High-Quality Model Aggregation for Blockchain-Based Federated Learning via Reputation-Motivated Task Participation

Jiahao Qi, Feilong Lin, *Member, IEEE*, Zhongyu Chen, Changbing Tang, *Member, IEEE*,
Riheng Jia, and Minglu Li, *Senior Member, IEEE*

***Abstract*—Federated learning is an emerging paradigm to conduct the machine learning collaboratively but avoid the leakage of original data. Then, how to motivate the data owners to participate federated learning and contribute high-quality data is the crucial issue. In this article, a blockchain-based federated learning (BFL) with a reputation mechanism for high-quality model aggregation is proposed. Specifically, the blockchain transforms the federated learning into a decentralized and trustworthy manner. Over the blockchain, federated learning tasks, undertaken by smart contracts, can be conducted transparently and fairly. Besides, a reputation-constrained data contribution and reward allocation mechanism is designed to encourage data owners to participate in BFL and contribute high-quality data. The noncooperative game is adopted to analyze the behavior strategies of data owners. The existence of the unique equilibrium is proved and the equilibrium point indicates that the data owners can acquire highest reward with the contribution of the highest quality data. Thus, the model quality of BFL is guaranteed. Finally, simulations on the public data sets (MNIST and CIFAR10) demonstrate that BFL with a reputation mechanism can well promote the high-quality model aggregation of federated learning as well as can prevent malicious nodes from corrupting the training task.**

***Index Terms*—Blockchain, federated learning, high-quality model, reputation mechanism, smart contract.**

I. Introduction

MACHINE learning can be easily used to train the functional model by collecting amount of data but without logical reasoning or complex computing. It has been extensively applied for, such as image understanding in traffic control [1], route planning in autonomous driving [2], intelligent diagnosis in medical treatment [3], speech information recognization and reconstruction [4], and so on. However, in many application scenarios, it is faced the security issue that privacy contents, such as personal daily route, health state, and even identity information, may be leaked, which will result in more serious consequences.

Recently, to deal with this security issue from machine learning, McMahan *et al.* [5], [6] from Google proposed a new paradigm called federated learning. Federated learning no longer trains the model using centrally stored data. It calls for participants to train the model locally using self-owned data and send the local models without raw data to the model aggregation server. Finally, the model aggregation server can obtain an accurate global model meanwhile keep the personal data secure. Since then, federated learning has received great attention from both academia and industrial circles.

Naturally, a critical problem faced by federated learning is how to encourage the data owners to join in the tasks, since that data owners have to contribute both data and local model training overhead in this new paradigm. An incentive mechanism is needed to motivate data owners to take part in federated learning tasks while pursuing the high quality of the federated learning model. The common approach is to reward each participant according to their respective contribution. A lot of work on incentive mechanisms have been done [7], such as individual profit sharing [8], Shapley game profit sharing [9], and fair-value game [10]. Nevertheless, how to design incentives for the appropriate federated learning scenario is still to be investigated. Furthermore, in such a distributed federated learning, the task may be attacked by malicious participants by such as giving weak or wrong local model or overstating their contribution. To this point, the appropriate model quality evaluation mechanism [11] and task participant management rules are necessary. Commonly, the reputation-based management rules [12], [13] together with reward mechanism can be leveraged to encourage task participants with honest behavior.

Although federated learning prevents the raw data leakage, the central aggregation server also faces the risk of single-point failure. Fortunately, the newly emerging blockchain technology has the potential to address this issue. Blockchain is essentially a collectively maintained data ledger by the peer-to-peer network. It built a new decentralized trust without a third-party credit endorsement [14]. In particular, with consensus protocol, the data ledger over blockchain network

Manuscript received 14 September 2021; revised 16 February 2022; accepted 14 March 2022. Date of publication 18 March 2022; date of current version 23 September 2022. This work was supported in part by the NSF of China under Grant 61877055 and Grant 61902358, and in part by the NSF of Zhejiang Province under Grant LY22F030006. *(Corresponding author: Feilong Lin.)*

Jiahao Qi, Feilong Lin, Zhongyu Chen, Riheng Jia, and Minglu Li are with the Key Laboratory of Intelligent Education Technology and Application of Zhejiang Province and the College of Mathematics and Computer Science, Zhejiang Normal University, Jinhua 321004, China (e-mail: qjh2020@zjnu.edu.cn; bruce_lin@zjnu.edu.cn; czy@zjnu.edu.cn; rihengjia@zjnu.edu.cn; mlli@zjnu.edu.cn).

Changbing Tang is with the College of Physics and Electronic Information Engineering, Zhejiang Normal University, Jinhua 321004, China (e-mail: tangcb@zjnu.edu.cn).

Digital Object Identifier 10.1109/JIOT.2022.3160425

exhibits the features in terms of tamper-resistant, nonrepudiable, and publicly verifiable [15]. The smart contract technology [16] helps convert the traditional businesses to a blockchain network environment and conduct the businesses with a transparent, traceable, and unforgeable manner. For example, Ferrag *et al.* [17], [18], and Xia [19] summarized in detail the application of blockchain in the context of IoT and considered very important factors, such as security issues, efficiency issues, or in the choice of consensus algorithms in the application process. To sum up, blockchain can be used to solve the centralized services with high security risk or high service charges [20].

In this article, with the purpose of increasing the model quality of federated learning, the integration design of blockchain technology and federated learning is considered. Together with smart contract technology, blockchain can build a trustworthy environment and make the federated learning tasks be executed transparently and fairly. Specifically, the local models contributed by data owners can be publicly evaluated. The global model aggregation can be decentrally fulfilled with properly designed blockchain consensus mechanism, thus to make the global model aggregation safe. Moreover, the reputation mechanism can be constructed in blockchain environment. Based on the evaluation of the contributed local model, one will be assigned with a certain reputation degree. Then, data owners can be rewarded with the consideration of their contributed data volume and reputation degree. With the above ideas, the data owners are supposed to have fine motivation to join the federated learning and contribute their high-quality data to learn the local model. The innovation and contribution of this article are summarized as follows.

1) A blockchain-based federated learning (BFL) mechanism is proposed. It is supposed to solve the single-point-failure security problem of the model aggregation center in traditional federated learning systems. Systematic smart contracts are designed to conduct federated learning over blockchain network. BFL is capable to improve the credibility and reliability of federated learning.
2) A reputation mechanism is designed based on the model quality and contribution evaluation, which also builds the foundation to coordinate the effective data contribution for local model, weighted global model aggregation, and reward allocation. The above measures can effectively encourage the data owners to join the BFL. Additionally, an optional grouping mechanism is proposed to cope with the high complexity brought by a large number of participants.
3) The decentralized execution of federated learning task is formulated by the noncooperative game, where the strategy to maximize individual profit is adopted. The unique existence of Nash equilibrium is proved. The equilibrium point shows that the data owner's profit reaches the maximum under the condition of contributing the highest quality data, thereby ensuring the high quality of the federated learning model.
4) Finally, extensive simulations have been conducted based on the public data sets (MNIST and CIFAR10). The results show that BFL can well promote the high-quality model aggregation of federated learning as well as can prevent malicious nodes from corrupting the training task.

The remainder of this article is organized as follows. Section II introduces the related work. The BFL system will be described in Section III. Section IV introduces the model quality-based reputation evaluation mechanism. Then, the reputation-based reward allocation and model aggregation algorithm is presented in Section V. The system implementation and simulation results are shown in Section VI. Finally, Section VII concludes this article.

II. Related Work

Since federated learning was first proposed and applied to the automatic input completion system by Google [5], [6], it has attracted much attention from both academic and industrial areas [21]. Recently, federated learning is constantly exploring applications in different fields. For example, in healthcare area, Elayan *et al.* [22] proposed a deep federated learning framework for sustainable healthcare data monitoring and analysis. Lim *et al.* [23] utilized federated learning to enable privacy-preserving collaborative model training with distributed IoT network. In terms of transportation, Li *et al.* [24] introduced the federated learning into autonomous driving to preserve vehicular privacy by keeping original data locally. Lu *et al.* [25] proposed a federated learning architecture to relieve transmission load and address privacy concerns of providers. In industrial IoT, Lu *et al.* [26] used the privacy-preserved federated learning architecture for secure industrial data modeling and sharing. Fu *et al.* [27] proposed a verifiable federated learning with privacy preserving for industrial big data processing, where the correctness of the aggregated gradients is verified by the Lagrange interpolation method. In addition, Su *et al.* [28] introduced an edge-cloud-assisted federated learning framework for communication-efficient and privacy-preserving energy data sharing and Song *et al.* [29] enhanced the users' identity privacy of federated learning in mobile-edge computing.

In federated learning networks, how to motivate users to participate in task and contribute the high-quality data is one of the most important research topic. Some researchers have paid efforts to address this issue. Fan *et al.* [30] utilized the reverse auction mechanism to motivate the users and introduced the Earth mover's distance model to measure the data with independently identical distribution (IID). Zeng *et al.* [31] designed a lightweight incentive mechanism with multidimensional procurement auction that encouraged edge nodes participating in the mobile-edge computing. Dong and Zhang [32] introduced the game theory to design the strategy that motivated nodes to participate in the model training task. Based on the idea of a noncooperative game, Zhan *et al.* [33] set up a total budget to reward the clients with their respective contributions. Wu *et al.* [34] and Ding *et al.* [35] designed incentive mechanisms for federated learning with the reasonable consideration of various factors, such as task expenditure, communication delay, and users' privacy issue. Sun *et al.* [36]

further proposed a contract-based incentive mechanism that provided compensation for privacy leakage.

In addition to keeping clients motivated, federated learning calls for the distributed and secure method to coordinate cooperation among the users. Blockchain has the potential to well manage the federated learning. Kim *et al.* [37] applied blockchain to coordinate the training process of federated learning and completed the decentralization of the training process. Kang *et al.* [12], [38] utilized smart contract to complete the reputation management of clients, and record the reputation evaluation through the blockchain ledger, thus to realize the tamper proof of the reputation. Qu *et al.* [39] used blockchain to fairly select the nodes participating in specified federated learning tasks in Industry 4.0.

These efforts have focused on how to address the performance, incentive, and management issues of federated learning. However, few works have systematically considered the quality of contributed data, model aggregation, and reward distribution in federated learning tasks. Based on the existing research, this article works toward a high-quality federated learning model by combining data quality with model aggregation and reward distribution.

III. Blockchain-Based Federated Learning

In this section, the federated learning is preliminarily introduced. Then, the illustration of the proposed BFL with reputation mechanism is presented. Smart contracts to conduct federated learning with high-quality model aggregation are finally provided.

A. Preliminaries on Federated Learning

Model aggregation server initializes a global model once a new task is published, which announces the beginning of federated learning. Clients fetch the latest global model from the server and update the model with their own data to produce corresponding local models. The model aggregation server collects these local models and aggregates them into the updated global model. Such a process is considered as a single round. Generally, the federated learning will be carried out with multiple rounds.

Suppose there are N clients denoted by $\mathcal{N} = \{1, 2, \ldots, N\}$. Each client n has a data set $\mathcal{D}_n$ of amount s_n, $\mathcal{D}_n = \{d_i | i = 1, 2, \ldots, s_n\}$. The single data is represented as $d_i = (x_i, y_i)$, where x_i denotes the feature of the data and y_i denotes the label of the data. The model parameters to be trained are denoted by w. For convenience, w will be used later to refer to the model as well. Therefore, the local loss function of client n is

$$F_n(w) = \frac{1}{s_n} \sum_{i=1}^{s_n} f_i(w) \tag{1}$$

where $f_i(w) = \left(x_i^T w - y_i\right)^2$. $f_i(w)$ reflects the distance between the output of the model and the actual result.

For the global model, its global loss function can be formulated as

$$F(w) = \sum_{n=1}^{N} \frac{s_n}{S} F_n(w) = \frac{1}{S} \sum_{n=1}^{N} \sum_{i=1}^{s_n} f_i(w) \tag{2}$$

where $S = \sum_{n \in \mathcal{N}} s_n$. The whole task of federated learning aims at minimizing the global loss function, i.e.,

$$\min_{\omega} \ F(w). \tag{3}$$

In this article, a gradient descent algorithm will be taken to find the best w. The model waiting to be updated in the tth round is denoted as w^t. In the tth round, client n computes its gradient $g_n = \nabla F_n(w^t)$ with the local data set and updates the new local model w_n^t under the learning rate η (a larger learning rate means a faster rate of change of the loss function)

$$w_n^{t+1} \leftarrow w^t - \eta g_n. \tag{4}$$

Client node n uploads w_n^{t+1} to the model aggregation server

$$w^{t+1} \leftarrow \sum_{n=1}^{N} \frac{s_n}{S} w_n^{t+1}. \tag{5}$$

In practical applications, a threshold δ can be set to indicate that the training results meet expectations. When $f(w^t) \leq \delta$, the model has reached a relatively good performance level and the task is finished.

B. Network Model of BFL

In the proposed BFL, federated learning chooses the consortium chain as its blockchain foundation. The consortium chain contains supervisable participants and predefined ledger nodes, enabling faster consensus and more secure operations among nodes. This satisfies the requirements of BFL for node security and task efficiency. The framework of BFL is illustrated by Fig. 1. A description of the members in BFL and their networks is given in the following.

1) *Data Owner:* Such node has data that can be used for model training but cannot be shared externally. The data owner trains the local model with private data and contributes data for a fee by sharing the local model. It is served by client node n in federated learning, $n \in \mathcal{N}$.
2) *Model Aggregator:* In BFL, a part of data is not used for training but for testing. The client nodes holding the test data can act as model aggregators. Suppose that there are M model aggregators, who form the model aggregation organization denoted by $\mathcal{M} = \{1, 2, \ldots, M\}$. Members of the model aggregation organization collaborate with each other and are jointly responsible for model aggregation.
3) *Network Establishment:* In the BFL network, data owners do not communicate directly, but obtain information through synchronous blockchain. A star topology is formed between the model aggregator and the surrounding data owners within the communication range. Model aggregators communicate through broadcasting.

The BFL network can be logically divided into four layers: 1) the blockchain layer; 2) the model layer; 3) the reputation layer; and 4) the reward layer, as follows.

1) *Blockchain Layer:* This layer is the basis of the BFL network. A model aggregator collects the local models of the data owners bound to it via peer-to-peer transfer. In order to get all local models, the model aggregators will share their collected models with each other.

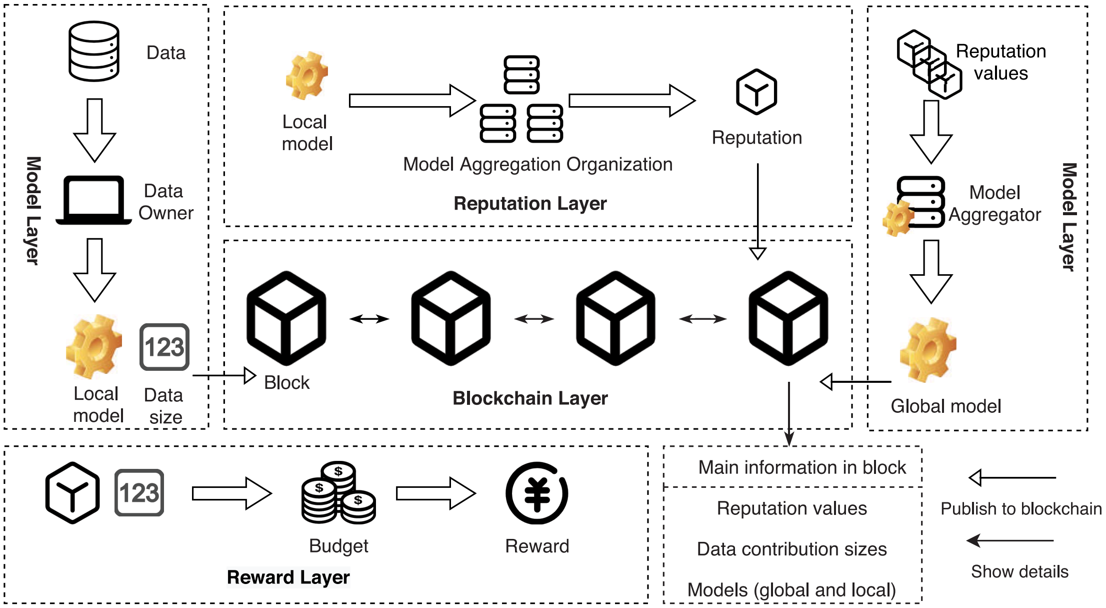


Fig. 1. Illustration of BFL with reputation mechanism.

Blockchain serves as a bridge to share global models and reputation. Data owners get the latest global model and their reputations from the blockchain ledger. After aggregating the new global model, the model aggregator publishes the model and the reputation evaluation of data owners to the blockchain.

2) *Reputation Layer:* The reputation layer completes the evaluation of the data owners' reputation. In the reputation layer, one data owner first shares its local model to all model aggregation nodes through the blockchain layer. Each model aggregation node performs model quality testing of the local model. By exchanging the test results, the model aggregation organization will collaboratively complete a reputation evaluation of this data owner and finally publish the reputation evaluation to blockchain.
3) *Model Layer:* The training part of the federal learning task is performed in the model training layer. Based on the blockchain layer and the reputation layer, the model training layer implements a decentralized model aggregation task combined with reputation. In this layer, after getting the global model, data owners update the model with their local data to get new local models. Then, data owners share their own local models across the blockchain layer. The model aggregation organization aggregates these collected local models with the reputation evaluations into a new global model.
4) *Reward Layer:* In the reward layer, the system will reward data owners with their contributions combining reputations. The existence of this layer encourages the data owner to actively participate in the federated learning task and contribute high-quality data.

C. Smart Contracts for Federated Learning

To enhance the trustworthiness and reliability of BFL, the federated learning tasks are undertaken by smart contracts. During task execution, nodes can complete the task by invoking the corresponding smart contract. The smart contracts for BFL include task initialization contract, member selection contract, federated learning contract, reputation contract, reward allocation contract, and query contract, which are presented in detail in the following.

1) Task Initialization Contract: This contract is invoked through the model aggregator when one user wants to issue a federated learning task to BFL. The contract will assign a unique task ID to this task and initialize the data type, model structure, and payoff budget for this training task. The contract writes the above information to the BFL and automatically invokes the member selection contract after task initialization.

2) Member Selection Contract: Member selection contract is used to determine the data owners to participate in federated training according to their data amount and reputation values. The reputation values of the data owners and the amount of data contributions allowed are updated by invoking the model processing contract. The contract provides the following two interfaces.

1) *MemberInitialization():* This interface can be called by the task initialization contract and will broadcast the information of this task to all data owners. Data owners willing to participate will reply to the contract with the amount of data they contributed in the first round. The contract will receive responses within time T_{msc}, then verify the legitimacy of participating nodes, and submit the reputation, data amount and public key of each node to the BFL network. At this point the reputation of all participating nodes is initialized to 1.
2) *MemberUpdating():* The interface reads the latest information of a node from the block, including the node's reputation value and the maximum amount of data permitted for model training in its next round. The permitted data amount is constrained by the reputation value, which will be defined in Section IV. If the node's

maximum amount of data is less than a threshold value $s_{\min}$, the node will be removed from the candidate list.

3) Federated Learning Contract: Before training the local model, data owners check the latest member list to confirm their eligibility and to obtain their reputation and the maximum amount of data allowed to contribute. The federated learning contract is divided into the following interfaces: local model uploading and downloading, model testing, model aggregation, and global model uploading and downloading.

1) *LocalModelUploading():* When a data owner trains a local model, it can upload the model parameters and data amount to the bound model aggregator by calling this interface.
2) *LocalModelDownloading():* During a local model training cycle T_{local}, the model aggregators call this interface to collect the local models. At the end of T_{local}, the local models collected by each other will be shared within the model aggregation organization.
3) *TestResultUploading():* When the model aggregation organization collects all the uploaded information or satisfies the collection cycle, the aggregators collaborate with each other to perform model quality testing for data owners. When the test is finished, this interface will be called to upload the test results and the reputation contract is invoked based on the testing results.
4) *ModelAggregation():* Model aggregators can aggregate local models into global models by calling this interface. The model aggregation organization internally decides an aggregator for the model aggregation in this round with a round-robin manner.
5) *GlobalModelUploading():* The model aggregator responsible for model aggregation can pack the global model, all local models, and data contributions into a new block by calling this interface after the model aggregation.
6) *GlobalModelDownloading():* Data owners waiting to start the next round of training can download the latest global model by calling this interface.

The contract will check whether the data owner is in the member list. If not, the contract will reject the calling from the data owner.

4) Reputation Contract: The contract is used to update the reputation of data owners. After completing the model testing by federated learning contract, the model aggregator can call the reputation update interface in this contract to complete the reputation update of each data owner.

1) *ReputationUpdating():* The interface takes the model quality testing results as input and combines the historical reputation to calculate the latest reputation value for each data owner. The contract utilizes the new reputation value to calculate the maximum amount of data allowed to contribute in the next round.

5) Reward Allocation Contract: Data owners will be paid for their participation in the current round by invoking the reward allocation interface.

1) *RewardAllocation():* The reward allocation interface will call the reputation query interface and the historical data contribution query interface in the query contract to get the reputation value and data amount, and use this information to calculate the reward for each node. The reward determination algorithm will be presented in Section V.

6) Query Contract: The block of BFL records the historical data contribution and reputation of each data owner and also maintains the member list for each round. The query contract provides the corresponding query interface for this information.

1) *HistoricalDataSizeQuery():* This interface provides a service for querying the amount of historical data contribution.
2) *HistoricalReputationQuery():* This interface provides a service for querying historical reputation.
3) *MemberListQuery():* This interface provides a service for querying the member list.

With the above contracts, the data owners and the model aggregators can interact reliably with each other over BFL network. Now, the BFL process can be described based on its contracts as follows.

1) *Start Task:* When BFL receives a federated learning task, it initiates the task by calling the task initialization contract.
2) *Select Member:* When the task starts, BFL automatically calls memberInitialization() to initialize the members participating in the task. Before the start of each round, the system updates the member list by calling memberUpdating().
3) *Train Local Model:* Data owners call memberListQuery() to confirm their legal identity and get the latest global model by calling globalModelDownloading() and then use their data to train the model locally.
4) *Upload Local Model:* After training is completed, the node calls localModelUploading() to upload the local models and corresponding contributions.
5) *Evaluate Reputation:* The model aggregation organization tests the uploaded local models by calling testResultUploading() and then updates the reputation by calling reputationUpdating().
6) *Aggregate Global Models:* The model aggregation organization calls modelAggregation() to aggregate the models and publishes them to BFL through globalModelUploading().
7) *Allocate Reward:* Data owners get their respective rewards by calling rewardAllocation().

Steps 2)–7) will be performed repeatedly until the federated learning is as effective as expected.

IV. Model Quality-Based Reputation Evaluation

In this section, the reputation evaluation based on model testing for BFL is introduced. First, a local reputation evaluation based on model quality is presented. Then, a global evaluation depended on local reputation evaluations is provided.

A. Local Reputation Evaluation Based on Model Quality

After acquiring all the local models, the model aggregator uses the owned data set to test the local models. The fair-value game [10], a loss-based marginal approach, is applied to test the quality of local models. It measures the impact of the local model on the global model aggregation. Let $A^t_{m,n}$ denote the evaluation result of model aggregator m on data owner n, i.e.,

$$A^t_{m,n} = G\big(w^t\big) - G\big(w^t_{-n}\big) \tag{6}$$

where w^t denotes the global model aggregated by using the local models from all nodes in the set $\mathcal{N}$, while w^t_{-n} denotes the global model aggregated by the set without the local model of node n. $G(\cdot)$ is the function for model accuracy measurement.

In the following, the local reputation evaluation mechanism based on these testing results is constructed. At the beginning, the system initializes the global reputation value $\lambda = 1$ for each data owner. At tth round, the local reputation value $\lambda^t_{m,n}$ of node n is updated according to the testing result $A^t_{m,n}$ and the historical reputation value, which can be formulated by

$$\lambda^t_{m,n} = \begin{cases} \lambda^{\mathcal{H}} + \rho_1 e^{\frac{A^t_{m,n}-\sigma_1}{\theta_1}}, & A^t_{m,n} > \sigma_1 > 0 \\ \lambda^{\mathcal{H}} - \rho_2 e^{\frac{\sigma_2 - A^t_{m,n}}{\theta_2}}, & A^t_{m,n} \le \sigma_2 \le 0 \\ \lambda^{\mathcal{H}}, & \text{others.} \end{cases} \tag{7}$$

Due to that a lot of parameters are introduced by (7), in the following, we present the parameter interpretation and particular function of this reputation evaluation.

1) $\lambda^{\mathcal{H}}$ is the historical reputation value. Considering that the reference value of reputation decreases with time, the time decay function $e^{-\alpha(t-\iota)}$ is used here to weight the historical reputation, i.e.,

$$\lambda^{\mathcal{H}} = \frac{\sum_{\iota=1}^{\iota=t-1} e^{-\alpha(t-\iota)} \lambda^{\iota}_n}{\sum_{\iota=1}^{\iota=t} e^{-\alpha(t-\iota)}}. \tag{8}$$

2) $e^{(\cdot)}$ in (7) is applied to transform activation levels to updating strength. Based on its curve feature, a higher activation level means a larger updating strength and curve slope. The variational curve slope makes the reputation updating more flexible in response to different levels of model quality. ρ_1 and ρ_2 are the reputation update value of unit updating strength.
3) The thresholds σ_1 and σ_2 are introduced for activating reputation update. $A^t_{m,n} - \sigma_1$ and $\sigma_2 - A^t_{m,n}$ reflect the activation levels. θ_1 and θ_2 denote the counting parameters for scaling the activation levels to limit the value to a small interval. The combination of $\{\sigma_1, \sigma_2\}$ and $\{\theta_1, \theta_2\}$ is called tolerance of this reputation mechanism.

Based on the historical reputation, combined with the reputation update value of unit strength, this mechanism updates the reputation by using the update strength converted from the model quality. A node trains a local model and shares it. The model quality testing mechanism gives the corresponding test result. When the model quality reaches the threshold for activation, the corresponding reputation update is activated. For instance, if the node contributes a bad model, the reputation penalty update is activated. The worse the quality of the model, the higher the strength of the reputation update, and this strength changes exponentially as the quality of the model gets worse. This property causes the reputation of the node with very bad model to quickly become 0.

B. Global Reputation Determination

Since each model aggregator has different test data, BFL gives a comprehensive reputation evaluation of the data owners by multiple model aggregators. When model aggregator m performs the tth round task of reputation evaluation, it also adopts the opinions of other model aggregators. Model aggregator m is required to judge the reference values of these recommendations before adopting them. $\eta^t_{m',n}(m)$ indicates the reference value provided by model aggregator m' to model aggregator m in the tth regarding the reputation evaluation of data owner n, i.e.,

$$\eta^t_{m',n}(m) = 1 - \left|\lambda^t_{m',n} - \frac{\sum_{i\in\mathcal{M}} \lambda^t_{i,n}}{M}\right|, \ \ \eta^t_{m',n}(m) > 0 \tag{9}$$

where $|\lambda^t_{m',n} - [(\sum_{i\in\mathcal{M}} \lambda^t_{i,n})/M]|$ represents the distance between the reputation evaluation from model aggregator m' and the average reputation evaluation from the model aggregation organization $\mathcal{M}$. The closer the value of $\eta^t_{m'm(n)}$ is to 1, the higher the reference value of node m' is. When $\eta^t_{m',n}(m) \le 0$, the reputation evaluation provided by m' does not have any reference value.

Model aggregator m collects the local reputation evaluations of all model aggregators for the data owner n, thus generating an indirect reputation evaluation of node m for node n by

$$\lambda^t_{\mathcal{M},n}(m) = \frac{\sum_{i\in\mathcal{M}} \eta^t_{i,n}(m)\lambda^t_{i,n}}{\sum_{i\in\mathcal{M}} \eta^t_{i,n}(m)}. \tag{10}$$

Finally, model aggregator m can obtain the tth global reputation evaluation of node n by

$$\widetilde{\lambda}^t{}_{m,n} = \mu\lambda^t_{m,n} + (1-\mu)\lambda^t_{\mathcal{M},n}(m) \tag{11}$$

where μ is used to balance the two reputation evaluations. If the model aggregator m pays more attention to its own reputation evaluation, then the value of μ can be increased appropriately.

The model aggregation organization takes turns to select one of the model aggregators to carry out this round of reputation updates, e.g., the node m is selected at the tth round to update $\widetilde{\lambda}_{m,n}$. The updated reputation values are recorded in the blockchain through the model aggregation organization. After that, the publicly accepted new reputation evaluation of node n is

$$\lambda^t_n = \widetilde{\lambda}^t{}_{m,n}. \tag{12}$$

For ease of expression, the superscript t will be omitted in the analysis of the next section.

C. Grouping Mechanism for Efficient Reputation Evaluation

Note that the reputation evaluation mechanism will confront high complexity and low efficiency when the number of participants is large. To deal with it, a grouping mechanism is designed. The data owners and model aggregators can be separated into a certain groups according to specified metric, e.g., hops from data owner to model aggregator in the communication topology. Then, the local model quality testing is carried out within the group. The reputation is also first evaluated within the group, then obtain a whole network evaluation by model aggregation organization. When the grouping mechanism is used, model aggregators can perform model quality testing in groups in parallel, which theoretically can greatly improve efficiency. Note that the grouping mechanism relatively narrows the data samples for learning within the group. Hence, the grouping mechanism is optional. Users can choose to turn on the grouping mechanism if the number of nodes is relatively large and sufficient to compensate for the lack of samples caused by grouping.

V. Reputation-Based Reward Allocation and Model Aggregation Algorithm

In this section, a reward allocation algorithm based on the reputation mechanism is first introduced. Then, the optimal data contribution is derived based on the reward allocation. Finally, a reputation weighted model aggregation algorithm is presented to improve the security and robustness of BFL.

A. Reward Allocation

The most intuitive way to allocate rewards is by the weight of the data amount s_n of each node to the total data amount. However, since data contributions are reported by data owners, dishonest owners have sufficient reasons to inflate their contributions. Therefore, a reward allocation algorithm based on reputation-weighted contribution is designed. If malicious node n exaggerates its contribution, then BFL will give it a relatively poor reputation evaluation based on the reputation mechanism introduced in Section IV. The poor reputation evaluation, as the weight for reward allocation, will decrease the reward for n. Based on this, the utility function of data owner n is expressed as

$$u(s_n) = \frac{\lambda_n s_n}{\sum_{i=1}^{N} \lambda_i s_i} R - C_n s_n \tag{13}$$

where R is the budget of the learning task and C_n denotes the unit cost of node n in training the model. The unit cost usually includes computational loss C_n^{com}, communication loss C_n^{cmp}, and storage loss C_n^{stg}, expressed as follows: $C_n = C_n^{\text{com}} + C_n^{\text{cmp}} + C_n^{\text{stg}}$.

In particular, in order to continuously reduce the participation of malicious nodes, such that the amount of data that node n can provide in each round must not be more than the reputation-weighted contribution in the previous round. To facilitate the derivation, the discrete data contribution is approximated as continuous values, expressed as

$$s_n \in \left[0, \lambda_n^{t-1} s_n^{t-1}\right]. \tag{14}$$

Furthermore, a threshold $s_{\min}$ of the minimum data contribution is set. When s_n is less than $s_{\min}$, the node is not allowed to participate in the task.

B. Optimal Data Contribution

Since all data owners jointly participate in the allocating the budget, the competition for reward can be viewed as noncooperative games and each node peruses more reward. Therefore, for node n, its objective function for solving the optimal amount of contribution in the tth round can be expressed as follows:

$$\max_{s_n} \; u(s_n) = \frac{\lambda_n s_n}{\sum_{i=1}^{N} \lambda_i s_i} R - C_n s_n \tag{15a}$$

$$\text{s.t.} \; s_n \le \lambda_n^{t-1} s_n^{t-1} \tag{15b}$$

$$s_n > 0 \tag{15c}$$

$$\frac{\lambda_n s_n}{\sum_{i=1}^{N} \lambda_i N_i} R - C_n s_n \ge 0 \tag{15d}$$

where (15b) and (15c) denote the limits on the data contribution reported, and (15d) means that nodes are individually rational (IR) in order to ensure their nonnegative returns.

Before going further analysis, the Nash equilibrium of the game is first introduced.

Definition 1 (Nash Equilibrium): For the data contributions $\{s_1, s_2, \ldots, s_N\}$ provided by all nodes $\mathcal{N}$, if the benefit obtained by any node $n (n \in \mathcal{N})$ after selecting the data amount s_n is not less than that when it selects any other data amount s'_n, i.e.,

$$u(s_n) \ge u\left(s'_n\right) \; \forall n \in \mathcal{N}. \tag{16}$$

At this time, the system reaches the Nash equilibrium. Under this equilibrium condition, all nodes are stable in a strategic choice.

Based on the definition of the optimization problem above, the following theorems and corollaries can be derived.

Theorem 1: Subject to constraints (15b)–(15d), there exists a unique Nash equilibrium for the noncooperative game consisting of data owners $\mathcal{N}$. The equilibrium point indicates that node n's optimal data contribution s_n^* is independent of others and is related to the unit resource consumption $\{C_i | i \in \mathcal{N}\}$ of each node, i.e.,

$$s_n^* = \frac{(N-1)R}{\lambda_n \sum_{i=1}^{N} C_i} \left(1 - \frac{(N-1)C_n}{\sum_{i=1}^{N} C_i}\right). \tag{17}$$

Proof: To obtain the first-order and second-order derivatives of s_n for the utility function $u(s_n)$, respectively

$$\frac{\partial u(s_n)}{\partial s_n} = \frac{-R\lambda_n s_n}{\left(\sum_{i=1}^{N} \lambda_i s_i\right)^2} + \frac{R}{\sum_{i=1}^{N} \lambda_i s_i} - C_n \tag{18}$$

and

$$\frac{\partial^2 u(s_n)}{\partial {s_n}^2} = -\frac{2R \sum_{i \ne n} \lambda_i s_i}{\left(\sum_{i=1}^{N} \lambda_i s_i\right)^3} < 0. \tag{19}$$

From (18) and (19), the utility function $u(s_n)$ is a convex function with respect to s_n. According to the conditions for the existence of Nash equilibrium, it is proved that there is a Nash equilibrium of the game. Because the utility function is convex and its constraint also satisfies convexity, the KKT condition can be used to solve for the optimal amount of data contribution.

Reform (15a)–(15d) into the standard form of the convex problem

$$\min_{s_n} \; U(s_n) = -u(s_n) = -\frac{\lambda_n s_n}{\sum_{i=1}^{N} \lambda_n s_i} R + C_n s_n \tag{20a}$$

$$\text{s.t. } s_n - \lambda_n^{t-1} s_n^{t-1} \leq 0 \tag{20b}$$

$$-s_n \leq 0 \tag{20c}$$

$$-\frac{\lambda_n s_n}{\sum_{i=1}^{N} \lambda_i N_i} R + C_n s_n \leq 0. \tag{20d}$$

Let α, β, and γ be the Lagrange multipliers. Then, the Lagrange function can be formulated by

$$L(s_n, \alpha, \beta, \gamma) = U(s_n) + \alpha\left(s_n - \lambda_n^{t-1} s_n^{t-1}\right) + \beta(-s_n) + \gamma\left(-\frac{\lambda_n s_n}{\sum_{i=1}^{N} \lambda_i N_i} R + C_n s_n\right). \tag{21}$$

In order to satisfy the KKT condition, the following constraints need to be satisfied:

$$\begin{cases} \frac{\partial L(s_n,\alpha,\beta,\gamma)}{\partial s_n} = 0 \\ \alpha\left(s_n - \lambda_n^{t-1} s_n^{t-1}\right) = 0 \\ \beta(-s_n) = 0 \\ \gamma\left(-\frac{\lambda_n s_n}{\sum_{i=1}^{N} \lambda_n N_i} R + C_n s_n\right) = 0 \\ \alpha \geq 0, \beta \geq 0, \gamma \geq 0. \end{cases} \tag{22}$$

Solve (21) under the constraint of (22), and get the optimal solution that relies on knowing the data contributions of other data owners, i.e.,

$$s_n^* = \frac{\sqrt{\frac{R \sum_{i \neq n} \lambda_i s_i}{C_n}} - \sum_{i \neq n} \lambda_i s_i}{\lambda_n}. \tag{23}$$

Inspired by [33], for all $i \in \mathcal{N}$, the following equation can be got from (23):

$$\sum_{i=1}^{N} \lambda_i s_i = \sqrt{\frac{R \sum_{i \neq n} \lambda_i s_i}{C_n}}. \tag{24}$$

Let $X = \sum_{i=1}^{N} \lambda_i s_i$, then the above equation can be transformed as

$$X^2 = \frac{R(X - \lambda_n s_n)}{C_n}. \tag{25}$$

Further transformations lead to

$$C_n X^2 = R(X - \lambda_n s_n), n \in \mathcal{N}. \tag{26}$$

Sum both sides of (26) for $n = 1, 2, \ldots, N$, that is

$$\begin{aligned} \sum_{i=1}^{N} C_i X^2 &= \sum_{i=1}^{N} R(X - \lambda_i s_i) \\ &= \sum_{i=1}^{N} RX - R \sum_{i=1}^{N} \lambda_i s_i \\ &= \mathrm{RXN} - \mathrm{RX}. \end{aligned} \tag{27}$$

Therefore

$$X^2 \sum_{i \in \mathcal{N}} C_i = \mathrm{RX}(N - 1) \tag{28}$$

and hence, (28) can be converted to

$$X = \frac{R(N-1)}{\sum_{i \in \mathcal{N}} C_i}. \tag{29}$$

Substituting (29) into (25), s_n^* by (17) can be solved. ■

Note that the quality of the data in local model training greatly affects the quality of the overall federated learning model. Hence, it is vital to urge nodes to contribute high-quality data.

Theorem 2: Within a round of federated learning task, for the high-quality data and low-quality data denoted by s_n^h and s_n^l, $u(s_n^h) > u(s_n^l)$ holds.

Proof: In the tth round, the reputation evaluation λ_n^h obtained by node n after contributing high-quality data is greater than the reputation evaluation λ_n^l after contributing low-quality data, i.e., $\lambda_n^h > \lambda_n^l$. From the optimal contribution (17), it follows that:

$$\lambda_n^h s_n^h = \frac{(N-1)R}{\sum_{i=1}^{N} C_i}\left(1 - \frac{(N-1)C_n}{\sum_{i=1}^{N} C_i}\right)$$

$$\lambda_n^l s_n^l = \frac{(N-1)R}{\sum_{i=1}^{N} C_i}\left(1 - \frac{(N-1)C_n}{\sum_{i=1}^{N} C_i}\right).$$

It can be seen that the right-hand side of the equation is a constant. Therefore, derived from $\lambda_n^h s_n^h = \lambda_n^l s_n^l$, $s_n^h < s_n^l$ and $C_n s_n^h < C_n s_n^l$ hold. Substituting them into the utility function u, respectively. $u(s_n^h)$ and $u(s_n^l)$ can be derived as

$$u\left(s_n^h\right) = \frac{\lambda_n^h s_n^h}{\sum_{i=1}^{N} \lambda_i s_i} R - C_n s_n^h$$

$$u\left(s_n^l\right) = \frac{\lambda_n^l s_n^l}{\sum_{i=1}^{N} \lambda_i s_i} R - C_n s_n^l$$

and hence, $u(s_n^h) > u(s_n^l)$. ■

If there are malicious nodes among the data owners, such nodes will maliciously exaggerate their contributions in order to get more rewards. This algorithm gives BFL the ability to counter with malicious nodes.

Corollary 1: Suppose that data owner n is a malicious node, which will contribute data of bad quality. n will be banned from participating in tasks in a given round. The worse its quality is, the faster it is banned.

Proof: If the malicious node chooses to continue destroying the task, its reputation will continue declining based on the reputation mechanism, i.e., $1 > \lambda_n^1 > \lambda_n^2 > \cdots > \lambda_n^{t-1} > \lambda_n$. If the amount of contribution reported in the first round is s_n^1 and its reputation is $\lambda_n^1 < 1$ according to the data contribution constraints by (15b)–(15d), the amount of data it can report in the next round must satisfy $s_n^2 \leq \lambda_n^1 s_n^1$. Hence, there must exist $t \in N^+$, and at the tth round, $s_n \leq \lambda_n^{t-1} s_n^{t-1} \leq \prod_{i=1}^{t-1} \lambda_n^i s_n^1 < s_{\min}$. In addition, the greater the node malice, the greater the penalty given by the reputation mechanism, the fewer rounds it takes to reach $s_n < s_{\min}$. When the amount of reported data is less than the threshold $s_{\min}$, the malicious node will be disqualified and leave the task. ■

C. Reputation Weighted Model Aggregation

To address the problem that the traditional federated average algorithm (FedAvg) is strongly influenced by the quality of local models and the authenticity of the contributed data volume when aggregating local models, reputation evaluation will be introduced here to assist in aggregation. Therefore, the optimized model aggregation method is as follows:

$$w \leftarrow \sum_{i=1}^{N} \frac{\lambda_i s_i}{\sum_{j=1}^{N} \lambda_j s_j} w_i. \tag{30}$$

Corollary 2: Suppose that data owner n is a malicious node, it has a limited and decreasing corrupting effect on the global model.

Proof: Since model aggregation employs reputation to weight the reported contributions, the impact of n's malicious behavior on model aggregation is reduced because of $\lambda_n < 1$ and $s_n < \lambda_n s_n$. From Corollary 1's proof, there must exist $t \in N^+$, and at tth round, $s_n \leq \lambda_n^{t-1} s_n^{t-1} \leq \prod_{i=1}^{t-1} \lambda_n^i s_n^1 < s_{\min}$. That is, as the training continues, $\lambda_n s_n$ has a decreasing share in aggregating the model until it exits the task. ■

D. Adaptability Analysis With Non-IID Data Set

In this part, the adaptability of BFL to the non-IID data set is analyzed, since the owner's data is often non-IID in the realistic scenario. The analysis is presented as follows.

1) The model aggregation organization consisting of all model aggregators has a complete test sample set, which ensures that there is no mistake due to insufficient samples when facing the non-IID data set.
2) A final reputation evaluation result is obtained by combining multiple model aggregators based on the reference value, i.e., (9) and (10) in this article. This greatly reduces the evaluation errors caused by the non-IID data set for each different local model.

Accordingly, the proposed BFL remains well adaptability to cases with non-IID data sets. More experimental verifications are presented in the following section.

VI. Performance Evaluation

A. Simulation Settings

1) Federated Learning Settings: In our experiments, ten data owners and five model aggregation nodes are set. Two public data sets, MNIST and CIFAR10, are selected for federated learning testing. According to the proposed BFL, each data owner reads a number of data from the prepared training set in each round of the task. To illustrate the adaptivity of BFL in the case of the non-IID data set, the data assigned to nodes are artificially set to be non-IID. In addition to this, the data from malicious nodes will tamper with the data labels according to their maliciousness, thus simulating data of different quality. 0%, 25%, 50%, 75%, and 100% error rates correspond to opposite data quality are considered, respectively. For example, if the experiment requires simulating data with an accuracy rate of 75%, then the 25% data label is artificially set to be wrong. Similarly, the test sets are equally distributed to the model aggregation nodes, and the data sets obtained by each node have different distributions. Two tolerances were prepared for the experiment: $\sigma_1 = 0.01$, $\sigma_2 = -0.01$, $\theta_1 = 0.01$, $\theta_2 = 0.004$ and $\sigma_1 = 0.02$, $\sigma_2 = -0.002$, $\theta_1 = 0.01$, $\theta_2 = 0.004$. Although the data owners have non-IID data sets, the system still ensures that all data covered the complete sample space.

Algorithm 1 Federated Learning Contract: Data Owner

```
Input: Data owner $n$, query contract $\mathcal{C}_q$, model aggregators $\mathcal{M}$
Initialization: Global model $w = 0$, new local model $w_n$
for $t = 1 \to T$ do
    if $n \notin \mathcal{C}_q$.memberListQuery() then
        break
    end if
    globalModelDownloading($n$) $\to$ $w$
    contribute $s_n$
    $w \xrightarrow{s_n} w_n$
    localModelUploading($\mathcal{M}$, $w_n$)
end for
```

Local training of nodes is performed for MNIST and CIFAR10 using two deep learning methods, respectively. The experiment utilizes the simplest three-layer fully connected network to train the MNIST model. The deep learning structure for training CIFAR10 consists of a convolutional layer and a fully connected layer. The convolutional layer consists of two convolutional operations and two pooling operations. The fully connected layer is a two-layer structure. Set their learning rate to 0.01. The times of the entire federated learning training are 10 and the epoch number in local training is 20.

2) Blockchain Settings: There are many existing blockchain platforms that support smart contracts, e.g., Ethereum [40] and Hyperledger Fabric [41]. Their own smart contract mechanism already does a good job of implementing the contract design of our proposed solution. The experiment implements the BFL based on Hyperledger Fabric.

Data owners and model aggregators exist as peer nodes in the Fabric network. Meanwhile, model aggregators are set up in the network as orderer nodes in Fabric. They perform the task of packing the blocks. Orderer creates a channel and all nodes join the channel to form a consortium. Set the maximum size of a transaction to 512 kB, the maximum size of a block to 10M, and the blockout period to 10 s. Configure the Fabric with MaxMessageCount = 20 so that when the number of transactions reaches 20, the block can still be packed even if the blockout period is not met. In combination with smart contracts, Algorithms 1 and 2 show the pseudocode implementations of the most important function, federated learning in BFL.

B. Results and Analysis

1) Performance of BFL: The experiments first evaluate the accuracy performance of Google's proposed FedAvg and our proposed BFL-based Federated Learning Algorithm when different numbers of malicious nodes are present in the network. The experiments are set to have 0%, 10%, 30%, and 50% of the number of malicious nodes, respectively. To make the effect most visible, each malicious node provides data of 0% quality. Fig. 2 shows the performance of the two algorithms executing MNIST and CIFAR10 training tasks in this environment. It can be seen that the task is indeed better trained under the independent identical distribution than under the non-IID. However, our proposed scheme obtains better performance

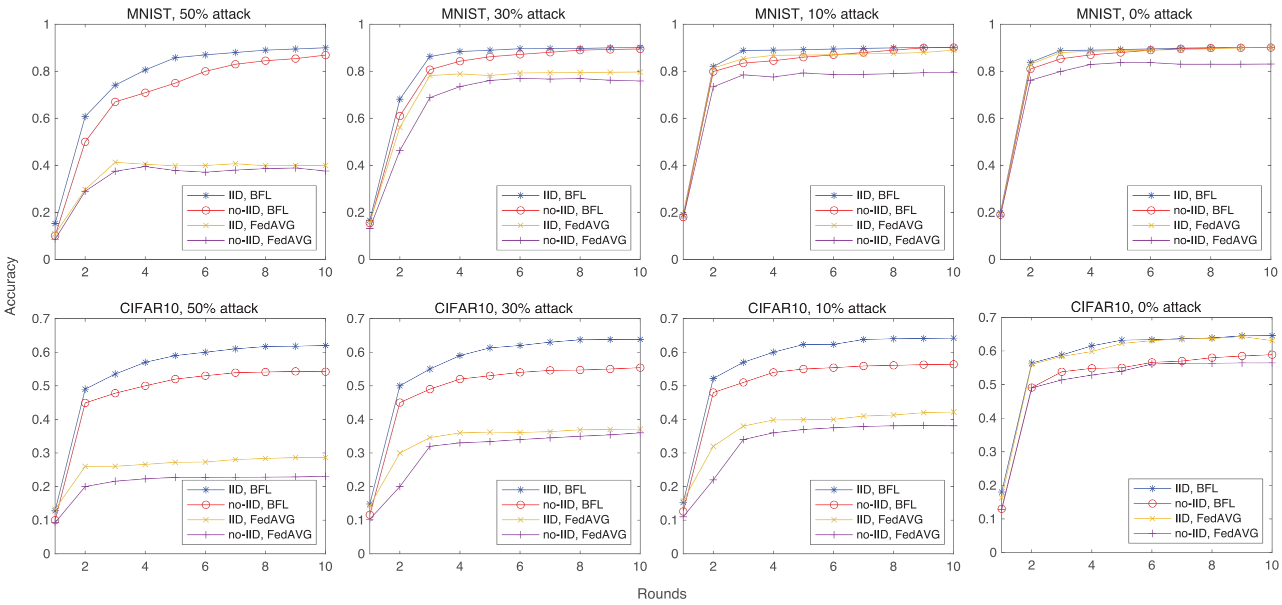


Fig. 2. Performance of our proposed BFL algorithm and FedAVG algorithm is compared in two data sets with different degrees of malicious node occupancy. (Both IID and non-IID cases are considered, respectively.)

Algorithm 2 Federated Learning Contract: Model Aggregator

```
Input: model aggregator $m$, data owners $\mathcal{N}$, model aggregators $\mathcal{M}$, query contract $\mathcal{C}_q$, reputation constract $\mathcal{C}_r$, reward constract $\mathcal{C}_{re}$
Initialization: Global model $w = 0$, local models $ws = \{\}$, contributions $ss = \{\}$, testing result $rs = \{\}$, reputations $reps = \{\}$
Output: $w$
for $t = 1 \rightarrow T$ do
    for $n \in \mathcal{N}$ do
        if $n \notin \mathcal{C}_q$.memberListQuery() then
            $\mathcal{N} - n \rightarrow \mathcal{N}$
        end if
    end for
    localModelDownloading($\mathcal{N}$) $\rightarrow ws$
    testResultUploading($ws$) $\rightarrow rs$
    $\mathcal{C}_r$.reputationUpdating($\mathcal{N}$, $rs$) $\rightarrow reps$
    modelAggregation($ws$, $reps$, $ss$) $\rightarrow w$
    globalModelUploading($w$)
    $\mathcal{C}_{re}$.rewardAllocation($\mathcal{N}$, $reps$, $ss$)
end for
return $w$
```

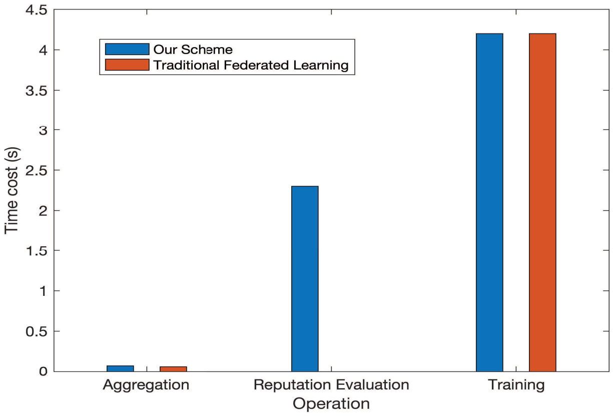


Fig. 3. Comparison of the time cost between our proposed scheme and the FedAVG algorithm.

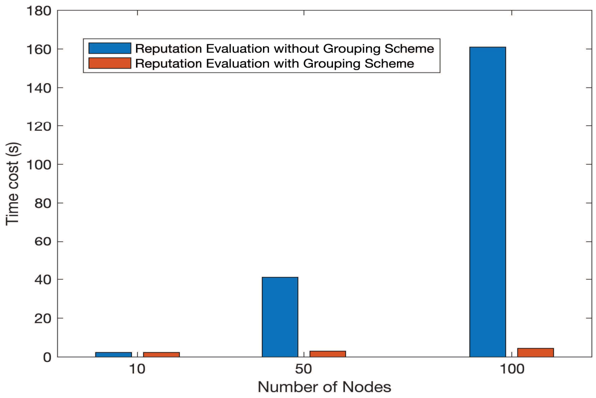


Fig. 4. Comparison of system time cost with and without the grouping mechanism turned on in the reputation evaluation mechanism in BFL.

than FedAVG in different environments. Comparing the impact of different levels of malicious node occupancy on the task, it is clear from the figure that our proposed algorithm is relatively resistant to malicious nodes. In particular, when the malicious node percentage reaches 50%, our proposed scheme still maintains a high performance, while FedAVG is severely damaged.

In addition to accuracy, the experiments also evaluate the time cost of the three most critical components of the proposed algorithm: 1) model aggregation; 2) model training; and 3) reputation evaluation mechanism. As can be seen from Fig. 3, our proposed algorithm has more time cost on reputation evaluation compared to the traditional FedAVG algorithm. It can be seen that when the number of nodes is 10, the time cost of reputation evaluation is close to half of the training time. To optimize the performance of reputation evaluation, optional grouping schemes are provided in our algorithm. Fig. 4 represents the comparison of time cost when grouping scheme is selected or not in different number of nodes. It can be

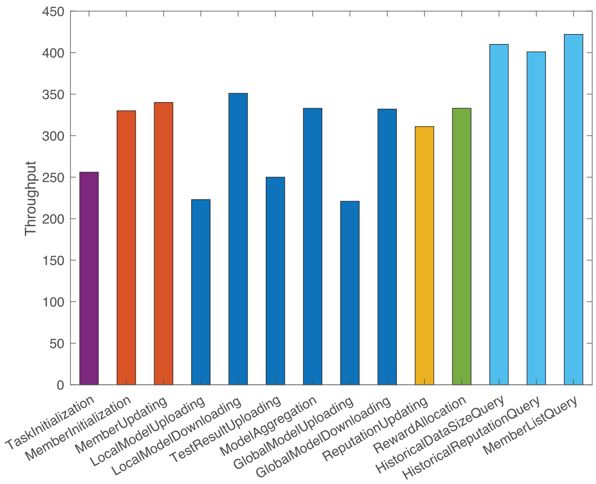


Fig. 5. Average throughput of each interface in the smart contracts when the number of client requests is 1000.

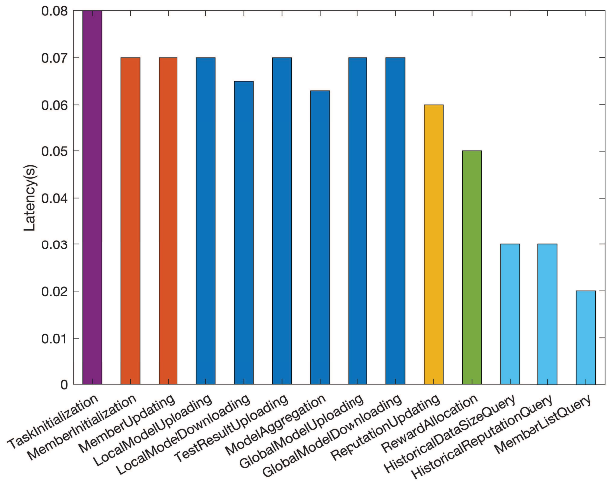


Fig. 6. Average response latency of each interface in the smart contracts when the number of data owners is 100.

clearly seen that as the number of nodes increases, the time cost of reputation evaluation without taking a grouping scheme becomes large. When the number of nodes is 100, it even takes close to 160 s, which is already much larger than the time cost of the training task itself. If the grouping scheme is turned on, the time cost of reputation evaluation is still kept small. Therefore, when the number of nodes is too large, the grouping scheme can be turned on, which improves the efficiency and also compensates the impact of grouping on the reputation evaluation accuracy due to the excessive number of nodes.

To further illustrate that our proposed scheme works properly on Hyperledger Fabric, experiments are conducted to test the performance of smart contracts using Hyperledger Caliper. Fig. 5 represents the throughput of the interfaces provided by each of the major smart contracts when the number of requests reaches 1000. It can be seen that the throughput of a single interface of the system is in the range of 200 times per second to 400 times per second. Fig. 6 shows the latency of each interface when the number of nodes is 100. The experiments demonstrate that the federated learning operations have relatively low latency in smart contracts. Thus, the lower latency and reasonable throughput indicate that the existing Fabric platform is well adapted to the proposed approach.

2) Performance of the Reputation Evaluation Mechanism: To show the performance of the proposed reputation evaluation mechanism, the experiments first test its node reputation variation under different levels of tolerance. The next experiment sets the percentage of malicious nodes to 10%. The experiment makes one node out of ten nodes as a malicious node and observes its performance. Fig. 7(a) represents the average reputation change for malicious (quality = 0%) and honest nodes with high and low tolerance, respectively. The reputation changes of malicious and honest nodes show very different trends at the same tolerance level. The reputation of the honest node grows slowly and is always greater than 1, while the reputation of the malicious node decreases rapidly and finally reaches 0. From another perspective, with different tolerance levels, the higher the tolerance level, the higher the reputation evaluation of honest nodes, while the reputation of malicious nodes decreases more slowly.

To illustrate the impact of the data quality on the reputation evaluation for a single node. Since there will be cases where the system holds nodes with poor data quality without malicious intent, it is necessary to make clear that our proposed scheme will not disadvantage such nodes. Fig. 7(b) shows the variation of nodes' reputation at different data quality. The higher the data quality is offered, the higher overall reputation is obtained. When nodes contribute 100% data quality, their reputation is always greater than 1. When nodes' data quality is too low, such as 0% and 25%, their reputation drops quickly and eventually becomes 0. The focus is on 50% and 75% data quality, where their reputation drops relatively slowly and eventually stays at a more suitable reputation evaluation. This is probably due to the fact that most of the data still have a positive effect on the task.

In the proposed scheme, the amount of data contribution that a node can report per round is limited by its reputation value. Fig. 7(c) represents the optimal amount of data for a single node when all the others contribute data with 100% data quality. It can be seen that the worse the data quality is, the faster its optimal data amount decreases. It is worth noting that when the data quality is 100%, the optimal data amount of a node decreases because its reputation value is greater than 1. This distinguishes it from the other cases. The data amount allowed to be reported will be decreased due to the decline of reputation.

Meanwhile, for different amounts of data contribution, the benefits of the node show corresponding differences. From an overall perspective, the higher the data quality is offered, the higher the benefit can be obtained, as shown in Fig. 7(d). When the data quality is 100%, the node's benefit continues to rise and finally tends to be stable. Other than that, the rest of the cases carrying wrong data, the node utilities show a trend of increasing and then decreasing. This is because although the data contribution of the malicious nodes is decreasing, at first the contribution of the honest nodes is also decreasing due to their rising reputation. This instead leads to an increase

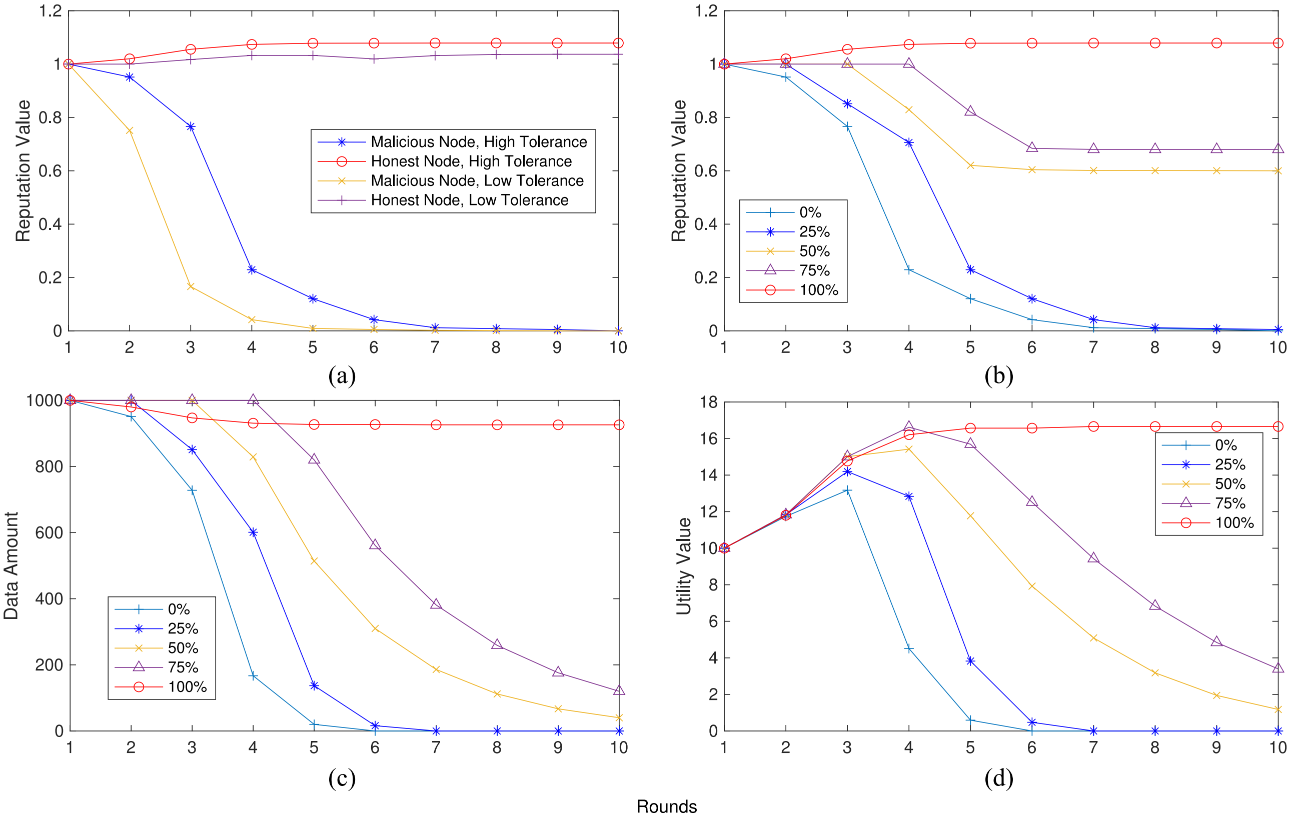


Fig. 7. Variation of node reputation, data contribution, and utilities under different scenarios. (a) Variation of average reputation of malicious and honest nodes under different tolerance levels. (b) Variation of reputation of a single node when contributing data of different quality. (c) Variation of optimal data contribution of a single node when contributing data of different quality. (d) Comparison of utilities of a single node when contributing data of different quality.

in their benefits in the short term. However, when the rise in reputation of honest nodes slows down, the decrease in their data contribution also starts to slow down. Therefore, the benefits of malicious nodes start to decrease, and the worse the data quality is the faster it decreases. The utility of a node with very poor quality will become zero.

C. Security and Reliability Analysis

Compared to traditional centralized federated learning systems, the proposed BFL improves the reliability and security of federated learning from the following aspects.

1) Smart Contract: Blockchain technology provides a decentralized environment for the deployment and execution of federated learning smart contracts. In BFL, smart contracts are backed up by the entire network. Smart contracts do not need to worry about malicious tampering and also do not need to worry about the contract not being executed after the contract conditions are met.

2) Model: In BFL, both the local and global models generated during the training process are stored in the blockchain. The tamper-proof feature of blockchain keeps these models from being maliciously broken and also enables the models to be audited at any time.

3) Reputation: The security and reliability are the cornerstone of the proper execution of the entire BFL. The reputation evaluations generated in BFL's reputation mechanism are backed up by nodes across the blockchain network, which prevents nodes from conspiring with each other to tamper with the reputation of others.

VII. Conclusion

In this article, the issue of how to improve the model quality of federated learning is concerned. The BFL with a reputation mechanism is proposed for high-quality model aggregation. A consortium blockchain with systematic smart contracts is designed to conduct the decentralized federated learning tasks in a trustworthy and reliable way. A reputation mechanism together with reputation constrained reward allocation is developed, which not only motivates the data owners to participate the learning tasks but also to contribute high-quality data. The behavior strategies of data owners are formulated by the noncooperative game. The derived unique equilibrium proves that the data owners can acquire highest reward with contribution of highest quality data. Simulations based on the public data sets show that the model accuracy by BFL can approach the ideal state and has advantage over some existing methods. In future work, more security issues on the BFL will be considered, such as identity privacy protection.

References

[1] C. Zhang, H. Zhang, J. Qiao, D. Yuan, and M. Zhang, "Deep transfer learning for intelligent cellular traffic prediction based on cross-domain big data," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1389–1401, Jun. 2019.

[2] S. Biswas, S. G. Anavatti, and M. A. Garratt, "Multiobjective mission route planning problem: A neural network-based forecasting model for mission planning," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 1, pp. 430–442, Jan. 2021.

[3] T. Yang *et al.*, "Intelligent imaging technology in diagnosis of colorectal cancer using deep learning," *IEEE Access*, vol. 7, pp. 178839–178847, 2019.

[4] Z. Ba, T. Zheng, X. Zhang, Z. Qin, B. Li, X. Liu, and K. Ren, "Learning-based practical smartphone eavesdropping with built-in accelerometer," in *Proc. Netw. Distrib. Syst. Symp.*, San Diego, CA, USA, 2020, pp. 23–26.

[5] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, Fort Lauderdale, FL, USA, 2017, pp. 1273–1282.

[6] H. B. McMahan, E. Moore, D. Ramage, and B. A. Y. Arcas. "Federated Learning of Deep Networks using Model Averaging." 2017. [Online]. Available: https://uk.arxiv.org/abs/1602.05629v1

[7] H. Yu *et al.*, "A fairness-aware incentive scheme for federated learning," in *Proc. AAAI/ACM Conf. AI Ethics Soc.*, New York, NY, USA, 2020, pp. 393–399.

[8] S. Yang, F. Wu, S. Tang, X. Gao, B. Yang, and G. Chen, "On designing data quality-aware truth estimation and surplus sharing method for mobile crowdsensing," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 4, pp. 832–847, Apr. 2017.

[9] J. Augustine, N. Chen, E. Elkind, A. Fanelli, N. Gravin, and D. Shiryaev, "Dynamics of profit-sharing games," *Internet Math.*, vol. 11, no. 1, pp. 1–22, 2015.

[10] S. Gollapudi, K. Kollias, D. Panigrahi, and V. Pliatsika, "Profit sharing and efficiency in utility games," in *Proc. Annu. Eur. Symp. Algorithms*, Dagstuhl, Germany, 2017, pp. 1–14.

[11] Y. Zhan, J. Zhang, Z. Hong, L. Wu, P. Li, and S. Guo, "A survey of incentive mechanism design for federated learning," *IEEE Trans. Emerg. Topics Comput.*, early access, Mar. 3, 2021, doi: 10.1109/TETC.2021.3063517.

[12] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10700–10714, Dec. 2019.

[13] Z. Song, H. Sun, H. H. Yang, X. Wang, Y. Zhang, and T. Q. S. Quek, "Reputation-based federated learning for secure wireless networks," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1212–1226, Jan. 2022, doi: 10.1109/JIOT.2021.3079104.

[14] S. Nakamoto. "Bitcoin: A Peer-to-peer Electronic Cash System." 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[15] Y. Yuan and F.-Y. Wang, "Blockchain and cryptocurrencies: Model, techniques, and applications," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 9, pp. 1421–1428, Sep. 2018.

[16] K. Upadhyay, R. Dantu, Z. Zaccagni, and S. Badruddoja, "Is your legal contract ambiguous? Convert to a smart legal contract," in *Proc. IEEE Int. Conf. Blockchain*, Rhodes, Greece, 2020, pp. 273–280.

[17] M. A. Ferrag, M. Derdour, M. Mukherjee, A. Derhab, L. Maglaras, and H. Janicke, "Blockchain technologies for the Internet of Things: Research issues and challenges," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2188–2204, Apr. 2019.

[18] M. A. Ferrag and L. Shu, "The performance evaluation of blockchain-based security and privacy systems for the Internet of Things: A tutorial," *IEEE Internet Things J.*, vol. 8, no. 24, pp. 17236–17260, Dec. 2021.

[19] S. Xia, F. Lin, Z. Chen, C. Tang, Y. Ma, and X. Yu, "A Bayesian game based vehicle-to-vehicle electricity trading scheme for blockchain-enabled Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 6856–6868, Jul. 2020.

[20] S. Wang, L. Ouyang, Y. Yuan, X. Ni, X. Han, and F.-Y. Wang, "Blockchain-enabled smart contracts: Architecture, applications, and future trends," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 11, pp. 2266–2277, Nov. 2019.

[21] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.

[22] H. Elayan, M. Aloqaily, and M. Guizani, "Sustainability of healthcare data analysis IoT-based systems using deep federated learning," *IEEE Internet Things J.*, early access, Aug. 9, 2021, doi: 10.1109/JIOT.2021.3103635.

[23] W. Y. B. Lim *et al.*, "Dynamic contract design for federated learning in smart healthcare applications," *IEEE Internet Things J.*, vol. 8, no. 23, pp. 16853–16862, Dec. 2021, doi: 10.1109/JIOT.2020.3033806.

[24] Y. Li, X. Tao, X. Zhang, J. Liu, and J. Xu, "Privacy-preserved federated learning for autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, early access, Jun. 1, 2021, doi: 10.1109/TITS.2021.3081560.

[25] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Blockchain empowered asynchronous federated learning for secure data sharing in Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4298–4311, Apr. 2020.

[26] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4177–4186, Jun. 2020.

[27] A. Fu, X. Zhang, N. Xiong, Y. Gao, H. Wang, and J. Zhang, "VFL: A verifiable federated learning with privacy-preserving for big data in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 18, no. 5, pp. 3316–3326, May 2022, doi: 10.1109/TII.2020.3036166.

[28] Z. Su *et al.*, "Secure and efficient federated learning for smart grid with edge-cloud collaboration," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1333–1344, Feb. 2022, doi: 10.1109/TII.2021.3095506.

[29] M. Song *et al.*, "Analyzing user-level privacy attack against federated learning," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 10, pp. 2430–2444, Oct. 2020.

[30] S. Fan, H. Zhang, Y. Zeng, and W. Cai, "Hybrid blockchain-based resource trading system for federated learning in edge computing," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2252–2264, Feb. 2021.

[31] R. Zeng, S. Zhang, J. Wang, and X. Chu, "FMore: An incentive scheme of multi-dimensional auction for federated learning in MEC," in *Proc. IEEE 40th Int. Conf. Distrib. Comput. Syst.*, Singapore, 2020, pp. 278–288.

[32] L. Dong and Y. Zhang, "Federated learning service market: A game theoretic analysis," in *Proc. Int. Conf. Wireless Commun. Signal Process.*, Nanjing, China, 2020, pp. 227–232.

[33] Y. Zhan, P. Li, Z. Qu, D. Zeng, and S. Guo, "A learning-based incentive mechanism for federated learning," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6360–6368, Jul. 2020.

[34] M. Wu, D. Ye, J. Ding, Y. Guo, R. Yu, and M. Pan, "Incentivizing differentially private federated learning: A multidimensional contract approach," *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10639–10651, Jul. 2021.

[35] N. Ding, Z. Fang, and J. Huang, "Optimal contract design for efficient federated learning with multi-dimensional private information," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 186–200, Jan. 2021.

[36] P. Sun *et al.*, "Pain-FL: Personalized privacy-preserving incentive for federated learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3805–3820, Dec. 2021.

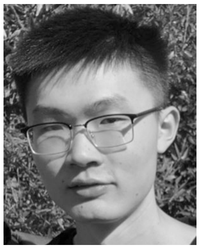
[37] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Commun. Lett.*, vol. 24, no. 6, pp. 1279–1283, Jun. 2020.

[38] J. Kang *et al.*, "Optimizing task assignment for reliable blockchain-empowered federated edge learning," *IEEE Trans. Veh. Technol.*, vol. 70, no. 2, pp. 1910–1923, Feb. 2021.

[39] Y. Qu, S. R. Pokhrel, S. Garg, L. Gao, and Y. Xiang, "A blockchained federated learning framework for cognitive computing in industry 4.0 networks," *IEEE Trans. Ind. Informat.*, vol. 17, no. 4, pp. 2964–2973, Apr. 2021.

[40] D. D. Wood. "Ethereum: A Secure Decentralised Generalised Transaction Ledger." 2014. [Online]. Available: https://ethereum.github.io/yellowpaper/paper.pdf

[41] E. Androulaki *et al.*, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proc. 13th EuroSys Conf.*, New York, NY, USA, 2018, pp. 1–15.



Jiahao Qi received the B.Eng. degree from the College of Mathematics and Computer Science, Zhejiang Normal University, Jinhua, China, in 2019, where he is currently pursuing the M.Eng. degree.

His research interests include federated learning and blockchain technology.

Feilong Lin (Member, IEEE) received the B.Eng. and M.Eng. degrees in electronic information engineering from Xidian University, Xi'an, China, in 2004 and 2007, respectively, and the Ph.D. degree in control science and engineering from Shanghai Jiao Tong University, Shanghai, China, in 2016.

He joined the School of Mathematics and Computer Science, Zhejiang Normal University, Jinhua, China, in 2016, where he is currently an Associate Professor and the Associate Director of the Department of Computer Science and Engineering. He is also the Deputy Director of the Blockchain Lab, Zhejiang Normal University. His research interests include blockchain, edge computing, and federated learning, and their applications in industrial networks, medical big data, and new energy networks.

Zhongyu Chen received the Ph.D. degree from the College of Computer, Shanghai University, Shanghai, China, in 2011.

He is currently a Full Professor with the Department of Computer Science and Engineering, Zhejiang Normal University, Jinhua, China. He is the Chief Director of the Blockchain Lab and the Center of Software R&D, Zhejiang Normal University and the Vice Chairman of Zhejiang Blockchain Technology Application Association. His research interests include software engineering, big data, and blockchain technology and applications.

Changbing Tang (Member, IEEE) received the B.S. and M.S. degrees in mathematics and applied mathematics from Zhejiang Normal University, Jinhua, China, in 2004 and 2007, respectively, and the Ph.D. degree from the Department of Electronic Engineering, Fudan University, Shanghai, China, in 2014.

He is currently an Associate Professor with the College of Physics and Electronics Information Engineering, Zhejiang Normal University. His current research interests include game theory, blockchain and its applications, networks, and distributed optimization.

Dr. Tang was a recipient of the Academic New Artist Doctoral Post Graduate from the Ministry of Education of China in 2012.

Riheng Jia received the B.E. degree in electronics and information engineering from Huazhong University of Science and Technology, Wuhan, China, in 2012, and the Ph.D. degree in computer science and technology from Shanghai Jiao Tong University, Shanghai, China, in 2018.

He is currently an Associate Professor with the Department of Computer Science and Engineering, Zhejiang Normal University, Jinhua, China. His current research interests include wireless networks, energy harvesting networks, and smart IoT.

Minglu Li (Senior Member, IEEE) received the Ph.D. degree in computer software from Shanghai Jiao Tong University, Shanghai, China, in 1996.

He is a Full Professor and the Director of Artificial Intelligence Internet of Things Center, Zhejiang Normal University, Jinhua, China. He is also holding the Director of the Network Computing Center, Shanghai Jiao Tong University. He has published more than 400 papers in academic journals and international conferences. His research interests include vehicular networks, big data, cloud computing, and wireless sensor networks.

Prof. Li was the Chairman of Technical Committee on Services Computing from 2004 to 2016 and Technical Committee on Distributed Processing from 2005 to 2017 of IEEE Computer Society in Great China region. He served as a General Co-Chair for IEEE SCC, IEEE CCGrid, IEEE ICPADS, and IEEE IPDPS and a Vice Chair for IEEE INFOCOM. He also served as a PC Member of more than 50 international conferences, including IEEE INFOCOM 2009–2016 and IEEE CCGrid 2008.