beihang University, Beijing, China
[b] Beijing Key Lab. of Network Technology, Beihang University, Beijing, China
[c] Guangxi Collaborative Innovation Center of Multi-Source Information Integration and Intelligent Processing, Guangxi Normal University, Guilin, China
[d] School of Computer Science and Technology, Zhengzhou University of Light Industry, Zhengzhou, China
[e] School of Cyber Science and Technology, Beihang University, Beijing, China

ARTICLE INFO

Keywords:
Federated learning
Blockchain
Pearson correlation coefficient
Spectral clustering

ABSTRACT

With the increasing importance of machine learning, the privacy and security of training data have become a concern. Federated learning, which stores data in distributed nodes and shares only model parameters, has gained significant attention for addressing this concern. However, a challenge arises in federated learning due to the byzantine attack problem, where malicious local models can compromise the global model's performance during aggregation. This article proposes the Blockchain-based Byzantine-Robust Federated Learning (BRFL) model, which combines federated learning with blockchain technology. We improve the robustness of federated learning by proposing a new consensus algorithm and aggregation algorithm for blockchain-based federated learning. Meanwhile, we modify the block saving rules of the blockchain to reduce the storage pressure of the nodes. Experimental results on public datasets demonstrate the superior byzantine robustness of our secure aggregation algorithm compared to other baseline aggregation methods, and reduce the storage pressure of the blockchain nodes.

1. Introduction

Federated learning (FL) [1] has revolutionized the traditional approach to machine learning by storing data locally on multiple nodes rather than on centralized servers. These nodes train the same model and then upload it to the server, allowing devices like smartphones, laptops, and in-car computers to participate in model training.

However, federated learning has both positive and negative aspects [2]. On the positive side, it enhances the privacy and security of training data by keeping it locally stored and not sharing it with the server. It finds applications in various domains such as transportation, healthcare, and finance. On the negative side, since training data is not stored on the server, the server needs to aggregate parameters from multiple local models, which can be vulnerable to attacks from malicious nodes. Research on robust global model aggregation focuses on situations where some local models are under attack, while maintaining high performance for the global model [3]. The key aspect of this research is to identify the attacked models and exclude them during the global model aggregation process, while retaining benign models for aggregation. Therefore, secure global model aggregation algorithms play a vital role in federated learning. Specifically, our research aims to combat malicious local models and trace their activities. According to the survey [4], the detection of malicious models can be achieved through model aggregation algorithms, and when combined with blockchain technology, it enables the traceability of malicious parameters. Additionally, it provides rewards to nodes participating in federated learning, mitigating potential attacks such as middleman attacks and single point failures, and increasing the enthusiasm of nodes to participate in federated learning training [5].

In blockchain-based federated learning, two inevitable problems arise in model aggregation: *byzantine-attack* and *resource-cost*. These problems occur due to the lack of guarantee that every local model trains in the intended direction. Local models are sent to the server for aggregation, but the server does not participate in their training. Consequently, the server cannot is unable to verify the validity of the local models it receives, as it cannot access to the local training data. In this article, we focus on a specific type of *byzantine-attack*: one that involves modifying local model parameters. In this type of attack, the attacker does not possess knowledge of the model parameters of other nodes. The goal of such attacks is to reduce the accuracy of the global model or manipulate predictions for specific data. *Resource-cost* refers to the significant computational resources consumed by existing secure

* Corresponding author.
E-mail addresses: johnli@buaa.edu.cn (Y. Li), wangtb@buaa.edu.cn (T. Wang).

https://doi.org/10.1016/j.jpdc.2024.104995
Received 25 March 2024; Received in revised form 28 July 2024; Accepted 8 October 2024
Available online 10 October 2024

aggregation methods that rely on local model accuracy verification. Furthermore, integrating blockchain technology into federated learning offers the benefits of tamper-proofing and traceability of model parameters, thereby bolstering the security of federated learning. However, this approach also encounters challenges associated with block storage, additional computational overhead, and communication costs. In this paper, we leverage the advantages of blockchain to safeguard the security of federated learning while placing a particular focus on addressing the storage issue.

Researchers have developed aggregation algorithms mainly based on l_p distance [6,7], cosine angle [8,9], and accuracy validation [10–12] to detect malicious local models for safe model aggregation in federated learning. The l_p distance-based method calculates the distance between local model parameters and assumes that the distance between malicious model parameters and benign model parameters is greater than the distance between benign models, thereby detecting malicious models. The cosine angle-based method considers the angle between local models and previous global models and identifies them as malicious if the cosine angle exceeds a certain threshold. The accuracy validation-based method verifies the accuracy of local models using a prepared test dataset and identifies them as malicious if their accuracy falls below a certain threshold.

It is unfortunate that these three primary aggregation algorithms are not effective in detecting malicious models. The l_p distance-based method is susceptible to attackers who introduce minor biases to the parameters of a normal model, thereby compromising the global model. In the cosine angle-based method, attackers can manipulate the angles between malicious model updates and benign model updates to be smaller than the discrimination threshold of the defense method used. With regard to the accuracy validation-based method, it necessitates the utilization of an additional test dataset and significant computational resources for the purpose of validating the accuracy of local models. Furthermore, all three aforementioned methods assume that the proportion of malicious models is less than 50% [6,8,10].

To address these issues, we propose the Blockchain-based Federated Learning model (BRFL) to address possible Byzantine attacks during federated learning and improve the security. This model contains two main components: the Proof of Pearson Correlation Coefficient consensus algorithm (PPCC) and the Precision-based Spectral Aggregation (PSA) algorithm. PPCC selects the aggregation node for the next round based on the Pearson correlation coefficient between the local model and the global model from previous rounds. Furthermore, the model addresses the lack of test datasets in FL by utilizing the aggregation node's local dataset for local model accuracy validation. PSA clusters local models with high correlation and verifies their accuracy by calculating the average value, thereby detecting malicious models and addressing the resource cost problem. Experimental results demonstrate that our model exhibits high robustness and effectively reduces resource consumption.

Our contributions can be summarized as follows.

1. Unlike traditional federated learning, we construct a federated learning model on the blockchain network, use the consensus algorithm of the blockchain to select trustworthy aggregation nodes in each training round, based on the linear similarity between the local models and the global model of the previous round, and use the open and transparent feature of the blockchain to record the federated learning process, which enhances the robustness of federated learning.
2. Based on the linear similarity between local models, we verify the accuracy of the average local model in each cluster after clustering the local models, rather than verifying the accuracy of each local model. This allows us to select the average model with higher accuracy as the global model, which reduces the consumption of computational resources required for the verification of the local models.
3. The identification of the transaction type contained in the block is added to the block header. A new block saving rule is proposed, whereby different types of nodes can save different types of blocks as needed, thus reducing the storage pressure on nodes in the blockchain network.
4. Experiments on two real-world datasets demonstrate that our aggregation method outperforms baseline approaches in terms of accuracy. Moreover, our model reduces storage resource costs and provides effective incentives for local training nodes.

The rest of this article is organized as follows. We first give a comprehensive review of related works in Section 2. Next, we demonstrate the background of this article in Section 3, followed by the presentation of detailed design of BRFL in Section 4. After that, we conduct a series of experiments on two public datasets to evaluate BRFL in Section 5. Finally, Section 6 concludes this work and discusses future directions.

2. Related work

This section provides a comprehensive review of relevant literature. We discuss two approaches to enhancing the security of federated learning: combining blockchain and Byzantine robust aggregation methods.

Federated learning and blockchain both employ a distributed architecture. Currently, there are solutions that enhance the security of federated learning by employing improved aggregation algorithms. For instance, Yinbin Miao et al. use cosine similarity to detect malicious gradients uploaded by malicious clients, employ fully homomorphic encryption for secure aggregation, and utilize blockchain systems to facilitate transparent processes and regulatory compliance [13]. G. Bian et al. propose a secure aggregation scheme using differential privacy and fully homomorphic encryption, achieving a privacy-preserving Byzantine-robust federated learning solution. They use blockchain to record the training process and integrate Byzantine fault-tolerant consensus to further enhance robustness [14]. Zixuan Shu et al. introduce a novel committee-based Byzantine federated learning algorithm that ensures the elasticity, convergence, and correctness of training. The algorithm establishes a committee responsible for reviewing locally uploaded gradients. The committee uses a scoring system to evaluate the gradients uploaded by clients, and the server limits and summarizes the gradients based on the scoring results using a specified function. At the end of each training round, committee members switch based on a selection strategy [15]. Georges A. Kamhoua et al. propose an elastic verifiable federated learning algorithm based on a reputation scheme. They develop a task publisher selection algorithm and a blockchain-based multiparty learning architecture, allowing secure exchange and verification of local model updates without a central authority [16]. Xinru Yan et al. utilize blockchain to address single-point failure and untrustworthy aggregation results. They achieve reliable model aggregation in an asynchronous environment using practical Byzantine fault-tolerant protocols and enhance system robustness using differential privacy [17]. Rong Wang et al. propose a permissioned blockchain-based asynchronous federated learning system, where a permissioned blockchain serves as the federated learning server, consisting of a main blockchain and multiple sub-blockchains. Based on this architecture, they propose a permissioned blockchain-based federated learning asynchronous aggregation protocol, integrating learned models into the blockchain and performing second-order aggregation calculations [18]. Shili Hu et al. introduce a crowdsourced federated learning solution that trains neural networks using a hybrid blockchain architecture. Smart contracts are used for data validation on the main chain, while proxy re-encryption is used for privacy protection. They propose an asynchronous practical Byzantine federated learning (APBFL) algorithm based on consensus on a sidechain to enhance the reliability and security of the model [19].

Furthermore, several blockchain-based federated learning models also focus on addressing resource consumption, and various methods have been proposed for this purpose. Tasiu Muazu et al. introduce an improved linear regression model as the global learning model for federated learning systems. Gradient parameters are encrypted using Paillier encryption on the federated server side and then shared by federated

Table 1
An overview of existing research.

Study	Year	Technologies			Analysis		Properties			
		FL	Blockchain	Clustering	Security	Resource Saving	$\mathfrak{P}1$	$\mathfrak{P}2$	$\mathfrak{P}3$	$\mathfrak{P}4$
[13]	2022	✔	✔	✘	✔	✘	✘	✘	✘	✘
[14]	2023	✔	✔	✘	✔	✘	✘	✘	✘	✘
[15]	2023	✔	✔	✘	✔	✘	✘	✔	✘	✘
[16]	2021	✔	✔	✘	✔	✘	✘	✘	✘	✘
[17]	2023	✔	✔	✘	✔	✘	✘	✘	✘	✘
[18]	2022	✔	✔	✘	✔	✘	✘	✘	✘	✘
[19]	2021	✔	✔	✘	✔	✘	✘	✘	✘	✘
[20]	2024	✔	✔	✘	✔	✔	✘	✘	✘	✘
[22]	2023	✔	✔	✘	✔	✔	✘	✘	✘	✘
[23]	2023	✔	✔	✘	✔	✔	✔	✔	✘	✘
[24]	2023	✔	✔	✘	✔	✔	✘	✘	✘	✔
[25]	2023	✔	✔	✘	✔	✔	✘	✘	✘	✘
[26]	2022	✔	✔	✘	✔	✔	✘	✘	✘	✘
BRFL	2024	✔	✔	✔	✔	✔	✔	✔	✔	✔

FL: Federated Learning; $\mathfrak{P}1$: Dynamic Aggregation Nodes; $\mathfrak{P}2$: High Efficiency Consensus Algorithm; $\mathfrak{P}3$: Block Categorize; $\mathfrak{P}4$: Local Model Accuracy Verification; ✔: true or supports; ✘: false or not supports.

clients. The deployment of blockchain provides new security features for edge computing. The proposed system reduces computational costs while achieving security and privacy advantages [20]. Ervin Moore et al. discuss the challenges, solutions, and future directions of deploying blockchain successfully in resource-constrained federated learning environments. They emphasize that if blockchain mechanisms require expensive computational resources, federated learning clients with limited resources cannot participate in training. They extensively analyze potential network threats observed in resource-constrained federated learning environments and discuss the crucial role of blockchain in preventing these network attacks [21]. Xiuhua Fu et al. highlight resource management as a key issue that future Internet of Things (IoT) needs to address and focus on federated learning-based resource management mechanisms in IoT. They propose a framework for IoT resource management integrating blockchain technology to ensure the security of FL model parameter exchange. They present a specific federated learning-based resource management and blockchain trust assurance algorithm and utilize support vector machine (SVM) classifiers to detect malicious nodes, thereby avoiding performance impact on FL-based algorithms [22]. Jiaxiang Zhang et al. propose a blockchain-based federated learning framework to support trusted and reliable federated learning patterns in IoT. They design a committee-based participant selection mechanism in the proposed framework, dynamically selecting aggregation nodes and local model updates to build the global model. Additionally, they jointly execute channel allocation, block size adjustment, and block producer selection to balance between the energy consumption and convergence speed of federated learning models [23]. Yueyue Dai et al. design a fine-grained resource allocation scheme for blockchain-based federated learning considering device credibility, data quality, and energy resources. They introduce credit-based blockchain-supported federated learning and address a resource allocation problem considering credit, data quality, accuracy, latency, and energy resources. They propose a deep reinforcement learning-based algorithm to solve this problem and construct a blockchain-supported federated learning platform [24]. Zhilin Wang et al. propose an edge server resource allocation scheme to provide optimal services at minimal cost. They model the resource allocation challenge as a multivariate, multi-constraint, and convex optimization problem, considering task energy consumption and treating the completion time of each task as a quality of service constraint. They design two alternating direction of multiplier-based algorithms for homogeneous and heterogeneous scenarios, employing equal and on-demand resource allocation strategies, respectively [25]. Additionally, Zhilin Wang et al. design an incentive mechanism to help task publishers allocate appropriate rewards for training and mining for each client, and then clients use a two-stage Stackelberg game to determine the computational resource allocation for each subtask. By analyzing the utilities of task publishers and clients, they transform the game model into two optimization problems, which are sequentially solved to derive the optimal strategies for task publishers and clients [26]. An overview of existing research shows in Table 1.

However, current research still has room for improvement in terms of accuracy in countering Byzantine attacks, as well as optimizing the utilization of computational and storage resources. To the best of our knowledge, this is the first work to address both *byzantine-attack* and *resource-cost* issues simultaneously. Compared to existing research, we reduce both computational and storage resource consumption while ensuring the privacy and security of federated learning. We aim to meet the privacy and security requirements of federated learning with minimal cost.

3. Background

3.1. Blockchain

Blockchain [27], proposed by Nakamoto in 2008, aims to establish a decentralized and untrusted digital cash system. It is based on a peer-to-peer network and incorporates encryption algorithms, consensus algorithms, and other technologies.

In traditional cash systems, an institution endorses the currency, thereby assigning value to the coin. People recognize the value of the currency by acknowledging the credibility of the endorsing organization. However, in the first blockchain application, Bitcoin, Nakamoto introduces a method for everyone to maintain records of digital cash transactions between any individuals, eliminating the need for a single organization to endorse the trust of the transaction. The more people involved in ledger-keeping, the more secure and reliable the transaction history becomes.

A blockchain consists of blocks that are connected through a bidirectional linked list. Each block is divided into a block head and a block body. The block head contains the block generation timestamp, the hash value of the previous block, the hash value of the current block, the Merkle root (which ensures transaction consistency within the block), the block height, and other information. The block body contains transactions that occurred during a specific time period. Once recorded in a block, the transactions cannot be modified.

Consensus algorithms [28] play a crucial role in blockchain by organizing network nodes to reach a consensus on transactions and ensuring the correctness and continuity of the blockchain. In Bitcoin, the Proof of Work (PoW) consensus algorithm [27] was introduced, where miners compete to solve a complex mathematical puzzle to generate the next block. The first miner to solve the puzzle successfully adds the block to the chain and receives rewards for their efforts.

Blockchain can be utilized to address privacy protection issues in federated learning [29–31]. By leveraging the tamper-proof, transparent, and decentralized nature of blockchain, the security of the federated learning system can be enhanced by storing important data on the blockchain.

3.2. Pearson correlation coefficient

Pearson correlation coefficient (PCC) [32] aims to describe the linear correlation between two variables, assume there has a variable $A = \{a_1, a_2, \ldots, a_n\}$, $B = \{b_1, b_2, \ldots, b_n\}$, the PCC score ρ can be calculated as:

$$\begin{aligned}
&\rho(A,B) \\
&= \frac{E[(A-\mu_A)(B-\mu_B)]}{\sigma_A \sigma_B} \\
&= \frac{E(A,B) - E(A)E(B)}{\sqrt{(E(A^2) - E^2(A))}\sqrt{(E(B^2) - E^2(B))}} \\
&= \frac{n\sum_{i=1}^{n} a_i b_i - (\sum_{i=1}^{n} a_i)(\sum_{i=1}^{n} b_i)}{\sqrt{n\sum_{i=1}^{n} a_i^2 - (\sum_{i=1}^{n} a_i)^2}\sqrt{n\sum_{i=1}^{n} b_i^2 - (\sum_{i=1}^{n} b_i)^2}}
\end{aligned} \tag{1}$$

Hence, $\rho(A,B) \in [-1,1]$, $\rho(A,B) \leftarrow 0$ means variable A and B has no linear correlations, $\rho(A,B) \leftarrow 1$ means A and B has positive linear correlations, when one of them increase, the other will also increase; $\rho(A,B) \leftarrow -1$ means A and B has negative linear correlations, when one of them increase, the other will decrease. PCC has features, such as:

1) Symmetry. ρ(A,B) = ρ(B,A), the PCC between the two specific variables is the same.
2) Displacement invariant. Changes in the expected values of variable A and variable B do not affect the PCC.
3) Scale invariance. Exponential change in variable A and variable B does not affect the PCC.

PCC can be used to describe two vectors relation. For the models in federated learning, the PCC can be used to describe the strength and direction of the linear relationship between every two models.

3.3. Spectral clustering

The fundamental concept of spectral clustering [33] is to convert samples and their similarity matrix into a graph model. By cutting the edges of the graph model, it aims to minimize the edge weights between different subgraphs and maximize the edge weights within each subgraph. Spectral clustering is employed to classify samples and tackle the clustering problem of sparse data, which is a limitation of the k-means clustering algorithm.

Assuming we have a dataset $X = \{x_1, x_2, ..., x_n\}$, that can be classified into k classes, the algorithm proceeds as follows:

1. Construct the similarity matrix $S \in \mathbb{R}^{n\times n}$ and obtain the corresponding degree matrix $D \in \mathbb{R}^{n\times n}$, adjacency matrix $W \in \mathbb{R}^{n\times n}$, and Laplacian matrix $L = D - W$.
2. Normalize L by $L^* = D^{-1/2} L D^{-1/2}$ and compute the m smallest eigenvalues of L^* and their corresponding eigenvectors f.
3. The eigenvector matrix F, composed of the corresponding eigenvectors f. Normalize F row-wise to obtain the final $n \times m$-dimensional feature matrix $F \in \mathbb{R}^{n\times m}$.
4. Cluster the row vectors of F, where each row vector represents an m-dimensional sample. With a total of n samples, set the number of clusters as k, and obtain the partitioning of clusters $C = \{c_1, c_2, ..., c_k\}$.

Characteristics of spectral clustering:

1. Spectral clustering only requires the similarity matrix of the data, making it effective for clustering sparse data.
2. Due to its utilization of dimensionality reduction techniques, spectral clustering exhibits better performance than traditional clustering algorithms when dealing with high-dimensional data.

4. Blockchain-based federated learning model

In this section, we present a novel approach that combines blockchain technology with federated learning to enhance security and robustness. We propose BRFL, which incorporates a customized consensus algorithm and a robust aggregation mechanism. Our approach leverages blockchain technology, Pearson correlation coefficient (PCC), and spectral clustering techniques to effectively filter out malicious model during the global model aggregation process. With blockchain, we utilize the consensus algorithm to select trustworthy nodes for model validation and block packaging. Blockchain allows us to track the activities of nodes during the federated learning training process. In case of any malicious behavior, we can trace it back for investigation. This approach offers enhanced security and robustness for federated learning while reducing computation power costs. Furthermore, we introduce the utilization of IPFS and a new block structure to minimize storage resource costs.

4.1. Model overview

In Fig. 1 shows the architecture of BRFL. In order to improve the security of global model aggregation, BRFL design goal is as follows:

- Aggregating local update model parameters makes the global model resilient against *byzantine-attack*, ensuring that the aggregated global model performs well even in the presence of such attacks.
- By incorporating consensus algorithms in blockchain, the selection of aggregation nodes is guaranteed to maintain the quality and efficiency of global model aggregation.
- Blockchain technology is utilized to achieve secure transmission and traceability of model updates, ensuring that the transmission process remains untampered and providing the ability to trace malicious model.

As depicted in Fig. 1, the local model and global model are stored in the blockchain. Local training nodes retrieve the global model from the blockchain and utilize their own datasets to train the local model. After local training, the updated local model parameters are uploaded to IPFS, and the Content Identifier (CID) returned by IPFS is then uploaded to the blockchain. We ensure the security of local models by storing the CID on the blockchain, leveraging the blockchain's immutable properties. The aggregation nodes obtain the local model parameters from IPFS through CID, perform aggregation, and store the global model into the blockchain for the next round of model training after the aggregation is completed. Unlike the existing centralized [1] and decentralized [34] aggregation methods in federated learning, we propose a new aggregation approach based on a consensus algorithm. This method leverages the correlation between local models and the global model, selecting nodes with higher correlation to the global model as aggregation nodes. By doing so, we avoid the privacy risks and communication costs associated with centralized aggregation methods, while also addressing the model consistency and communication complexity [35]issues of decentralized aggregation methods.

4.2. Initialization

In BRFL we denote all nodes' set is $N = \{n_1, n_2, \ldots, n_\alpha\}, \alpha \in \mathbb{N}^+$ is the number of all nodes. N contains aggregation nodes set, local training

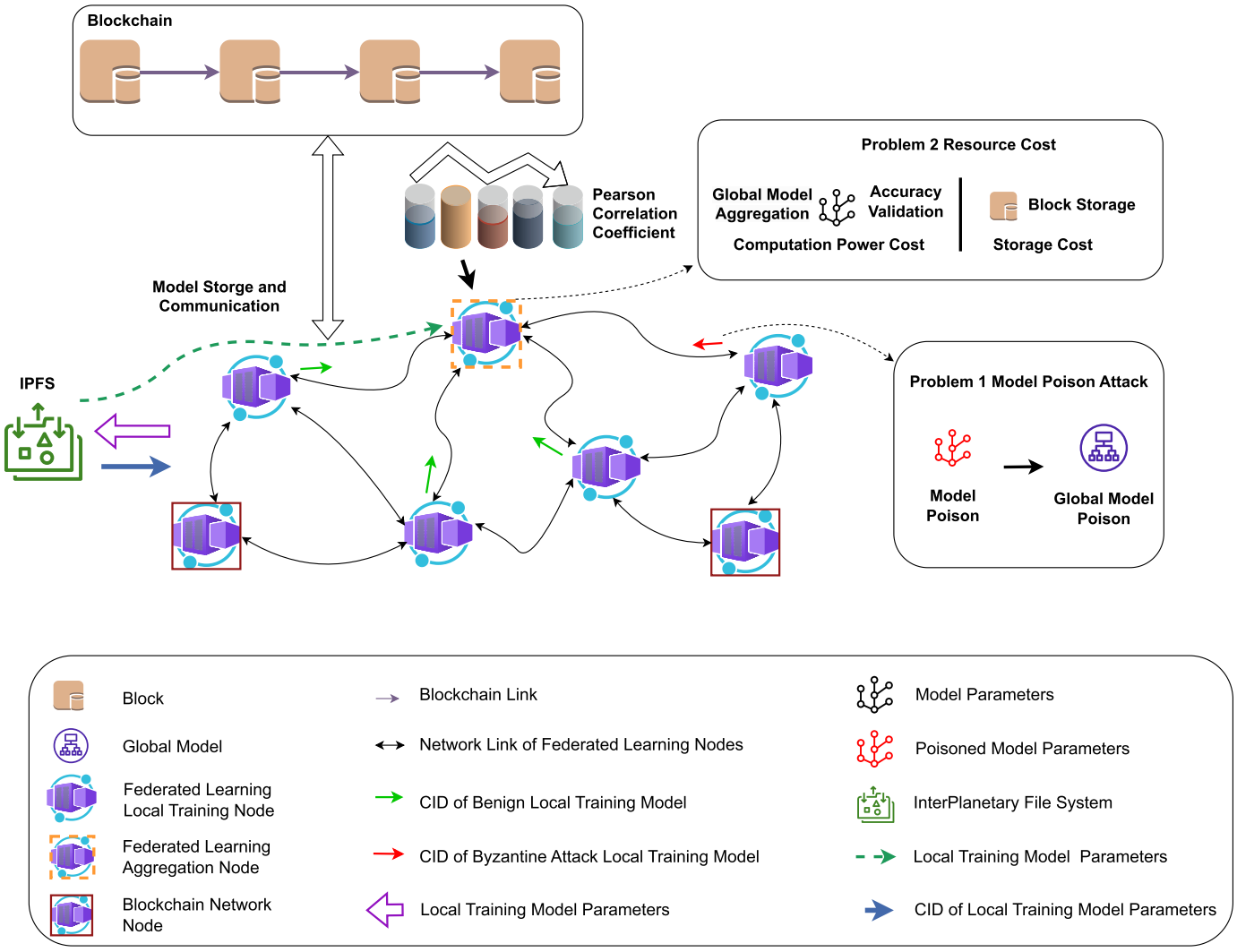


Fig. 1. BRFL model structure.

nodes set and blockchain nodes set. There are three types of nodes in BRFL:

- Aggregation node (An): The set of aggregation nodes is denoted as $AN = \{An_1, An_2, \ldots, An_\beta\}$, where $\beta \in \mathbb{N}^+$. These nodes aggregate local models to form the global model and package blocks.
- Local training node (Lt): The set of local training nodes is denoted as $LT = \{Lt_1, Lt_2, \ldots, Lt_\gamma\}$, where $\gamma \in \mathbb{N}^+$. The local training nodes are responsible for training the local models and sending them to the aggregation nodes (An) for the aggregation of the global model. Both the local training nodes (Lt) and the aggregation nodes (An) have their own datasets.
- Blockchain network node (Bn): The blockchain network nodes (Bn) do not participate in the federated learning process. They are denoted as $BN = Bn_1, Bn_2, \ldots, Bn_\varkappa$, where $\varkappa \in \mathbb{N}^+$. The presence of more nodes in the blockchain network ensures that the ledger is consensus by a larger number of nodes, thereby enhancing the security of the blocks within the blockchain.

There are two types of models in BRFL:

- Local model (lm): The local model set is denoted as $LM = \{lm_1^r, lm_2^r, \ldots, lm_\gamma^r\}$, where $r \in \mathbb{N}^+$ represents the global training round, and γ represents the node id. lm refers to the local training model (Lt) and in each communication round. After local training, the updated lm is published on IPFS.
- Global model (gm): The global model set is denoted as $GM = \{gm_1^\theta, gm_2^\theta, \ldots, gm_\chi^\theta\}$, where $\chi \in \mathbb{N}^+$ represents the global round number and θ is the aggregation node ID. The global model is aggregated by the aggregation nodes (An). The global model represents a comprehensive representation of the collective information within the federated learning network, combining the data characteristics and insights contributed by each participant.

In the beginning of federated learning in BRFL, several steps are followed. First, the maximum training round (MTR) is set, along with the model structure. The global model parameters are randomly initialized. Next, a certain number (β) of LT are randomly selected to become AN. The global model is then packaged and stored in the blockchain.

4.3. Local model training and communication

Blockchain possesses characteristics such as transparency, immutability, and decentralized maintenance. These features ensure the security and transparency of data on the blockchain, making it accessible to all nodes. However, these advantages also introduce certain challenges, including storage space and communication efficiency issues.

In a blockchain, blocks need to be stored on nodes. The more nodes that store blocks in the blockchain network, the higher the security level

and the lower the likelihood of tampering. However, storing blocks requires local storage space from the nodes. In the case of federated learning, where there are numerous training rounds, multiple participating nodes, and long training cycles, the types of participating nodes vary greatly. These nodes can include mobile phones, cars, computers, and more, each with different storage capacities.

To address these challenges, we storing the local models in the InterPlanetary File System (IPFS), while storing the pointers returned by IPFS within the blockchain. In federated learning, with multiple participating local training nodes, storing all the local models on the blockchain would result in numerous blocks generated in each training round, which hampers communication efficiency. IPFS will return a hash sequence, Content Identifier (CID), for each certain file upload, aggregation nodes can retrieve the local models from IPFS based on the CID, perform model aggregation, and store the aggregated global model within the blockchain. By storing the global model only once per round within the blockchain, local training nodes can directly access it from the blockchain instead of IPFS, thereby improving their work efficiency. Assume current round is φ th round, $Lt_{\varpi}, \varpi \in [1,\gamma]$ get $gm^{\theta}_{\varphi-1}$ from blockchain, set $gm^{\theta}_{\varphi-1}$ as its local model $lm^{r}_{\varpi} \leftarrow gm^{\theta}_{\varphi-1}$, doing local training using its own dataset and computation power, get updated local model $lm^{r}_{u;\varpi}$. Lt_{ϖ} upload their LM to IPFS, get CID returns from IPFS for the uploaded $lm^{r}_{u;\varpi}$, denoted as $CID_{lm^{r}_{u;\varpi}}$, which can be used to retrieve $lm^{r}_{u;\varpi}$. LT publish $CID_{lm^{r}_{u;\varpi}}$ as a transaction to blockchain network.

4.4. Proof of Pearson correlation coefficient consensus algorithm

In BRFL, we propose a consensus algorithm called Proof of Pearson Correlation Coefficient (PPCC) to select the LT that will become AN at the beginning of each communication round. In federated learning, one challenge is that training data is stored in training nodes and not shared with others to protect privacy. While this enhances data security, it also poses a problem for the server to verify the validity of local models, leading to potential issues such as *byzantine-attack* including data poisoning and model poisoning attacks.

To address this challenge, a committee of AN from the LT is selected in each training round. The Pearson correlation coefficient is then calculated between the local model of the current round and the global model aggregated from the previous round. This is one of the most commonly used similarity measures for high-dimensional variables, as cited in reference [36]. The global model from the previous round is employed as the evaluation criterion for the current round's local model. This approach is based on the key insight that the local model in the current round with the highest linear correlation to the global model of the previous round is considered trustworthy. It is hypothesized that the lower the linear correlation between the local model and the global model from the previous round, the higher the likelihood that this model is malicious, due to the lower similarity between the local and global models. The selected aggregation nodes utilize their local datasets to validate the veracity of the local models generated by the participating training nodes. This methodology addresses the absence of a test dataset, thereby enhancing the overall security and integrity of the federated learning process.

Assume current round is $\epsilon(\epsilon > 1)$ th round, the algorithm details shows in Algorithm 1, the steps of PPCC can be summarized as follows:

1. Calculate last round's LM and gm different layer's PCC between LM and gm;

$$\sigma(\Theta(lm^{\epsilon-1}_{\eta})) = \sqrt{E((\Theta(lm^{\epsilon-1}_{\eta}))^2) - E^2(\Theta(lm^{\epsilon-1}_{\eta}))} \tag{2}$$

$$\sigma(\Theta(gm^{\theta}_{\epsilon-1})) = \sqrt{E((\Theta(gm^{\theta}_{\epsilon-1}))^2) - E^2(\Theta(gm^{\theta}_{\epsilon-1}))} \tag{3}$$

$$\rho(lm^{\epsilon-1}_{\eta}, gm^{\theta}_{\epsilon-1}) = \frac{cov(lm^{\epsilon-1}_{\eta}, gm^{\theta}_{\epsilon-1})}{\sigma(\Theta(lm^{\epsilon-1}_{\eta}))\sigma(\Theta(gm^{\theta}_{\epsilon-1}))} \tag{4}$$

Hence, $lm^{\epsilon-1}_{\eta}, \eta \in [1,\gamma]$ is the η th local model in $\epsilon-1$ round, $gm^{\theta}_{\epsilon-1}$ is the global model aggregate by $Lt_{\theta}, \theta \in [1,\gamma]$ in $\epsilon-1$ round, $layer_{\rho}$ is the ρ th layer in the local model parameters, $layer_{\tau}$ is the τ th layer in the global model parameters. $\Theta(lm^{\epsilon-1}_{\eta})$ is concatenate different layers in $lm^{\epsilon-1}_{\eta}$ to one vector, so as $\Theta(gm^{\theta}_{\epsilon-1})$.

2. Sort the values of ρ in descending order. Select the top β local models based on this sorting. The nodes Lt that uploaded these selected local models are designated as the AN for this round.

$$AN \leftarrow Sort(\rho(lm^{\epsilon-1}_{\eta}, gm^{\theta}_{\epsilon-1}) \mid \eta \in [1,\gamma])[:\beta] \tag{5}$$

The selected AN verify the accuracy of the LM and package the transactions to the blockchain.

The block will be packaged by one An, but all the AN in this round will be considered the block miner. Therefore, a block may have multiple miners, and all the miners will receive tokens as rewards. The algorithm detailed steps as shown in Algorithm 1.

By demonstrating the linear correlation between LM and the GM, we select AN to package blocks. We measure the credibility of nodes in this way. The GM is aggregated from the local models of the previous round. We believe that if the PCC between the local model and the global model is larger, then its trend is closer, and the model better aligns with the advancement direction of the global model.

In comparison to the prevalent PoW consensus algorithm [27], our proposed PPCC consensus algorithm draws inspiration from the DPoS consensus algorithm [37]. In each round, it selects packaging nodes based on the similarity between models, thereby improving packaging speed without the need for extensive computational resources. Unlike the DPoS consensus algorithm, PPCC no longer relies on node voting to select delegate nodes. In contrast, it addresses the issue of collusion and coordinated attacks among nodes through the use of model similarity and accuracy validation, thereby providing a reliable basis for the selection of packaging nodes.

Algorithm 1 PPCC consensus algorithm.

Input: L_lm: last round local models list; gm: last round global model; num_AN: number of AN;

Initialization: PL: Pearson Correlation Coefficient list; AN: Aggregation nodes list;
if gm == None **then**
  return random select AN from LT;
end if
for i = 0; i < L_lm.length; i++ **do**
  for each layer in $L\_lm[i]$ **do**
    $\rho = \frac{E[(Vector_{gm}-\mu(Vector_{gm}))(Vector_{L\_lm[i]}-\mu(Vector_{L\_lm[i]}))]}{\sigma(Vector_{gm})\sigma(Vector_{L\_lm[i]})}$;
    PL[i] = ρ;
  end for
end for
sorted_dict ← sort PL in descend order;
AN ← sorted_dict[:num_AN]

Output: AN; PL

4.5. Precision-based spectral aggregation algorithm

BRFL utilizes IPFS to store the LM and stores the CID returned by IPFS in the blockchain. This approach addresses the challenge of storing a large number of local models when there are numerous local training nodes (LT) in the network. The aggregation nodes (AN) can retrieve the corresponding LM using the CID stored in the blockchain.

Existing security aggregation algorithms often identify *byzantine-attack* based on factors such as the distance between local models, the accuracy of local models, and the cosine angle between local and global models. However, these methods have limitations including low fault tolerance, high resource consumption, and insensitivity to a small number of features. In this work, we propose a secure aggregation algorithm

called the Precision-based Spectral Aggregation (PSA) algorithm. It combines the Pearson Correlation Coefficient (PCC), spectral clustering, and accuracy verification to detect attacked lm.

Firstly, the local and global models' different layers are concatenated to form a single vector, after which the PCC between the local models is calculated, as shown in Equations (6), (7), (8). Secondly, the local models are clustered using spectral clustering, based on the PCC. This allows the average parameters for each cluster to be calculated after clustering. The aggregation nodes selected by the PPCC consensus algorithm then verify the accuracy rate of the average parameters using its local database. The global model is derived from the average parameters with the highest average accuracy, validated by aggregation nodes. This approach enhances fault tolerance, reduces resource consumption, and improves sensitivity to small samples compared to existing methods.

The algorithm details show in Algorithm 2.

$$\sigma(\Theta(lm_{\eta}^{\epsilon})) = \sqrt{E((\Theta(lm_{\eta}^{\epsilon}))^2) - E^2(\Theta(lm_{\eta}^{\epsilon}))} \tag{6}$$

$$\sigma(\Theta(lm_{\emptyset}^{\epsilon})) = \sqrt{E((\Theta(lm_{\emptyset}^{\epsilon}))^2) - E^2(\Theta(lm_{\emptyset}^{\epsilon}))} \tag{7}$$

$$\rho(lm_{\eta}^{\epsilon}, lm_{\emptyset}^{\epsilon}) = \frac{cov(lm_{\eta}^{\epsilon}, lm_{\emptyset}^{\epsilon})}{\sigma(\Theta(lm_{\eta}^{\epsilon}))\sigma(\Theta(lm_{\emptyset}^{\epsilon}))} \tag{8}$$

$$\rho_{\mathbf{L},\mathbf{L}}^{\epsilon} = \begin{bmatrix} 1 & pcc_{lm_1,lm_2} & \cdots & pcc_{lm_1,lm_\gamma} \\ pcc_{lm_2,lm_1} & 1 & \cdots & pcc_{lm_2,lm_\gamma} \\ \vdots & \vdots & \ddots & \vdots \\ pcc_{lm_\gamma,lm_1} & pcc_{lm_\gamma,lm_2} & \cdots & 1 \end{bmatrix} \tag{9}$$

Hence, $\rho_{\mathbf{L},\mathbf{L}}^{\epsilon} \in \mathbb{R}^{(\gamma,\gamma)}$ in Equation (9) represents the matrix of Pearson Correlation Coefficient (PCC) relationships between local models in round ϵ.

The strength of the linear correlation between two local models (lm) can be determined by their PCC value. By clustering the local models based on PCC, we can group together models that exhibit strong linear correlation. We adopted PCC as the evaluation criterion for the relationship between models. We believe that the closer the PCC is to 1 between two nodes, the stronger the linear correlation, and the closer the performance of these two nodes. In comparison to the approach proposed by [38], we utilize the Pearson correlation coefficient (PCC) as the evaluation criterion for measuring the linear correlation between models. Furthermore, we employ spectral clustering to cluster the local models, thereby aggregating more relevant models together and achieving a more accurate classification of the models.

Existing security aggregation algorithms have proposed schemes to verify the accuracy of local models to detect malicious attacks. However, individually verifying the accuracy of each local model can consume a significant amount of computational resources of the aggregation nodes (AN). Moreover, the computational capacity of the AN limits the application of such schemes, and if the local training process generates a large number of models, the validation time will also increase. The algorithm outlines the steps involved in the process, as shown in Algorithm 2.

Let's assume there are ν clusters ($\nu \in \mathbb{N}^+$), and the average of the local model in the λ-th cluster, denoted as $CL_{a;\lambda}^{lm}$, as shown in Equation (10). Since local models within the same cluster exhibit similar linear correlations, $CL_{a;\lambda}^{lm}$ can effectively represent the model distribution in that cluster. After clustering the local models, we only verify the accuracy of the cluster average model parameters using the AN. Here, $CL_{a;\lambda}^{lm}$ represents the linear trend of the models in cluster λ. By evaluating $CL_{a;\lambda}^{lm}$, we can verify the nodes in that cluster without consuming excessive computational resources. The global model for the current round will be chosen based on the $CL_{a;\lambda}^{lm}$ with higher accuracy. Following one round of federated learning, each node Lt will collectively share their local training rewards (lsr) based on the Pearson correlation coefficient (PPCC) if their local models are in the higher accuracy cluster. Additionally, each active node An will receive an aggregation stake reward (asr) for model validation and block packaging purposes.

$$CL_{a;\lambda}^{lm} = \frac{\sum_{\mu=1}^{\omega} lm_{\mu}}{\omega} \tag{10}$$

Hence, let $\omega \in \mathbb{N}^+$ and $\omega < \beta$, where β is a dynamic number that indicates the number of nodes in each cluster.

Global model aggregation mechanism based on conventional federated learning methods Fedavg, as shown in Equation (11):

$$gm_{\epsilon}^{\theta} = \frac{\sum_{\lambda=1}^{\nu} CL_{a;\lambda}^{lm}}{\nu} \tag{11}$$

Algorithm 2 PSA algorithm.

Input: LMS: local model set; $Num_{cluster}$: cluster number; LT: local training nodes set; AN: aggregation nodes set;

Initialization: PM: Pearson Correlation Coefficient matrix; AL: accuracy validation list; C: vector;
for j = 0; j <LMS.length(); j++ **do**
  for k = j+1; k <LMS.length(); k++ **do**
    $\rho \leftarrow \frac{E[(C_{LMS[j]} - \mu(C_{LMS[j]}))(C_{LMS[k]} - \mu(C_{LMS[k]}))]}{\sigma(C_{LMS[j]})\sigma(C_{LMS[k]})}$;
  end for
  $PM[j][k] \leftarrow \rho$; $PM[k][j] \leftarrow \rho$;
end for
cluster_result ← spectral_clustering(PM, $Num_{cluster}$)
for $\lambda = 0$; $\lambda < Num_{cluster}$; λ ++ **do**
  for lm in *cluster_result*[λ] **do**
    $Sum_{lm} \leftarrow Sum_{lm} + lm$
  end for
  $CL_{a;\lambda}^{lm} \leftarrow \frac{Sum_{lm}}{Count(cluster\_result[\lambda])}$; $An_{num} \leftarrow 0$;
  for An in AN **do**
    $SUM_{acc} \leftarrow SUM_{acc} + An.validate(CL_{a;\lambda}^{lm})$
    $An_{num} \leftarrow An_{num} + 1$
  end for
  $AL[\lambda] \leftarrow SUM_{acc}/An_{num}$
end for
$Max\_index \leftarrow \max(AL)$;
$gm_{\epsilon}^{\theta} \leftarrow CL_{a;Max\_index}^{lm}$

Output: gm_{ϵ}^{θ} : Global Model in ϵ round

4.6. Block package

Storing models on the blockchain ensures data security. However, if all the local models generated by participating nodes in federated learning training are stored on the blockchain, it would impose significant storage pressure on the nodes.

To address this problem, we propose a new transaction type (denoted as $\mathbb{T}$) called identity in the block header to distinguish different types of transactions in the block body, as shown in Fig. 2. This approach allows nodes in the blockchain to save specific blocks associated with a particular transaction type ($\mathbb{T}$), thereby reducing the storage resources required by the nodes.

Once the global model aggregation is completed, the BRFL packages the following components into distinct blocks: the Content Identifier (CID), global model, accuracy records, and PCC transactions. The $\mathbb{T}$ can be categorized into four types:

1. Local Model Link CID ($\mathbb{T}_{CID}$): The transactions in the block body contain the Content Identifier (CID) linked to the local model stored in IPFS. Transactions with CID in a round will be packed into the same block.
2. Global Model ($\mathbb{T}_{GM}$): The transactions in the block contain the global model.

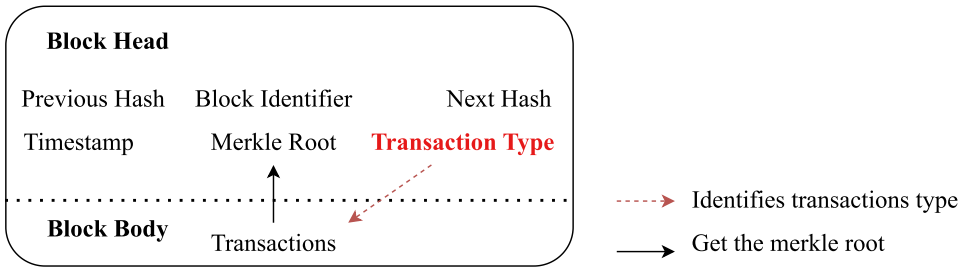


Fig. 2. Block structure in BRFL.

3. Accuracy Verification Record ($\mathbb{T}_{AVR}$): The transactions contain the accuracy record of $CL^{lm}_{a;\lambda}$ verified by AN.
4. Pearson Correlation Coefficient Records ($\mathbb{T}_{PCC}$): The transactions contain the PCC records between the local models and the global model in the previous round, as well as the PCC records between the local models in the current round.

The nodes in the blockchain have the flexibility to choose which type of transaction type ($\mathbb{T}$) blocks they want to save. In the case of LT, the $\mathbb{T}_{GM}$ block is necessary, while the $\mathbb{T}_{CID}$ block, $\mathbb{T}_{AVR}$ block, and $\mathbb{T}_{PCC}$ block are optional. On the other hand, for each round of AN, all four blocks ($\mathbb{T}_{GM}$, $\mathbb{T}_{CID}$, $\mathbb{T}_{AVR}$, and $\mathbb{T}_{PCC}$) must be downloaded to avoid any security issues arising from unsaved blocks. The BN node can save all the blocks to maintain ledger security through consensus.

5. Performance evaluation

5.1. Performance analysis

5.1.1. Time cost analysis

In BRFL, we propose the PPCC consensus algorithm inspired by the DPoS consensus algorithm, which also adopts the selection of a group of nodes as packing nodes, and thus has a similar packing efficiency as the DPoS consensus algorithm. At the same time, our proposed block categorize strategy increases the number of blocks, but the amount of transmitted data in the blockchain network does not change significantly, so it will not cost more time on the communication and transmission of the blockchain.

5.1.2. Computation cost analysis

The places where computational overheads in BRFL may occur include the calculation of Pearson correlation coefficients between local and global models and among local models, spectral clustering of local models, and calculating the accuracy of the average of local models for each cluster. In these processes, compared to the previously mentioned federated aggregation methods based on l_p distance, cosine similarity, accuracy, etc., we combine multiple algorithms simultaneously in our model, which increases the computational resource consumption to a certain extent. Nevertheless, we have enhanced the consensus algorithm in the blockchain to accommodate the specific characteristics of federated learning. This has resulted in a reduction in the consumption of computational resources in comparison to the widely used PoW consensus algorithm, while simultaneously enhancing the robustness of federated learning.

5.1.3. Security analysis of PPCC

Sybil Attack. A Sybil Attack involves forging identities to control a majority of nodes in a blockchain, enabling the attacker to launch attacks against other nodes. In PPCC, nodes are selected to enter the packaging queue by comparing the Pearson similarity coefficient between local model parameters and the previous round's global model parameters. For an attacker to execute a Sybil Attack, they would need to control a significant number of nodes. Notably, even if an attacker controls a substantial number of nodes, the malicious local model parameters must closely resemble the previous round's global model parameters to be included in the packaging queue and launch an attack. However, if the malicious local model parameters are similar to the previous round's global model parameters, the Sybil Attack will have minimal impact on the global model. Therefore, our method enhances the defense against Sybil Attacks.

Eclipse Attack. Eclipse Attack forging the identities of legitimate nodes to isolate the targeted node from other nodes in the network. As analyzed above, our proposed PPCC algorithm can effectively resist Eclipse Attacks by calculating the linear similarity between local models and the global model.

DDoS Attack. The proposed PPCC is analogous to the DPoS consensus algorithm, whereby a group of representative nodes is selected to pack the blocks in each communication round. This prevents other nodes from participating in the packing process, thereby reducing the likelihood of an attacker controlling the process through a DDoS attack. Furthermore, the representative nodes of PPCC in each round are randomly selected, enhancing the resilience of the PPCC consensus algorithm to DDoS attacks.

51% Attack. An attacker can manipulate the block packaging process by controlling more than 51% of the nodes in a blockchain network, known as a 51% Attack. Double-spend attacks, selfish mining attacks, and long-range attacks can all be considered variations of 51% attacks. In our proposed PPCC algorithm, an attacker would need to control more than 51% of the blockchain nodes to initiate a double-spend attack. However, since the selection of packaging nodes requires comparing the similarity between local models and the global model, the packaging process remains under the control of benign nodes even if more than 51% of the nodes are compromised. During a double-spend attack, the attacker, with control over 51% of the nodes, will generate a new blockchain at a faster rate than the honest chain. According to [27], the probability of the attacker's chain catching up with the honest chain follows a Poisson distribution:

$$\mathfrak{T} = \mathfrak{z}\mathfrak{q}/\mathfrak{p} \tag{12}$$

where $\mathfrak{z}$ represents the number of blocks required to confirm a transaction after the attacker initiates it, $\mathfrak{q}$ is the probability that the attacker generates the next block, and $\mathfrak{p}$ is the probability that the honest chain generates the next block. Given the assumption that $\mathfrak{p} > \mathfrak{q}$, we can find the probability that the attacker's chain can catch up benign chain:

$$\mathbb{P}(\mathfrak{z}, \mathfrak{p}, \mathfrak{q}) = 1 - \sum_{i=0}^{\mathfrak{z}} \frac{\mathfrak{T}^i e^{-\mathfrak{T}}}{i!}(1 - (q/p)^{\mathfrak{z}-i}) \tag{13}$$

Here, $\mathbb{P}$ represents the probability that the attacker successfully reverses a previously initiated transaction. In the PPCC algorithm, the selection of packaging nodes is based on verifying the similarity between local models and the global model. Consequently, even if an attacker controls 51% of the nodes, they cannot gain the block packaging rights, which makes PPCC effectively resisting the 51% attack.

Double Spend Attack. A double spend attack is achieved by the attacker through the creation of a new, longer chain, thereby gaining control of the blockchain. In our proposed PPCC consensus algorithm,

the security of the blockchain is ensured through the representative node system, which randomly selects packing nodes, limits the number of nodes with packing rights, and selects representative nodes by comparing the linear similarity between the local and global models. This prevents double spend attacks.

5.2. Experiment results

5.2.1. Runtime environment

Our experiments were conducted on a server, using PyTorch version 2.0.0, CUDA version 11.8, with a system equipped with 56GB of RAM and a 30GB hard disk, the GPUs in the server are Tesla T4 (with a VRAM of 16GB).

5.2.2. Datasets

- MNIST: The MNIST dataset comprises handwritten numbers from 250 different individuals, totaling 70,000 images. The training set contains 60,000 images, while the test set contains 10,000 images. All images are grayscale and have dimensions of 28×28 pixels, with each image featuring a single handwritten digit.
- Fashion-MNIST: The Fashion-MNIST dataset consists of 70,000 frontal images of various fashion products across 10 categories. The dataset's size, format, and division into training and test sets are consistent with the MNIST dataset. It also follows the 60,000/10,000 division for training and testing, with images of 28×28 pixels in grayscale.

5.2.3. Attacks

1. Noise attack: The noise attacks put random noise variance to the update model or local iterates.
2. Sign-flipping Attack(SF) [39]: The sign-flipping attacks flip the signs of update parameters or local iterates and enlarge the magnitudes.

5.2.4. Baselines

Krum: The Krum [40] aggregation algorithm selects the centroid of the local model parameters as the global model. It considers the Euclidean distance between malicious parameters and benign parameters to be large, while the Euclidean distance between two benign parameters is small.

Median: Median [41], proposed in 2021, aggregates input parameters by calculating the median of the values for each dimension of the local parameters. The algorithm collects and sorts the parameters in each dimension. For each dimension, the median parameters are selected as the parameters of the global model in that dimension.

Mean: Mean aggregates its input parameters by calculating the mean value of each dimension of local parameters.

Clippedclustering: Clippedclustering [42] proposed in 2023, designed an automated clipping strategy to defend against potentially amplifying malicious updates.

5.2.5. Accuracy

In this section, we evaluate the performance of PSA in both noise attack and sign-flipping attack scenarios. We compare the performance of PSA with other aggregation algorithms, namely Krum, Median, Mean, and Clippedclustering, in the context of BRFL. Table 2 summarizes the default configuration of the experiments.

In Fig. 3, we present the results of testing five different aggregation algorithms under noise attack and sign-flipping attack scenarios. The evaluations are conducted on the MNIST and Fashion-MNIST datasets, considering both independent and identically distributed (IID) data distribution and non-IID data distribution. Additionally, we set the number of malicious nodes to be half of the total number of local training nodes, which is five malicious nodes in total. The total number of LT and AN is 12 nodes, while the number of BN is 2. For each round, 2 out of the 12 nodes are selected as aggregation nodes, and we introduce 5 malicious nodes into the system.

Table 2
Model parameters.

parameter	value
Batch size of client's local training	10
Optimizer of client's local training	SGD
Learning Rate of client's local training	0.01
Local Epochs of client's local training	5
Max running round of client's local training	100
Noise Variance(in noise attack)	1
Numer of Spectral Clusters (in PSA)	2
Number of aggregation nodes (in PPCC)	2
Number of local training nodes	10
Number of blockchain network nodes	2
Initial stake for blockchain nodes	0
Local training total stake reward	20
Aggregation stake reward	2

Fig. 3a illustrates the accuracy comparison between our proposed PSA and the baselines when using the MNIST dataset with IID data distribution and noise attack as the model poison attack. From Fig. 3a, we observe that our PSA achieves the highest accuracy among all the algorithms. Krum, median, mean, and Clippedclustering exhibit lower performance. Similarly, in Fig. 3b, which represents the performance under sign-flipping attacks on the MNIST dataset with IID data distribution, PSA and Krum outperform the other baselines. However, when the dataset distribution changes from IID to Non-IID, PSA maintains a consistently high performance, while Krum's accuracy decreases compared to PSA. PSA demonstrates superior performance in the Non-IID data distribution. In Fig. 3c, Krum, median, mean, and Clippedclustering exhibit stable performance, but PSA outperforms them all. Fig. 3d shows that Clippedclustering exhibits large fluctuations, remaining in the lower range. Apart from PSA, the accuracy of the four baselines is below 0.5.

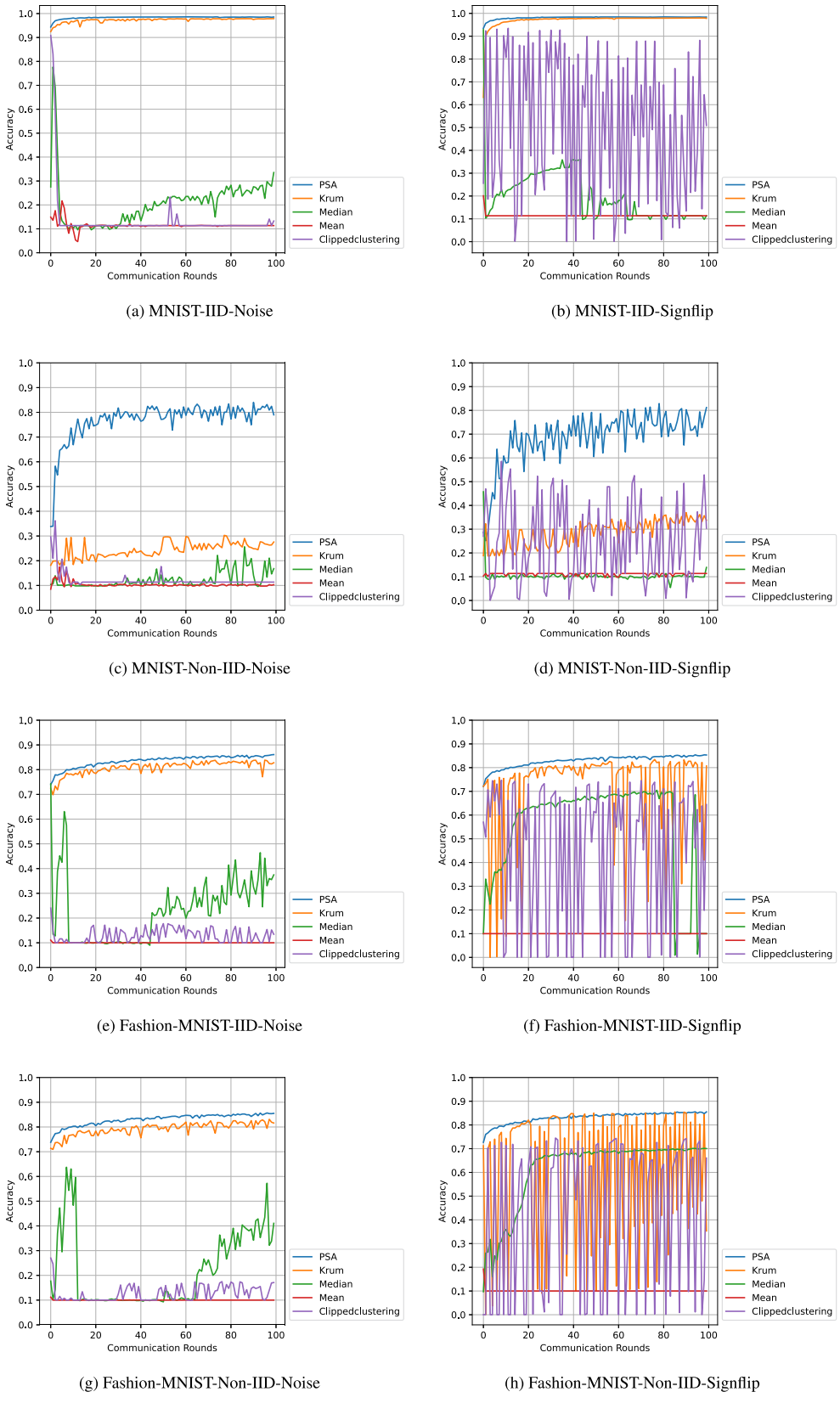
When using the Fashion-MNIST dataset, as shown in Fig. 3e, PSA achieves the best accuracy when the data distribution is IID and the attack type is noise. Krum performs slightly worse than PSA but still above 0.8. The accuracy of the other three baselines is below 0.5. In Fig. 3f, with the attack type changing to sign-flipping compared to Fig. 3e, PSA maintains stable and high performance. Krum and Clippedclustering exhibit large fluctuations, while median remains stable for the first 80 rounds of training but starts to fluctuate later. The mean aggregation algorithm performs poorly. When the data distribution is Non-IID, as shown in Figs. 3g and 3h, PSA outperforms Krum and the other three baselines. In Fig. 3h, with sign-flipping attack as the attack type, PSA maintains high and stable performance, while Krum and Clippedclustering exhibit large fluctuations and fail to surpass PSA. Median achieves an accuracy of 0.7, whereas the mean algorithm remains at 0.1.

5.2.6. Stake trends

In this section, we discuss the stake trends of different types of nodes in BRFL under various datasets, data distributions, and attack types. Node's stake benefits from PPCC in BRFL for model training, aggregation and block package.

The labels of Fig. 4 consist of two parts: the client's name and whether the client is malicious. The nodes are named from *device_1* to *device_14*, except for *device_2* and *device_4*, which are blockchain network nodes. Thus, we did not count their stakes. The letter *B* after the client's name indicates a benign client, while *M* signifies a malicious client. We use dots to represent the stakes of benign nodes and circles to represent the stakes of malicious nodes. From Fig. 4, we observe that the stakes of benign nodes show a nonlinear increasing trend, while the stakes of malicious nodes remain stable around 0.

From Fig. 4a, we can see that the benign client named *device_12* has the largest number of stakes in the final result. This suggests that it either acts as an aggregation node or its local model (lm) is frequently selected for the AN, resulting in the largest stake benefit from the consensus algorithm due to its effective benign local training work. The client



(a) MNIST-IID-Noise (b) MNIST-IID-Signflip

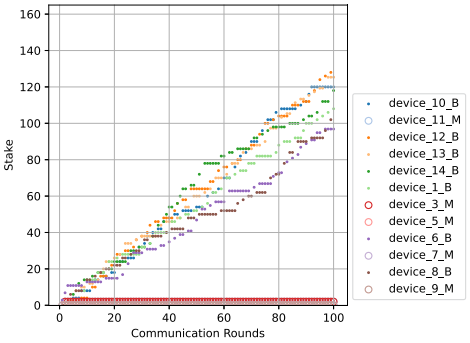
(c) MNIST-Non-IID-Noise (d) MNIST-Non-IID-Signflip

(e) Fashion-MNIST-IID-Noise (f) Fashion-MNIST-IID-Signflip

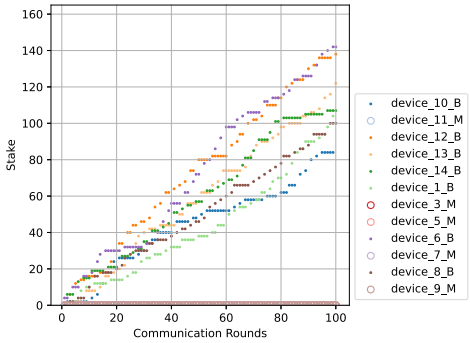
(g) Fashion-MNIST-Non-IID-Noise (h) Fashion-MNIST-Non-IID-Signflip

Fig. 3. Accuracy of BRFL, Krum, Median, Mean, Clippedclustering for MNIST and Fashion-MNIST task.

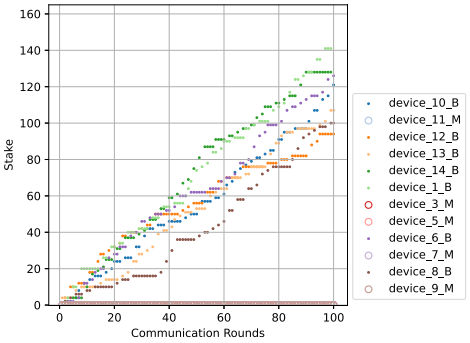
named *device_13* has the second-largest number of stakes. In the last three rounds, its stake remains unchanged, indicating that although it participates in local model training, its lm does not become the AN, and it does not become an aggregation node in these three rounds, resulting in its stake maintaining a constant value. nodes *device_11*, *device_3*, *device_5*, *device_7*, and *device_9* are malicious nodes, and their stakes are around 0, indicating that their generated lm rarely or never enters the AN, which validates the effectiveness of our proposed PPCC. The fig-



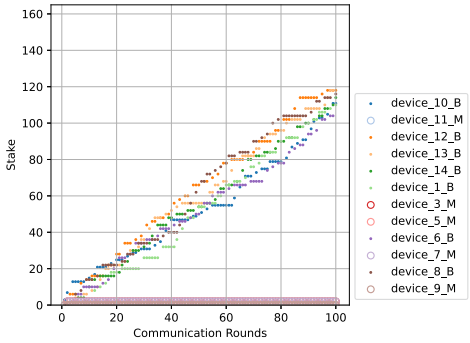
(a) MNIST-IID-Noise



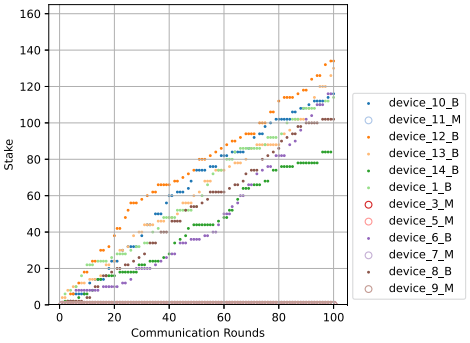
(b) MNIST-IID-Signflip



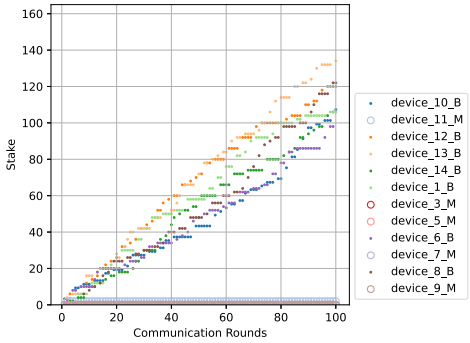
(c) MNIST-Non-IID-Noise



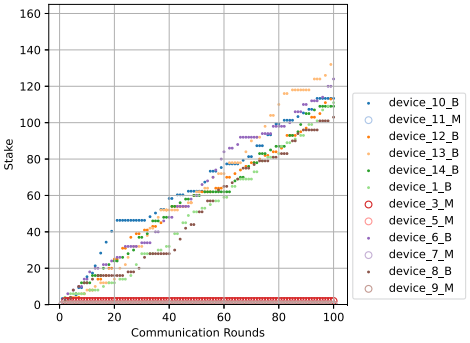
(d) MNIST-Non-IID-Signflip



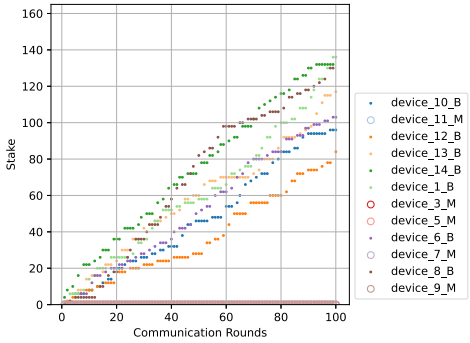
(e) Fashion-MNIST-IID-Noise



(f) Fashion-MNIST-IID-Signflip

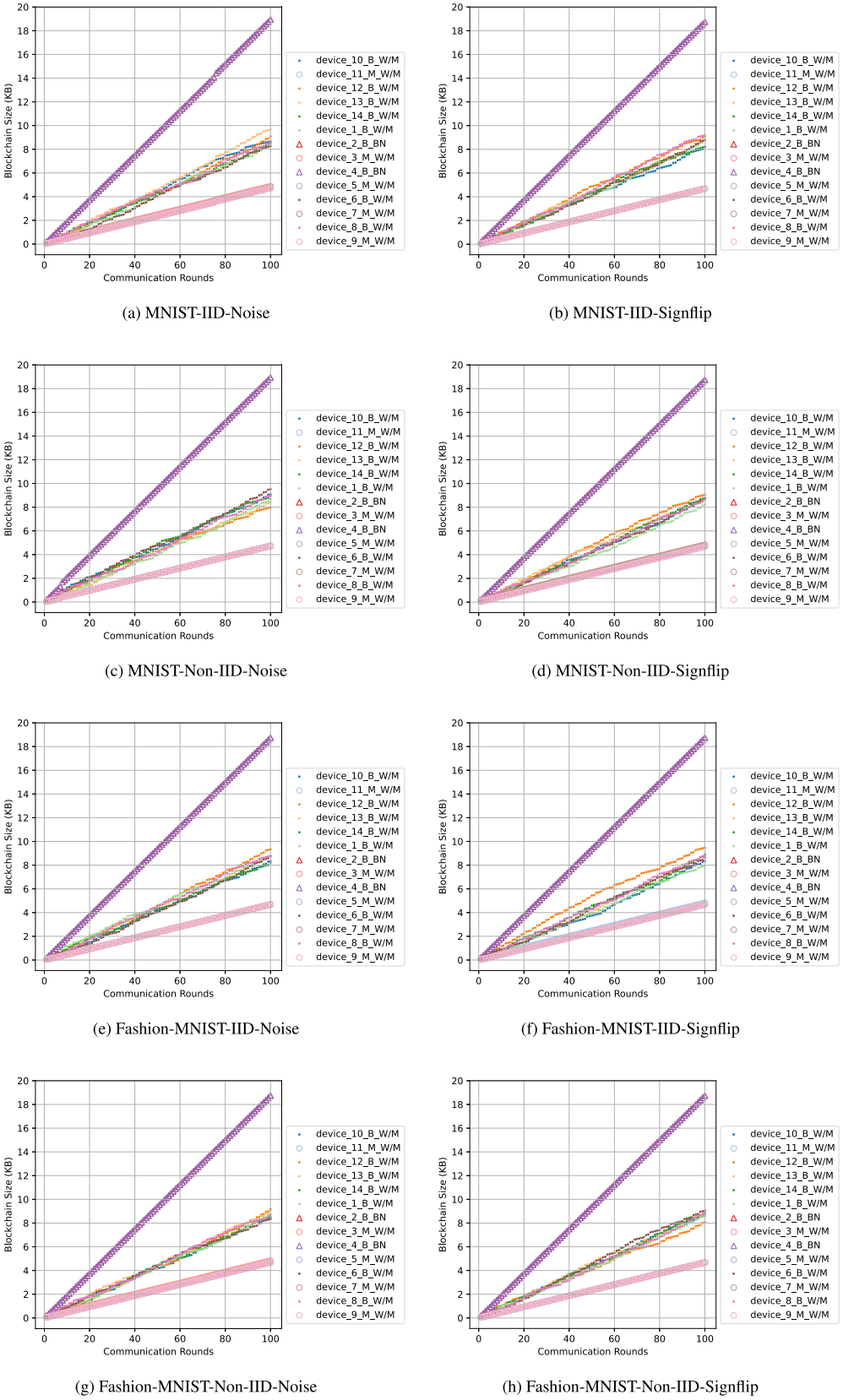


(g) Fashion-MNIST-Non-IID-Noise



(h) Fashion-MNIST-Non-IID-Signflip

Fig. 4. Stake of BRFL for MNIST and Fashion-MNIST task.



(a) MNIST-IID-Noise (b) MNIST-IID-Signflip

(c) MNIST-Non-IID-Noise (d) MNIST-Non-IID-Signflip

(e) Fashion-MNIST-IID-Noise (f) Fashion-MNIST-IID-Signflip

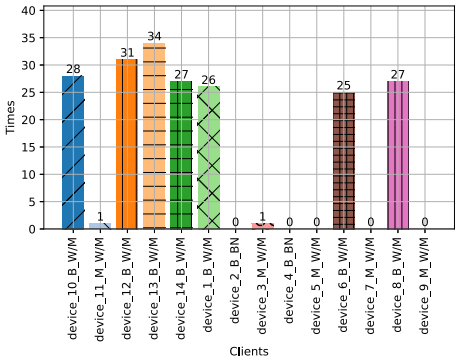
(g) Fashion-MNIST-Non-IID-Noise (h) Fashion-MNIST-Non-IID-Signflip

Fig. 5. Blockchain size of BRFL with two types of *byzantine-attack* for MNIST and Fashion-MNIST tasks.

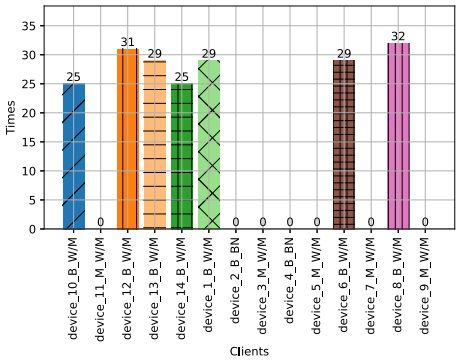
ures in Fig. 4 correspond to Fig. 3 and depict the stake trends under different datasets, data distributions, and attack types when malicious nodes account for 5/12 of the total.

5.2.7. *Blockchain size trends*

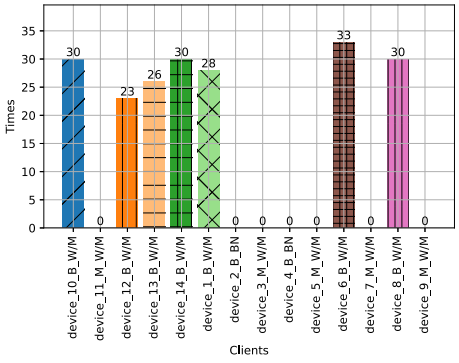
In this section, we discuss the trend of blockchain size in nodes within BRFL in the same environment as depicted in Fig. 3 and Fig. 4. In



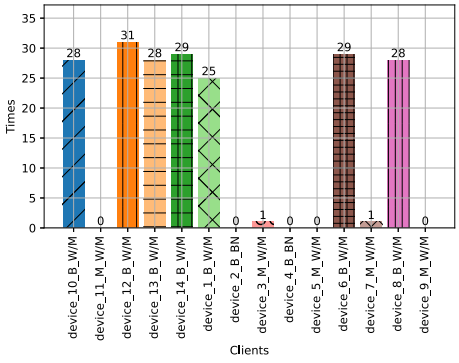
(a) MNIST-IID-Noise



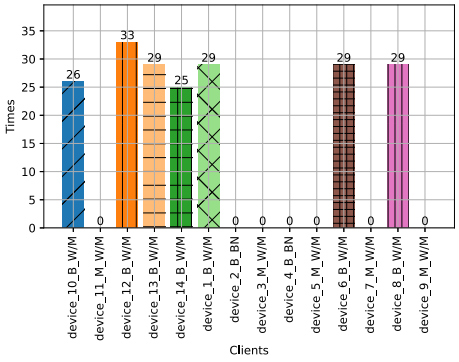
(b) MNIST-IID-Signflip



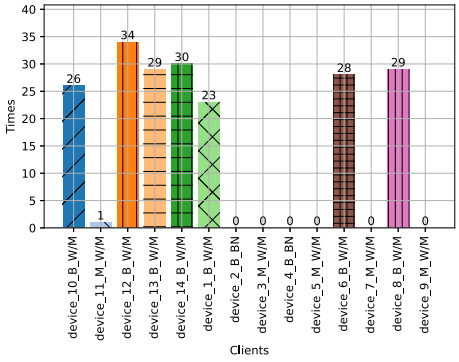
(c) MNIST-Non-IID-Noise



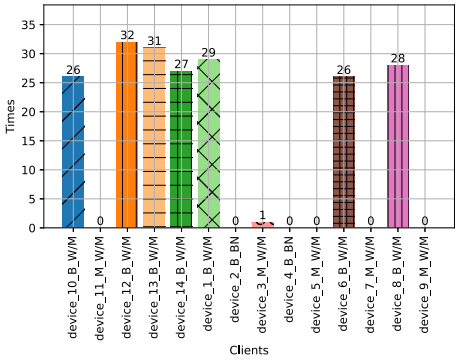
(d) MNIST-Non-IID-Signflip



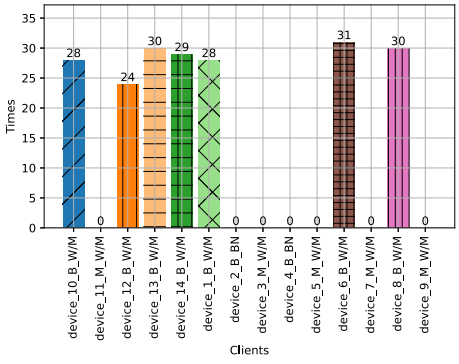
(e) Fashion-MNIST-IID-Noise



(f) Fashion-MNIST-IID-Signflip



(g) Fashion-MNIST-Non-IID-Noise



(h) Fashion-MNIST-Non-IID-Signflip

Fig. 6. Frequency of nodes in BRFL as aggregation nodes with two types of byzantine attacks for MNIST and Fashion-MNIST tasks.

Fig. 5, we examine the variation in the size of the blockchain footprint stored on different devices.

We use triangles to represent blockchain network nodes. In BRFL, we set both blockchain network nodes and aggregation nodes store all types of blocks, while local training nodes only need to store blocks of the global model type. Different colored dots represent benign nodes, while different colored circles represent malicious nodes. The label in Fig. 5 consists of three parts: device name, whether it is a malicious node, and the node type. The device names range from *device_1* to *device_14*, with *B* denoting a benign node, *M* denoting a malicious node, *W/M* denoting a node that can be either a local training node or an aggregation node, and *BN* indicating that the node is a blockchain network node that does not participate in the training and aggregation of federated learning and only exists in the blockchain network.

Fig. 5 reveals that the blockchain size stored by the blockchain network nodes exhibits a linear growth trend. This is because the blockchain network nodes store all types of blocks in the blockchain, and each round of the federated learning process produces blocks that are of the same or similar size. Among the nodes of type *W/M*, the block size of the benign nodes shows a nonlinear increasing trend. This is because when a node becomes an aggregation node, it stores all types of blocks, whereas as a local training node, it only stores one type of block. As the benign node alternates between being an aggregation node and a local training node, the size of the blockchain it stores shows a nonlinear increasing trend. The blockchain size in malicious nodes also shows a linear increasing trend, but the increase is smaller. This is because in the BRFL model, malicious models are detected, and malicious nodes are prevented from becoming aggregation nodes. Therefore, malicious nodes only store blocks of the global model type as local training nodes.

5.2.8. Number of being aggregation nodes times trends

In this section, we discuss the number of times local training nodes are selected as aggregation nodes using the PPCC consensus algorithm, as shown in Fig. 6, within the same environment as Fig. 3, 4, and 5.

Fig. 6 is a bar chart that represents the number of times different nodes are selected to become aggregation nodes during federated learning. The x-axis represents the different nodes, and the y-axis represents the number of times a client has been selected as an aggregation node, with corresponding numbers labeled in the figure.

As seen in Fig. 6, blockchain nodes, such as *device_2_B_BN* and *device_4_B_BN*, do not participate in the federated learning process, so their count is 0. Among the nodes of type *W/M*, the number of times benign nodes become aggregation nodes is relatively evenly distributed, without excessive or under-representation. This is due to the fact that the PPCC algorithm does not allow continuous selection as an aggregation node each time.

In most cases, the number of times malicious nodes become aggregation nodes is 0. However, there are still a small number of cases where they become aggregation nodes. This may occur when there is no global model at the initialization of federated learning or when the global model is not yet perfect, resulting in random selection of nodes as aggregation nodes during the first round. However, as observed from the Fig. 6, the occurrences of malicious nodes becoming aggregation nodes are minimal. This indicates that the PPCC algorithm effectively prevents malicious nodes from being selected as aggregation nodes.

6. Conclusion and future work

In this work, we propose BRFL, a blockchain-based Byzantine-robust federated learning model. Our objective is to address the issues of *byzantine-attack* and *resource-cost* during model aggregation. The *resource-cost* problem can be further divided into computation resource costs and storage resource costs. To achieve this, we introduce two key components: the Proof of Pearson Correlation Coefficient (PPCC) consensus algorithm and the Precision-based Spectral Aggregation (PSA) algorithm. These components verify the accuracy of the average parameters by clustering the local model parameters and store federated learning communication records in the blockchain. Additionally, we utilize IPFS to store the local models and allow nodes to selectively store blocks, effectively managing storage requirements. Compared to the existing mainstream methods, our proposed method has higher accuracy and security in resisting Byzantine attacks, while improving the blockchain and reducing the consumption of computational resources required by the consensus algorithm and the pressure on block storage. Experimental results demonstrate that BRFL maintains Byzantine robustness with reduced resource costs.

We propose a new aggregation algorithm to improve the security of federated learning, using blockchain as the basis of federated learning network. However, since the blockchain is a peer-to-peer network and the transmitted data is open and transparent, there are still need to be improved in terms of data privacy and communication efficiency. Meanwhile, some computation is required in our proposed aggregation algorithm, which is an area where our work is inadequate. In our future work, we plan to enhance the privacy security of the local model and the global model in federated learning, improve the communication efficiency of the blockchain network, reduce the computational steps required by the aggregation algorithm when constructing the blockchain-based federated learning model.

CRediT authorship contribution statement

Yang Li: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Chunhe Xia:** Supervision, Funding acquisition. **Chang Li:** Writing – review & editing, Validation. **Tianbo Wang:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant No. 62272024.

Data availability

The authors do not have permission to share data.

References

[1] H.B. McMahan, E. Moore, D. Ramage, S. Hampson, B.A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: International Conference on Artificial Intelligence and Statistics, 2016.
[2] T. Li, A.K. Sahu, A. Talwalkar, V. Smith, Federated learning: challenges, methods, and future directions, IEEE Signal Process. Mag. 37 (2019) 50–60.
[3] M. Fang, X. Cao, J. Jia, N.Z. Gong, Local model poisoning attacks to byzantine-robust federated learning, in: USENIX Security Symposium, 2019.
[4] Y. Qu, M.P. Uddin, C. Gan, Y. Xiang, L. Gao, J. Yearwood, Blockchain-enabled federated learning: a survey, ACM Comput. Surv. 55 (2022) 1–35.
[5] Y. Liu, W. Yu, Z. Ai, G. Xu, L. Zhao, Z. Tian, A blockchain-empowered federated learning in healthcare-based cyber physical systems, IEEE Trans. Netw. Sci. Eng. 10 (2023) 2685–2696.
[6] Y. Chen, L. Su, J. Xu, Distributed statistical machine learning in adversarial settings: byzantine parameters descent, in: PERV, 2017.
[7] D. Cao, S. Chang, Z. Lin, G. Liu, D. Sun, Understanding distributed poisoning attack in federated learning, in: 2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS), 2019, pp. 233–239.
[8] Y. Khazbak, T. Tan, G. Cao, Mlguard: mitigating poisoning attacks in privacy preserving distributed collaborative learning, in: 2020 29th International Conference on Computer Communications and Networks (ICCCN), 2020, pp. 1–9.

[9] C. Fung, C.J.M. Yoon, I. Beschastnikh, The limitations of federated learning in sybil settings, in: International Symposium on Recent Advances in Intrusion Detection, 2020.
[10] Y. Wang, T. Zhu, W. Chang, S. Shen, W. Ren, Model poisoning defense on federated learning: a validation based approach, in: International Conference on Network and System Security, 2020.
[11] J. Tan, Y.-C. Liang, N.C. Luong, D.T. Niyato, Toward smart security enhancement of federated learning networks, IEEE Netw. 35 (2020) 340–347.
[12] Z. Chen, P. Tian, W. Liao, W. Yu, Zero knowledge clustering based adversarial mitigation in heterogeneous federated learning, IEEE Trans. Netw. Sci. Eng. 8 (2021) 1070–1083.
[13] Y. Miao, Z. Liu, H. Li, K.R. Choo, R.H. Deng, Privacy-preserving byzantine-robust federated learning via blockchain systems, IEEE Trans. Inf. Forensics Secur. 17 (2022) 2848–2861.
[14] G. Bian, W. Qu, B. Shao, Blockchain-based trusted federated learning with pretrained models for covid-19 detection, Electronics (2023).
[15] Z. Shu, H. Zhao, B. Xu, W. Xun, B. he Xu, Privacy-preserving federated learning framework via blockchain and committee mechanism, in: 2023 IEEE 23rd International Conference on Communication Technology (ICCT), 2023, pp. 1269–1274.
[16] G.A. Kamhoua, E. Bandara, P.B. Foytik, P. Aggarwal, S. Shetty, Resilient and verifiable federated learning against byzantine colluding attacks, in: 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), 2021, pp. 31–40.
[17] X. Yan, Y. Miao, X. Li, K.R. Choo, X. Meng, R.H. Deng, Privacy-preserving asynchronous federated learning framework in distributed iot, IEEE Int. Things J. 10 (2023) 13281–13291.
[18] R. Wang, W.T. Tsai, Asynchronous federated learning system based on permissioned blockchains, Sensors 22 (2022).
[19] S. Hu, J. Li, Q. Zhao, C. Zhang, Z. jian Zhang, Y. Shi, Blockdl: privacy-preserving and crowd-sourced deep learning through blockchain, in: 2021 IEEE Symposium on Computers and Communications (ISCC), 2021, pp. 1–7.
[20] T. Muazu, Y. Mao, A.U. Muhammad, M. Ibrahim, S. Omaji, P. Tiwari, Iomt: a medical resource management system using edge empowered blockchain federated learning, IEEE Trans. Netw. Serv. Manag. 21 (2024) 517–534.
[21] E. Moore, A. Imteaj, S. Rezapour, S.M.I.M.H. Amini, M.H. Amini, F.A. Imteaj, A survey on secure and private federated learning using blockchain: theory and application in resource-constrained computing, IEEE Int. Things J. 10 (2023) 21942–21958.
[22] X. Fu, R. Peng, W. Yuan, T. Ding, Z. Zhang, Y. Peng, M. Kadoch, Federated learning-based resource management with blockchain trust assurance in smart iot, Electronics (2023).
[23] J. Zhang, Y. Liu, X. Qin, X. Xu, P. Zhang, Adaptive resource allocation for blockchain-based federated learning in Internet of things, IEEE Int. Things J. 10 (2023) 10621–10635.
[24] Y. Dai, H. Yang, H. Yang, Deep reinforcement learning for resource allocation in blockchain-based federated learning, in: ICC 2023 - IEEE International Conference on Communications, 2023, pp. 179–184.
[25] Z. Wang, Q. Hu, Z. Xiong, Y. Liu, D. Niyato, Resource optimization for blockchain-based federated learning in mobile edge computing, IEEE Int. Things J. (2023) 1.
[26] Z. Wang, Q. Hu, R. Li, M. Xu, Z. Xiong, Incentive mechanism design for joint resource allocation in blockchain-based federated learning, IEEE Trans. Parallel Distrib. Syst. 34 (2022) 1536–1547.
[27] S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, 2008.
[28] H. Xiong, M. Chen, C. Wu, Y. Zhao, W. Yi, Research on progress of blockchain consensus algorithm: a review on recent progress of blockchain consensus algorithms, Future Internet 14 (2022) 47.
[29] K. Farooq, H.J. Syed, S.O. Alqahtani, W.A.E. Nagmeldin, A.O. Ibrahim, A. Gani, Blockchain federated learning for in-home health monitoring, Electronics (2022).
[30] S. Aich, N.K. Sinai, S. Kumar, M. Ali, Y.R. Choi, M.-I. Joo, H.-C. Kim, Protecting personal healthcare record using blockchain & federated learning technologies, in: 2021 23rd International Conference on Advanced Communication Technology (ICACT), 2021, pp. 109–112.
[31] W. Jiao, H. Zhao, P. Feng, Q. Chen, A blockchain federated learning scheme based on personalized differential privacy and reputation mechanisms, in: 2023 4th International Conference on Information Science, Parallel and Distributed Systems (ISPDS), 2023, pp. 630–635.
[32] K. Pearson, Contributions to the mathematical theory of evolution, Proc. R. Soc. Lond. 54 (1894) 329–333.
[33] U. von Luxburg, A tutorial on spectral clustering, Stat. Comput. 17 (2007) 395–416.
[34] C. Hu, J. Jiang, Z. Wang, Decentralized federated learning: a segmented gossip approach, arXiv:1908.07782 [abs], 2019, Online.
[35] P. Kairouz, H.B. McMahan, B. Avent, A. Bellet, M. Bennis, A.N. Bhagoji, K. Bonawitz, Z.B. Charles, G. Cormode, R. Cummings, R.G.L. D'Oliveira, S.Y.E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P.B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konecný, A. Korolova, F. Koushanfar, O. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, M. Raykova, H. Qi, D. Ramage, R. Raskar, D.X. Song, W. Song, S.U. Stich, Z. Sun, A.T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F.X. Yu, H. Yu, S. Zhao, Advances and open problems in federated learning, Found. Trends Mach. Learn. 14 (2019) 1–210.
[36] X. Liu, H. Li, G. Xu, Z. Chen, X. Huang, R. Lu, Privacy-enhanced federated learning against poisoning adversaries, IEEE Trans. Inf. Forensics Secur. 16 (2021) 4574–4588.
[37] S.M.S. Saad, R.Z.R.M. Radzi, Comparative review of the blockchain consensus algorithm between proof of stake (pos) and delegated proof of stake (dpos), Int. J. Innov. Comput. 10 (2) (2020).
[38] J. Xu, S.-L. Huang, L. Song, T. Lan, Byzantine-robust federated learning through collaborative malicious gradient filtering, in: 2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS), 2022.
[39] L. Li, W. Xu, T. Chen, G.B. Giannakis, Q. Ling, Rsa: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets, 2018.
[40] P. Blanchard, E.M.E. Mhamdi, R. Guerraoui, J. Stainer, Machine learning with adversaries: byzantine tolerant gradient descent, in: Neural Information Processing Systems, 2017.
[41] D. Yin, Y. Chen, K. Ramchandran, P.L. Bartlett, Byzantine-robust distributed learning: towards optimal statistical rates, in: International Conference on Machine Learning, 2018.
[42] S. Li, E.C.H. Ngai, T. Voigt, An experimental study of byzantine-robust aggregation schemes in federated learning, arXiv:2302.07173 [abs], 2023.

Yang Li received the master's degree from the Zhengzhou University, Zhengzhou, Henan, China, in 2021. He is currently pursuing the Ph.D. degree with Beihang University, Beijing, China, under the supervision of Prof. Chunhe Xia. His research interests include blockchain and federated learning.

Chunhe Xia received the Ph.D. degree in computer application from Beihang University, Beijing, China, in 2003. He is currently a Supervisor and a Professor at Beihang University and the Director of the Beijing Key Laboratory of Network Technology. He has participated in different national major research projects and published more than 70 research papers in important international conferences and journals. His current research interests include network and information security, information countermeasure, cloud security, and network measurement.

Chang Li is currently pursuing the bachelor degree with Zhengzhou University of Light Industry, Zhengzhou, China. His research interests include cyber security, machine learning.

Tianbo Wang received the Ph.D. degree in computer application from Beihang University, Beijing, China, in 2018. He is currently an Associate Professor with Beihang University. He has participated in several National Natural Science Foundations and other research projects. His research interests include network and information security, intrusion detection technology, and information countermeasure.