International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18 (2026) pp. 1-16

https://doi.org/10.70917/ijcisim-2026-1872
Article

Research into Developing a Framework for Cross-Organizational Financial Information Confidentiality Safeguarding and Cooperative Examination via a Distributed Learning Structure

Fengjuan Yu [1,*], Lei Chen [1], Shuai Zeng [1], Zhuo Yang [1] and Li Yue [1]

[1] School of Accounting, Tianfu College of SWUFE, Mianyang, Sichuan, 618500, China
* Correspondence author: yufengjuan@tfswufe.edu.cn

Abstract: Owing to the inherent contradiction between "data silos" and the threat of financial information leakage within cross-firm cooperative audits, this research introduces a framework for fiscal data confidentiality preservation and joint auditing via federated learning. This architecture incorporates a blockchain consensus protocol that facilitates dual-factor participation, utilizes differential privacy to shield the integrity of intermediate weights during federated training, and develops a dynamic model synthesis strategy rooted in quality metrics and trust ratings; the operation of this procedure is autonomously initiated by smart contracts. A collective-based synthesis auditing mechanism is employed to swiftly pinpoint corporate participants who have submitted tainted data, while the collective financial analytics framework and equity among data contributors remain accurate. Based on the findings, the precision of the universal model under defense remains nearly identical to that of the non-poisoned group-summarized model, reaching 91.39%. The effectiveness of the universal model in poisoning scenarios is 90.82%. Free-rider maneuvers are more conspicuous and simpler to identify in collective summation, thereby preventing the exposure of transient parameters, and consequently, a reliable federated learning structure with robust privacy defense is realized.

Keywords: Federated learning; Consensus algorithm; Blockchain; Corporate financial data privacy protection; Collaborative auditing

1. Introduction

Within the digital age, information has evolved into a major production asset, and partnership between financial firms regarding data is utilized to derive greater insights from this data, including cross-organizational risk management [1]. Nevertheless, because of progressively stringent data security regulations and the challenge of "information silos", inter-firm collaborative auditing has been significantly hindered. Consequently, federated learning has emerged. Multiple entities can partner to develop models without exchanging the raw data. By transmitting encoded parameters, and hence guaranteeing that "the data stays local while the algorithm evolves", both the extraction of data utility and confidentiality preservation are attained [2-3].

Federated learning represents a decentralized machine learning framework which develops a unified predictive model across numerous devices or clusters by merely sharing model parameters instead of original datasets [4-5]. This approach differs from traditional centralized learning; specifically, data remains stored locally on individual devices of the respective owners rather than being aggregated onto a single server [6-7]. Utilizing cryptographic techniques, distributed algorithms, and security mechanisms, computations are performed locally by all stakeholders, and subsequently, the encoded

outcomes are transmitted to a central hub or coordination point for collective model synthesis [8-10]. Without accessing raw records, the central entity refines the universal model based on collective feedback from participants and returns the optimized version to each party. Subsequently, these entities utilize the enhanced global architecture and local information to iterate training cycles until the desired convergence is achieved [11-14]. Considering these factors, investigating the construction of a cross-industry financial information confidentiality preservation and cooperative verification architecture within a federated learning landscape is essential for the enduring prosperity of organizations.

This research establishes a cross-organization financial information confidentiality safeguarding and cooperative verification scheme within a federated learning architecture by combining blockchain systems and differential privacy techniques. A blockchain framework is utilized to log the interim parameters of the network throughout the learning phase, and via this mechanism, it incentivizes participating nodes to execute collective integrated audit and authentication tasks. Dynamic model synthesis is carried out by modifying the weight factors of the consolidated network based on local model performance evaluation outcomes and past credibility ratings of the entities. We introduce a blockchain consensus protocol rooted in dual-factor contributions for the distributed learning context to encourage federated learning participants to engage in training and enhance the robustness of the framework. To safeguard the confidentiality of these interim variables, Laplace noise is incorporated at varying intensities depending on the model's quality, thereby realizing privacy-secured manipulation of these intermediate values. Ultimately, the viability of this framework will be demonstrated through a prototype implementation and empirical testing.

2. Prerequisites

2.1. Blockchain

To be precise, a blockchain represents a distributed consensus database. By integrating these aforementioned components in a novel manner, the veracity of interactions between unconnected entities is guaranteed; consequently, the necessity for a third-party credit endorsement in conventional trading frameworks is diminished. According to the specific requirements of implementation, three categories of blockchain architecture are generally categorized as public, consortium, and private. Public blockchains allow any node to participate without restriction; consortium blockchains necessitate that individual nodes undergo identity validation by the collective; private blockchains demand similar authentication by a specific organization. Public blockchains are decentralized yet possess limited throughput and are challenging to monitor; private blockchains provide superior throughput but remain centralized; therefore, consortium blockchains have emerged as a perfect deployment framework because they are multi-centralized, offer high throughput, and are more manageable. Each block within a blockchain comprises a header storing metadata and a body holding transactional information. The header establishes a chain by incorporating the hash of the preceding block, while the Merkle root hash guarantees the presence and consistency of the transactional data within the body. The body stores transactional information in various formats depending on the specific use cases, and every transaction is fundamentally a signed data unit that shifts from one account address to another. A consensus protocol must be utilized by all decentralized nodes in a blockchain to reach agreement on the blocks of aggregated transactions and adheres to unique regulations. Depending on whether the blockchain is public or a consortium, the two primary consensus algorithms are public blockchain consensus protocols and consortium blockchain consensus protocols.

2.2. Federated Learning

Within federated learning, a parameter server assigns training jobs to individual nodes, which jointly construct a collective model through parameter exchange. Typically, a group of entities participating in federated learning is considered. Under this framework, participant p_i maintains a private local dataset $D_i = \{(x_1, y_1), (x_2, y_2), \cdots, (x_m, y_m)\}$, where $x_i \in X^u$ is a u-dimensional input vector and y_i is the corresponding label value. The complete training data pool is $D = \bigcup_{i=1}^{N} D_i$. The federated learning procedure is formally characterized as follows:

1) System startup. Initially, the federated learning objective is specified, and the training parameters, including the epoch count and gradient step, are set; subsequently, the initialized global architecture w_G^0 is disseminated to the distributed learning contributor set P.

2) Local model optimization. Within the k -th execution of this procedure, participant p_i develops the regional model w_i^k by using its individual dataset D_i and the preceding regional model w_i^{k-1}. The goal is to reduce the local model loss metric, namely:

$$w_i^k \leftarrow \min_{w_i^{k-1}} \left\{ \frac{1}{m} \sum_{j=1}^{m} L\left(f\left(w_i^{k-1}, x_j \right), y_j \right) \right\} \tag{1}$$

In Eq. (1), $L(w_i^{k-1})$ denotes the loss term associated with the previous local model, $f(w_i^{k-1}, x_j)$ is the model output for input feature x_i , y_i is the actual label, and k is the local iteration index. Equivalently, the local loss can be written as $L\left(w_i^{k-1}\right) = \frac{1}{m} \sum_{j=1}^{m} L\left(f\left(w_i^{k-1}, x_j \right), y_j \right)$.

Ordinarily, regional parameter adjustments are executed via stochastic gradient descent (SGD). To enhance learning performance within federated systems, model derivatives are generally calculated using a mini-batch $B_i \in D_i$, specifically:

$$w_i^k = w_i^{k-1} - \eta_i \nabla g\left(w_i^{k-1}; B_i \right) \tag{2}$$

In Eq. (2), η_i represents the learning rate used by participant p_i . After t epochs, participant p_i transmits the sequentially refined local model w_i^t to the parameter server; the corresponding mini-batch gradient is denoted as $\nabla g\left(w_i^{k-1}; B_i \right) = \frac{1}{|B_i|} \sum_{j=1}^{|B_i|} \frac{\partial L\left(f\left(w_i^{k-1}, x_j \right), y_j \right)}{\partial w_i^{k-1}}$.

3) Universal model synthesis. Upon the parameter server gathering the individual models submitted by the participant set P , it executes protected aggregation to create the fresh global model w_G^t , which is computed by $w_G^t = \frac{1}{\sum_{i=1}^{N} |B_i|} \sum_{i=1}^{N} |B_i| w_i^t$ and then distributed to the participant group for an additional iteration. Existing scholars have introduced diverse secure aggregation techniques, the most prevalent being Federated Averaging (FedAvg). The parameter server evaluates the global model loss metric following t iterations:

$$L\left(w_G^t\right) = \frac{1}{N} \sum_{i=1}^{N} L\left(w_i^t\right) = \frac{1}{Nm} \sum_{i=1}^{N} \sum_{j=1}^{m} L\left(f\left(w_i^t, x_j \right), y_j \right) \tag{3}$$

Execute procedures (2) and (3) until the loss metric for the federated learning collective model reaches convergence or a desirable degree of precision is attained.

2.3. Differential Privacy

The primary challenge that differential privacy tackles involves database privacy leakage. The fundamental concept involves introducing perturbation noise into the database entries so that the inclusion or exclusion of an individual record fails to alter the computational outcome, thereby making it challenging for adversaries to glean any information regarding the initial data entries through the observed outputs. Differential privacy differs from anonymization techniques by not requiring knowledge of the attacker's background, and it relies on a solid mathematical framework to quantitatively evaluate privacy security levels. While differential privacy is relatively straightforward to implement, it necessitates trading off certain data precision for privacy preservation. We shall subsequently present a formal differential privacy definition alongside various approaches for its realization.

Definition 1 (Differential Privacy). Consider any pair of adjacent datasets D_1, D_2 that vary by precisely one data entry. If a stochastic algorithm M fulfills the condition below for every subset O of all possible outputs:

$$\Pr[M(D_1) \in O] \leq \exp(\varepsilon) \cdot \Pr[M(D_2) \in O] \tag{4}$$

After Eq. (4), this procedure M is deemed to uphold ε -differential privacy, where ε is the privacy parameter governing the probability ratio under which M yields identical computational

outcomes for D_1, D_2. The smaller ε is, the stronger the privacy guarantee becomes. If $\varepsilon = 0$, privacy safeguarding reaches its maximum level, implying that M has identical likelihoods of generating the same outputs for D_1, D_2.

For a query function q, introducing disturbance into the output of the retrieval function represents the fundamental technique for realizing differential privacy. Typical approaches for incorporating noise encompass the Laplace mechanism and the exponential mechanism. The Laplace mechanism is appropriate for numerical results, whereas the exponential mechanism is frequently utilized for non-numeric results. The quantity of noise necessary to meet differential privacy requirements is determined by the global sensitivity, which denotes the maximal influence that removing a single data entry exerts on the query outcome.

Definition 2 (Global Sensitivity). Let $q : D \to R^d$ be a query function, where D represents the input dataset and R^d denotes the d-dimensional real-vector output of the query. For adjacent datasets D_1, D_2 that vary by one data entry, the global sensitivity of query function q is specified as:

$$\Delta q = \max_{D_1, D_2} \left\| q(D_1) - q(D_2) \right\|_l \tag{5}$$

In Eq. (5), l represents the vector norm used to measure distance, and the 1-norm is commonly applied.

Definition 3 (Laplace Mechanism). Considering a query function $q : D \to R^d$, the global sensitivity of q is Δq, provided the output of randomized algorithm M fulfills:

$$M(D) = q(D) + \left\langle Lap\left(\frac{\Delta q}{\varepsilon}\right) \right\rangle^d \tag{6}$$

The stochastic technique M in Eq. (6) fulfills ε-differential privacy. Specifically, $\left\langle Lap\left(\frac{\Delta q}{\varepsilon}\right) \right\rangle^d$ represents a d-dimensional stochastic vector adhering to a Laplace distribution with scale parameter $\frac{\Delta q}{\varepsilon}$.

Definition 4 (Exponential Mechanism). For a utility function $u : (D, Q) \to R$, D represents the input dataset and Q denotes the non-numeric output set of query function q. Regarding any candidate $s\left(s \in Q\right)$ within Q, the global sensitivity of the utility function $u : (D, s)$ equals Δu. If randomized algorithm M generates s with probability proportional to $\exp\left(\frac{\varepsilon q(D, s)}{2\Delta u}\right)$, specifically $M(D, q) = \left\{ s : \Pr[s \in Q] \propto \exp\left(\frac{\varepsilon q(D, s)}{2\Delta u}\right) \right\}$, then M fulfills ε-differential privacy.

3. A Model for Cross-Enterprise Financial Data Federation Privacy Protection and Collaborative Auditing

3.1. On-premises model training

Conventional distributed machine learning necessitates gathering regional data from every involved node onto a centralized server for model optimization. Federated learning facilitates distributed machine learning while maintaining data on individual nodes, thereby avoiding source-data exposure. Assume that the number of contributors is N and that the contributor set is $P = \{P_1, P_2, \cdots, P_N\}$. For node P_k, its local labelled dataset is $d_k = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$, where x_i denotes the input feature and y_i denotes the anticipated output. The union of local datasets is $D = \cup d_i$. Assuming model parameters $m_k = \{m_1, m_2, \cdots, m_m\}$ are optimized locally on node P_k, the goal is to achieve the global optimization model $M_G = h_m(x^i)$ that reduces loss $L(M_G)$ over all participant datasets D. The loss function for node P_k regarding dataset d_k is formulated as:

$$L(h_{m_k}(x^i)) = \frac{1}{|d_k|}\sum_{j\in d_k} f_j\left(h_{m_k}(x^i), y_j\right) \tag{7}$$

In Eq. (7), $f_j(h_{m_k}(x^i), y_j)$ represents the loss metric for data label (x_j, y_j) based on model $h_{m_k}(x^i)$. The training goal for participating node P_k during T iterations is to refine the local model $h^*_{m_k}(x^i)$ under privacy-preserving constraints in order to reduce its loss function, which is:

$$\begin{gathered} h^*_{m_k}(x^i) = \underset{m_k\in\{m_k(t);t<T\}}{\arg\min}\ L\left(h_{m_k}(x^i)\right) \\ \text{s.t.}\Pr(m_k \in \mathbb{R}_d) \le e^{\grave{o}} \Pr\left(m'_k \in \mathbb{R}_d\right) \\ \forall P_k \in P, k \in (1,2,\cdots,N) \end{gathered} \tag{8}$$

In Eq. (8), $m_k(t)$ represents the collection of parameters within the t -round collaborative learning process, where T signifies the maximum quantity of weight-optimization cycles. The condition $\Pr(m_k \in R_d) \le e^{\grave{o}} \Pr(m'_k \in R_d)$ indicates the differential privacy constraint for modifying weights m_k.

Stochastic gradient descent (SGD) is utilized for local node training to minimize the loss function. SGD adjusts the weights in the negative direction of the objective function's gradient to decrease the loss, and this gradient is calculated as:

$$\nabla L(h_{m_k}(x^i)) = \frac{\partial L(h_{m_k}(x^i))}{\partial m_k} \tag{9}$$

For node P_k throughout iteration t, the modification of model coefficients is formulated as:

$$m_k(t) = m_k(t-1) + \alpha_t \cdot \nabla L\left(h_{m_k}(x^i)\right) \tag{10}$$

Within this framework, α_t denotes the magnitude of the update in the direction contrary to the gradient, namely the learning rate, which enables the model to steadily approach its optimum by moving opposite to the derivative of the loss function.

3.2. A Blockchain Consensus Mechanism Based on Dual-Factor Contribution

In this framework, blockchain entities are categorized into primary nodes, consensus nodes, and general nodes. Consensus nodes are selected by regular nodes according to model performance, whereas the primary node is the consensus node with the highest behavioral contribution. Every blockchain entity is linked to multiple federated learning agents. Throughout each training cycle, the parameter submitted by client i and evaluated by blockchain node j is scored according to Eqs. (11) and (12):

$$score_i^k = \frac{1}{\frac{1}{S_v}\sum_{(x_n,y_n)\in D_v} L\left(f\left(\tilde{w}_i^k, x_n\right), y_n\right)} \tag{11}$$

$$\tilde{w}_i^k = w_i^{k-1} - \nabla\tilde{f}\left(w_i^k\right) \tag{12}$$

In Eqs. (11) and (12), D_v denotes the verification dataset supplied by the project assigner; S_v represents the test-set size; $\tilde{w}_i^k$ indicates the model weights delivered by client i throughout the k -th learning cycle; and $L\left(f\left(\tilde{w}_i^k, x_n\right), y_n\right)$ signifies the network error metric.

The impact of every blockchain participant is established through the aggregate rating of the users linked to it. The contribution of blockchain node j in round k, denoted as con_j^k, incorporates the current ratings and the historical contribution record. However, calculating only the contribution of the current round is unfair to nodes with high historical contributions but low current contributions; therefore, their historical contributions must be taken into account. To prevent nodes with high

historical model contributions from reducing their participation enthusiasm in the system, newer model contributions should be assigned higher weights.

Consequently, to assess the comprehensive influence of past models for vertices, this approach employs a temporal-decay mechanism derived from Newton's cooling principle to compute the legacy model weight scores of these entities.

The calculation process for con_j^k is as follows:

$$con_j^k = \sum_{r=1}^{k} score_j^r \times e^{-\alpha(k-r)} \tag{13}$$

$$score_j^r = \sum\nolimits_{i \in I_j} score_i^r \tag{14}$$

In Eqs. (13) and (14), k denotes the current training iteration; $score_j^r$ signifies the contribution metric of node j during round r. The client set I_j is associated with node j; $e^{-\alpha(k-r)}$ indicates the weight allocated to the r-th round score when computing the k-th round contribution; and α is the decay coefficient derived from Newton's cooling principle. A larger value produces faster historical-weight decay.

Poisoning attacks are a prevalent threat within federated learning because they undermine distributed training by deliberately altering local data instances or submitting erroneous gradient vectors. To counteract such attacks, this study introduces a contribution-oriented federated aggregation framework. During each global-gradient aggregation round, gradients from nodes with superior selection scores S are accumulated and weighted according to participant ratings. The universal model is consolidated as follows:

$$g_k = \frac{1}{score_L} \sum\nolimits_{score_i^k \in L} \nabla \tilde{f}(w_i^k) \times score_i^k \tag{15}$$

In Eq. (15), L denotes the selected high-score client-evaluation set, while $score_L$ signifies the total score of the identified user assessments.

Once every client has submitted their gradients or exceeded the specified waiting period, the framework shall commence the selection process for the consensus committee. Compute the model contribution metrics for each node and subsequently establish a consensus committee comprised of those nodes possessing the greatest model contributions.

All the member nodes must remain functional for the upkeep of a distributed ledger framework, and their reliability depends on the smooth operation of each node. Consequently, this research proposes a node activity merit system to evaluate performance and conduct, thus enhancing the likelihood that reliable nodes will engage in the distributed ledger's operations. Nodes which successfully fulfill the network's requirements will receive behavioral contribution scores; conversely, a specific quantity of these credits shall be removed. Following the selection of the Consensus Committee participants, according to the nodes' behavioral scores, one particular node will be selected as the coordinator for block formation.

The actions of a participant encompass block packaging, committee agreement, and client-variable validation. The rewards related to these three tasks are represented by $\langle b_1, b_2, b_3 \rangle$. For blockchain participant j, the reward for every activity is calculated as follows:

$$b_1(j) = count_1(j) - \alpha_1 \cdot \overline{count_1(j)} \tag{16}$$

$$b_2(j) = count_2(j) - \alpha_2 \cdot \overline{count_2(j)} \tag{17}$$

$$b_3(j) = \sum_{r=1}^{k} \frac{V_r(j)}{N_c} \tag{18}$$

In Eqs. (16)-(18), $count_1(j)$ denotes the number of times node j successfully produced a block on time after being selected as leader; $\overline{count_1(j)}$ indicates failed block-packaging attempts within the designated interval; α_1 is the penalty factor for unsuccessful block packaging; $count_2(j)$ signifies successful committee-agreement participation; $\overline{count_2(j)}$ denotes committee-agreement failures; α_2 is the penalty weight for committee-agreement breakdown; $V_r(j)$ represents the number of users

validated by node j during period r. Node j's total behavioral performance $BS(j)$ is determined by prioritizing the three behavior categories:

$$BS(j) = \delta_1 \times b_1(j) + \delta_2 \times b_2(j) + \delta_3 \times b_3(j) \tag{19}$$

In Eq. (19), $\delta_1, \delta_2, \delta_3$ denotes the proportions allocated to the three categories of conduct, satisfying $\delta_1 + \delta_2 + \delta_3 = 1$. Because block packaging and agreement formation are vital for distributed-ledger robustness, the three values should follow $\delta_1 > \delta_2 \geq \delta_3$.

To diminish the concentration of blockchain authority, this research incorporates a leader-node cooling strategy into the consensus protocol. By establishing a node-election cooling period from q to q, the approach avoids any individual node repeatedly dominating block-production authority.

During the leader selection process, nodes within the consensus group are prioritized according to their behavioral contribution metrics. After the consensus assembly and leader are established, the designated leader gathers the global gradient g_k for this cycle and produces the corresponding block $Block_k$.

The primary node has recently generated a block, and currently, a PBFT consensus cycle is being executed within the network, encompassing the four stages of pre-prepare, prepare, commit, and reply. Once the consensus committee members reach a consensus regarding the new block's validity, the client, the primary node, and the consensus committee nodes which have provided valid updates will all receive blockchain incentives according to their contribution metrics. Provided the count of Byzantine nodes is $\leq \frac{1}{3}$, then PBFT may still accomplish eventual consistency.

3.3. Differential Privacy for Intermediate Parameters

To realize differential privacy within local training, this research employs a Laplace mechanism derived from LDP to inject noise into intermediate parameters during the model updating stage, which is mathematically expressed as:

$$\begin{aligned} \tilde{m}_k(t) &= m_k(t-1) \\ &+ \alpha_t \cdot \left(\nabla L(h_{m_k}(x^i)) + \frac{1}{1+e^{-H_k(f(x_i),y_i)}} \cdot \langle Lap(s/\text{ò}) \rangle \right) \end{aligned} \tag{20}$$

In Eq. (20), $\langle Lap(s/\text{ò}) \rangle$ denotes Laplace noise, s indicates local sensitivity, ò denotes the privacy budget, and $H_k(f(x_i), y_i)$ signifies the cross-entropy metric for assessing model quality; a lower value suggests superior model quality. The adaptive privacy-noise tuning coefficient $\frac{1}{1+e^{-H_k(f(x_i),y_i)}}$ is normalized as $\gamma = \frac{1}{1+e^{-H_k(f(x_i),y_i)}} \in [0,1]$ and regulates the noise magnitude using cross-entropy. The privacy budget is bounded by ò; the iteration limit is T; the accumulated budget is 蝌 $\sum_{t=1}^{K}$ $_t$; during the t-th iteration, the privacy allocation is ò$_t = \frac{\text{ò}}{T}, 1 \leq t \leq T$; if the accumulated budget surpasses ò, training halts.

By integrating Laplace-based differential privacy mechanisms into intermediate gradients throughout the federated learning model aggregation phase, the likelihood of local data exposure triggered by inference attacks is mitigated. Every model-training epoch fulfills ò-differential privacy. Relying on the sequential composability of differential privacy, the budget terms ò and 蝌$_2$ + ⋯ + 蝌 ≤ satisfy ò; consequently, the aggregated model maintains the corresponding privacy guarantee.

Evaluating the influence of local differential privacy on errors within intermediate gradient variables, LDP guarantees privacy security via noise injection. The asymptotic error bound depends on the candidate-domain size d, the number of candidate values k, the privacy budget ò, and the number of participants n. For RAPPOR-style local differential privacy, the asymptotic error bound is $O\left(\frac{dk}{\text{ò}\sqrt{n}}\right)$.

3.4. Adaptive Model Aggregation Algorithm

Gather regional models from training servers and utilize blockchain nodes for collective synthesis. The primary conventional model fusion technique is Federated Averaging (FedAvg), yet it fails to resolve the issue of diminished collective model performance triggered by the incorporation of substandard models within the integration procedure. This manuscript suggests an adaptive model integration framework (FedAdp) derived from FedAvg. By utilizing outputs from model quality assessments and reputation ratings of participants throughout the cooperative phase, FedAdp modifies the integration weights to assign greater importance to high-quality model variables in the merged model, thereby enhancing the precision of the synthesized model.

Model performance evaluation is mainly decided by the node loss in the t-th localized optimization round. Cross-entropy $H(f(x_i), y_i)$ is used to gauge the local model error of participant P_k. The quality-evaluation weight $Q_i(P_k)$ for the dynamic model integration approach is formulated in the i-th round as:

$$Q_i(P_k) = 1 - \frac{H_k(f(x_i), y_i)}{\sum_{j=1}^{N} H_j(f(x_i), y_i)} \tag{21}$$

To avoid cooperative entities from submitting deceptive model-performance assessment outcomes, this research utilizes a collective non-synchronous weight validation procedure. Participants disseminate network weights W and cross-entropy values $H(f(x_i), y_i)$ through the blockchain channel, and joint nodes use the private annotated collection d_u to authenticate the parameter values of node P_k.

Whenever the framework consolidates the evaluation importance factors across every vertex, the data are stored on the distributed ledger. The reliability achievement of vertex P_k throughout the integration phase is denoted by the accumulated trust metric $S(P_k)$.

$$S(P_k) = \sum_{i=1}^{\tau} \frac{1}{1 + e^{-\log(Q_i(P_k))}} \Big/ \tau \tag{22}$$

In Eq. (22), τ denotes the frequency at which node P_k has engaged in model synthesis, while $Q_i(P_k)$ signifies the quality-evaluation weight stored on the blockchain for node P_k throughout its i-th participation. The accumulated score $S(P_k)$ is used as the long-term credibility rating of node P_k within federated learning.

At present, the procedure for global model updating computation within the adaptive model aggregation framework may be depicted as:

$$m_G(t) = m_G(t-1) + \frac{1}{N} \sum_{k=1}^{N} S(P_k) \cdot Q_i(P_k) \cdot \tilde{m}_k(t) \tag{23}$$

In Eq. (23), N denotes the number of nodes involved in aggregation; $Q_i(P_k)$ signifies the importance factor allocated to the local model quality evaluation of node P_k during the i-th aggregation step; and $S(P_k)$ indicates the credibility ranking of node P_k across previous stages.

The framework reaches convergence when the cross-entropy for the global model $m_G(t)$ remains below the preset threshold H_T, or when the iteration counter reaches the maximum limit T. The global model term $m_G(t)$ is therefore evaluated under the following stopping criterion:

$$\begin{aligned} M_G &\leftarrow m_G(t) \\ s.t. H(m_i(t)) &< H_T \,|\, t = T \end{aligned} \tag{24}$$

3.5. Grouped Aggregation Quality Audit

Collective validation of regional models is employed to locate a cluster containing a deviant instance, and subsequently more accurately pinpoint the precise faulty model. The primary objective of the protective target for this quality evaluation scheme is to identify any architectures that exhibit issues within monetary information and to assess the magnitude of such difficulties.

Participants evaluate each unencrypted regional architecture $state_i$ to acquire a quality certification and submit it together with the obfuscated version to the distributed ledger. During the audit of a specific cohort, the collective mean serves as the benchmark for identifying deviant structures. The horizontal threshold τ and vertical threshold τ are determined as follows:

$$\tau_{r_x} = \frac{1}{|r_x|} \sum_{m_i \in r_x} state_i \tag{25}$$

$$\tau_{c_y} = \frac{1}{|c_y|} \sum_{m_i \in c_y} state_i \tag{26}$$

In Eqs. (25) and (26), τ_{r_x} and τ_{c_y} denote the boundaries for horizontal and vertical aggregation, respectively. This approach first evaluates local models horizontally and then vertically. During horizontal-cluster examination, the horizontal aggregate model $m_{r_x} = \frac{1}{|r_x|} \sum_{m_i \in r_x} m_i^r$ is evaluated against τ_{r_x} ; if the precision drops below this boundary, all models in the horizontal cluster are identified. After horizontal grouping, the vertical aggregate model $m_{c_y} = \frac{1}{|c_y|} \sum_{m_i \in c_y} m_i^c$ is evaluated against τ_{c_y} ; if the integrated model falls below the boundary, the corresponding model is categorized as suspected defective.

Upon performing cross-sectional and longitudinal clustering evaluations, a collection of prospective problematic frameworks PQ is obtained. For each framework m_i within PQ , if it represents the sole entry in its horizontal or vertical grouping, the following conditions are checked:

$$\left| \left\{ id \,\middle|\, id \in PQ \,\&\, id \in \left\{ r_x \,\middle|\, id \in r_x \right\} \right\} \right| = 1 \tag{27}$$

$$\left| \left\{ id \,\middle|\, id \in PQ \,\&\, id \in \left\{ c_y \,\middle|\, id \in c_y \right\} \right\} \right| = 1 \tag{28}$$

If either condition in Eqs. (27) and (28) holds, the candidate model is formally categorized as a problematic model and incorporated into the issue-framework collection P ; otherwise, it remains in the collection of prospective issue frameworks.

Scenario 1: Within the same segment, all defective models are categorized into a specific horizontal or vertical cluster. When all faulty models reside within a single horizontal cluster, the two-stage audit labels the defective unit as the sole probable defective unit in its vertical grouping and includes it in the defective-model collection P .

Scenario 2: Among identical segments, defective frameworks are situated in distinct horizontal and vertical clusters. In this case, the integration hub can only conclude that faulty units exist within the probable-defect set PQ ; it cannot pinpoint the exact defective framework.

4. Experiments and Analysis of Results

4.1. Blockchain Implementation

To evaluate transactional throughput (TPS) and transactional processing rate (TPSD), this framework was tested within an Ethereum consortium-chain network. By adjusting the difficulty parameter in genesis.json, block-creation overhead was examined under different settings. Equations (29) and (30) assess chain efficiency by incorporating the block interval Δt , the gas limit per transaction, and the Ethereum block GasLimit:

$$TPS = (GasLimit / gaslimit) / \Delta t \tag{29}$$

$$TPSD = \Delta t / (GasLimit / gaslimit) \tag{30}$$

The diagram illustrates that TPS plus TPSD exhibit a direct correlation with block duration; these metrics are labeled as TPS and TPSD accordingly. When block duration expands, system efficiency diminishes; consequently, TPS and TPSD are lowered. Block Time also impacts the client-side reaction

velocity. Reduced block duration allows for a faster response to users. A more compact block interval can enhance both processing speed and the volume of transactions per block; nevertheless, simultaneously, this heightens the intensity of node rivalry and complicates the transaction validation procedure, causing certain transactions to be dismissed or postponed. Creating blocks too quickly will burden the nodes; hence, network reliability will suffer. Consequently, when selecting a block interval, we must reconcile the demand for high-throughput and rapid transaction execution with network integrity. Considering the necessity to harmonize efficiency and stability, the block interval for this architecture is established at 1 second. Such configurations will function optimally and be more accessible for end-users.

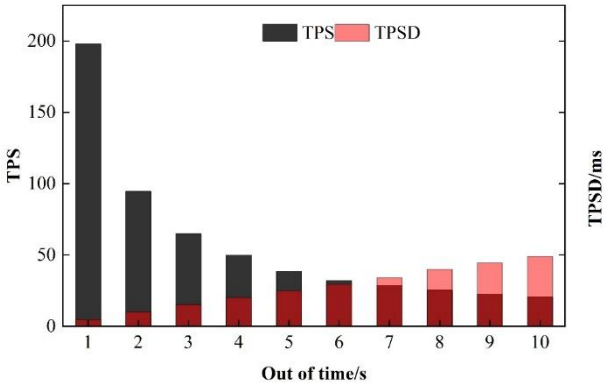


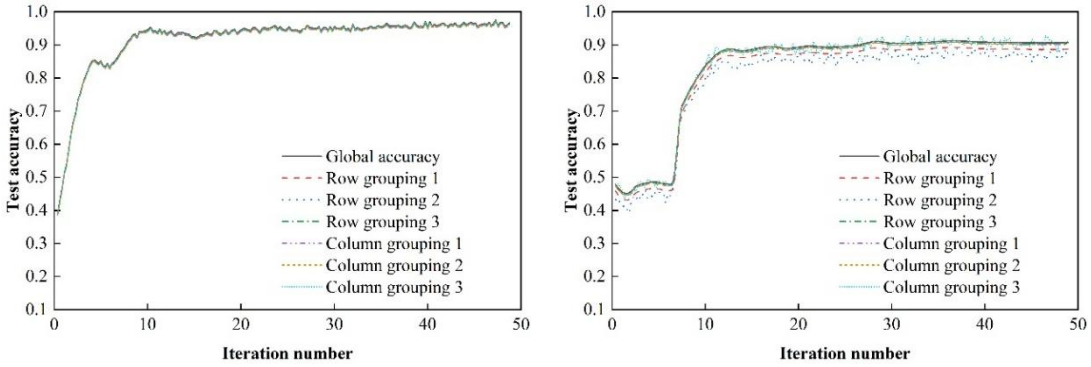
Figure 1. Time Series Relationship of TPSP and TPSD

4.2. Safety Analysis

In this subsection, deep learning frameworks for image identification are developed utilizing the MNIST and CelebA collections. Ten CPU nodes (Hygon C86 7159 featuring 16-core chips) were utilized to mimic a primary server plus nine data custodians for model optimization regarding the MNIST dataset. Regarding model refinement on the CelebA collection, a solitary GPU machine (equipped with 8 NVIDIA Tesla T4 accelerators, 16GB per unit) was employed to process data holders in stages. The behavior sequence and audit sequence modules are executed on six CPU nodes. The deep learning training process is performed within Python (3.8.3) and the PyTorch (1.6.0) package, while data serialization and protection utilize the Charm-crypto (0.5.0) and NumPy (1.18.5) tools.

1) No Malicious Attack

To emulate federated learning simulations, the data on every CPU is partitioned into multiple identical portions prior to the commencement of training. These CPUs choose unique subsets for their local optimization. The global training batch size is fixed at 50, while the local batch size is 1, with a global learning rate of 0.0001; every 10 batches, the learning rate is decreased by half. For the MNIST dataset, the subset size was 800, whereas the training batch size was 50. The CelebA dataset subset size was configured to 1280, utilizing a training batch size of 64. During this trial, the model parameters at identical iterations were arranged into a 3x3 matrix, featuring 3 row groups and 3 column groups. Initially, using the MNIST dataset, we monitored the training precision and loss metrics in the absence of any malicious interference. The model's precision in the absence of malicious activities is depicted in Figure 2. Without malicious interventions, the training precision on the MNIST dataset rose quickly, reaching 90.78% following 10 epochs; ultimately, it peaked at 96.33% upon completion of the process. The model began to demonstrate improved accuracy starting from the 15th iteration on the CelebA dataset, reaching 88.98%, and eventually attaining a peak of 92.12%.



(a) MNIST (b) CelebA

Figure 2. Accuracy of Model without Malicious Attack

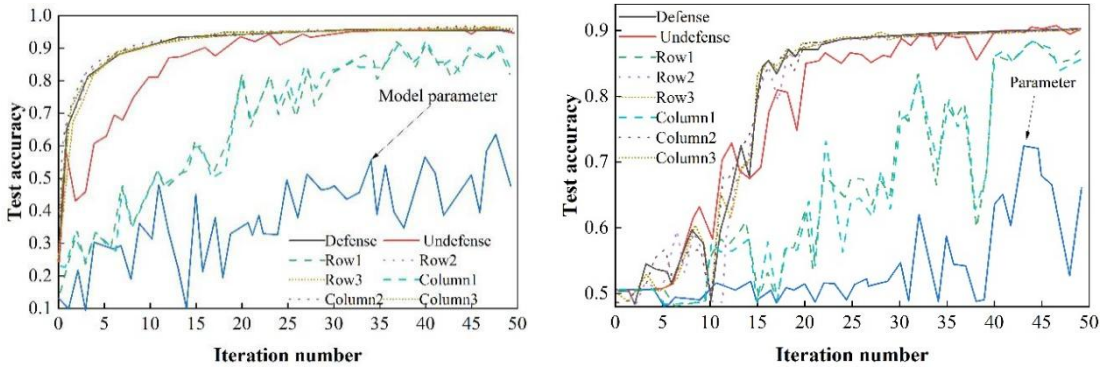
2) Malicious Poisoning Attacks

Illustration 3 depicts the model's precision following malicious poisoning assaults. Both datasets indicate that the poisoned framework's performance remains under 60% and fluctuates severely; thus, the stability of Row Group 1 and Column Group 1 is highly variable, while other clusters demonstrate relatively consistent optimization and accuracy gains. Since no defense verification mechanism was applied, the global model exhibited merely a minor decline in precision and remained relatively stable. Consequently, attacks involving malicious data poisoning on the global model are challenging to identify throughout the training phase.

The examination of fluctuations in numerical loss metrics further reveals the following insights and findings:

(1) This harmful contamination framework demonstrates elevated loss metrics relative to alternative architectures, leading to increased loss figures for the collective models within its category. Regarding the MNIST dataset, the loss figure for the rogue poisoned model peaked at 70.17, thereby increasing the group-integrated model losses to 10.08 and 10.93—significantly higher than those of other clusters. Within the CelebA dataset, the loss metric for the deceptive poisoned model likewise raised its group's loss, varying between 0.82 and 1.53. Consequently, toxic information is discerned by contrasting loss statistics among different categories.

(2) The harm triggered by a contaminated model on a group-aggregated framework is significantly more severe than that on a global-aggregated structure. Because the global-aggregated architecture stays in a sluggish convergence phase, it proves challenging for trainers to identify whether the global-aggregated system has been tampered with. Group aggregation monitoring enables model poisoning to be more readily identified within the collective aggregation system, assisting in the identification of data corruption. In defensive measures, the universal model preserves an accuracy level nearly identical to that of the unspoiled group aggregation model, achieving 91.39%, whereas the precision level of the global model during poisoning reaches 90.82%, showing that this monitoring approach can successfully protect against financial data contamination.



(a) MNIST (b) CelebA

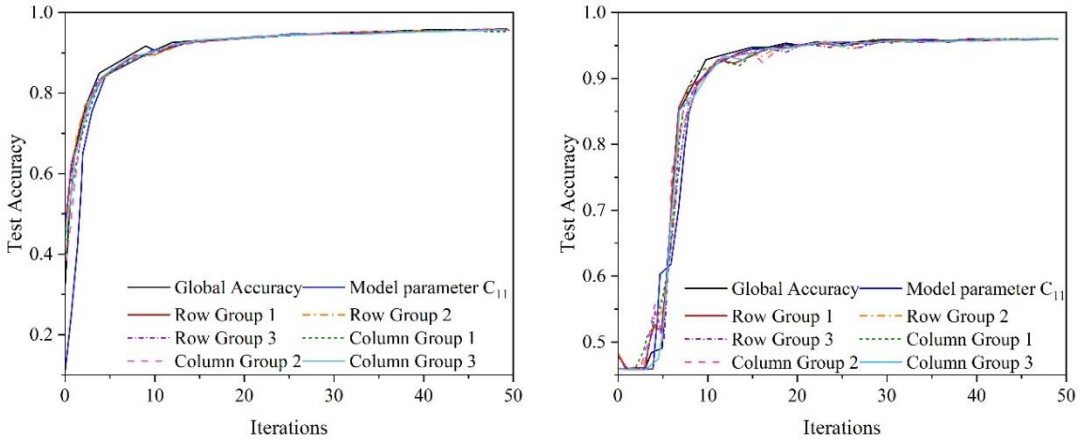
Figure 3. Accuracy of Model Accuracy in Malicious Attack

3) Occurrence of Free-Rider Problems

Regarding free-rider assaults, the typical parameter C11 for row group 1 and column group 1 was deliberately replaced by a malicious model in this trial, specifically an untrained one. The testing accuracy and loss metrics are illustrated in Figures 4 and 5. Such free-riding tactics inflict minimal harm on the model or diminish its ultimate training precision, yet they significantly impair training velocity and hence retard the convergence rate of the framework. These aforementioned assaults are particularly potent during the interval when the overall model's accuracy rises swiftly. Subsequent evaluations and findings can be deduced from the variations in the loss measurement:

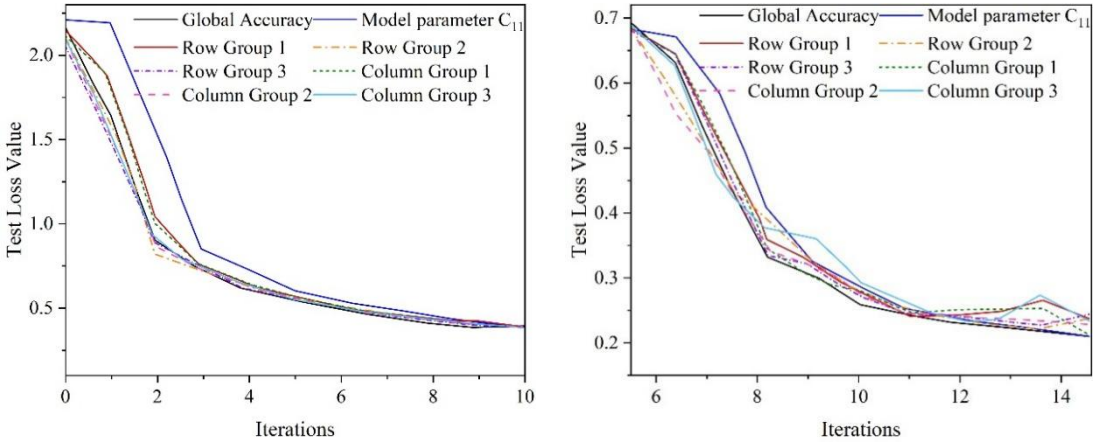
(1) The loss value of the free-riding model is slightly higher than that of other models, causing the aggregate model loss of its group to increase as well, though to a lesser extent than in poisoning attacks.

(2) The verification algorithm only slightly reduces the global model loss during training, but the difference is minimal, making free-riding attacks difficult to detect. However, free-riding attacks become more pronounced during group aggregation and are thus easier to detect.



(a) MNIST (b) CelebA

Figure 4. Displays the model's precision regarding the free-hitch attack.



(a) MNIST (b) CelebA

Figure 5. The model loss value of a hitchhike attack

To identify free-riding intrusions more precisely, this methodology integrates relative model-performance evaluation. The derived metric is represented as $(C_{ij},[a,b])$. Regarding parameter C_{ij}, relative contribution a is used; a lower value indicates a higher probability of free-riding. The parameter b denotes the relative separation between C_{ij} and other models. As shown in Table 1, C_{11} ranked among the lowest three in relative contribution throughout 8 of the first 9 epochs, whereas C_{12} and C_{13} from the identical group appeared only 4 and 3 times, respectively, and their contributions exceeded C_{11}'s in most scenarios.

Table 1. Model Quality Scores for MNIST Datasets

Training batch	Audit results		
1	(C_{13},[0.987,0.999])	(C_{11},[0.992,0.999])	(C_{12},[0.996,0.999])
2	(C_{11},[0.000,0.000])	(C_{21},[1.073,0.935])	(C_{23},[1.111,0.962])
3	(C_{23},[0.000,0.000])	(C_{31},[1.072,0.989])	(C_{33},[1.084,0.992])
4	(C_{11},[0.969,1.000])	(C_{31},[0.969,1.001])	(C_{33},[0.985,1.001])
5	(C_{11},[0.977,1.000])	(C_{13},[0.977,1.001])	(C_{12},[0.996,1.000])
6	(C_{23},[0.979,1.000])	(C_{21},[0.983,1.000])	(C_{33},[0.994,1.000])
7	(C_{11},[0.978,1.000])	(C_{21},[0.991,1.000])	(C_{22},[0.994,1.000])
8	(C_{11},[0.994,1.000])	(C_{13},[1.000,1.000])	(C_{31},[1.000,1.000])
9	(C_{12},[0.960,1.000])	(C_{13},[0.973,1.000])	(C_{11},[0.999,1.000])

The aggregation strategy is utilized to compile every client's model from SGX, subsequently creating a unified global model via a basic weighted mean. More precisely, this approach involves computing a weighted mean of these models, where the weight for each client's data points is defined as the inverse of the total sample count derived from that specific participant. Merge to derive the collective model, then evaluate its performance using the validation dataset. Should the precision satisfy the criteria, terminate training and deliver the global model; otherwise, proceed with further iterations.

To assess the extent that TEE enhances the velocity of model synchronization, numerous trials were conducted, and subsequently, the peak, lowest and mean duration expenses for parameter integration within both SGX and non-SGX settings were documented, as displayed in Table 2 and Figure 6. These aforementioned experimental findings may be utilized to evaluate how much SGX boosts the productivity of algorithm implementation and how effectively the efficiency of model pooling functions under cryptographic security.

Table 2. Model Aggregation Time Overhead Contrast

	Mean time overhead	Maximum time overhead	Minimum time overhead
SGX environment	9355.2	11712.5	8088.8
Non-SGX environment	6585.6	10699.5	5622.8

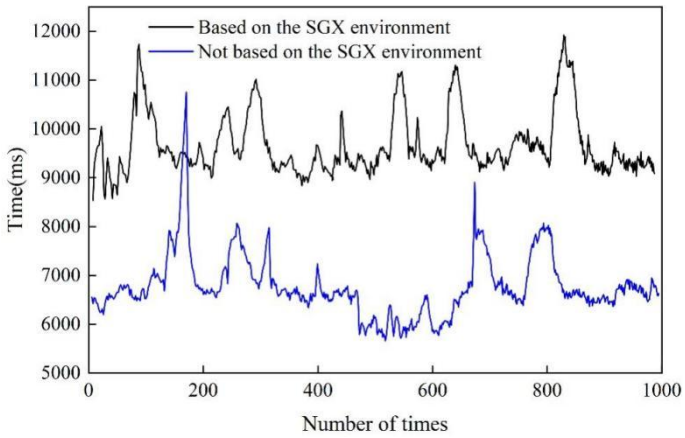


Figure 6. Model Aggregation Time Overhead Contrast

4.3. Performance Analysis of Blockchain Consensus

1) Latency Analysis

The entire network's transaction delay was evaluated with 5, 10, 20, 30, 40, and 50 nodes, performing 20 trials for every setup. According to the aforementioned trials, because of network instability throughout the testing, the individual run delays exhibited minor differences. Consequently, the average of the remaining results from these 20 trials was calculated and a chart depicting the correlation between transaction response time and node count was plotted. Figure 7 illustrates the system performance latency for various node quantities.

Both the conventional PBFT consensus mechanism and the distributed agreement scheme introduced herein have experienced extended consensus durations as the node count rises. The latency

of the PBFT framework also expands nearly exponentially alongside an increment in the total nodes; for example, at 50 participants, it has climbed to around 6.2 seconds of mean latency. Conversely, the response time of the novel consensus protocol presented here grows linearly with the node quantity. For identical node counts, the synchronization period of this innovation is less than that of the classic PBFT mechanism, and such disparity becomes more pronounced with a rise in the node volume. A cluster of 50 nodes will possess a cumulative latency of approximately 1.2 seconds for a single transaction. Considering that local model training within federated learning also requires duration, this gap is entirely tolerable. Moreover, because the suggested consensus structure picks nodes with superior precision to establish a consensus set, in a real-world scenario, where 50 nodes engage in consensus, up to 500 or more nodes might be employed for the complete model computation. Thus, this consensus technique is appropriate for numerous nodes.

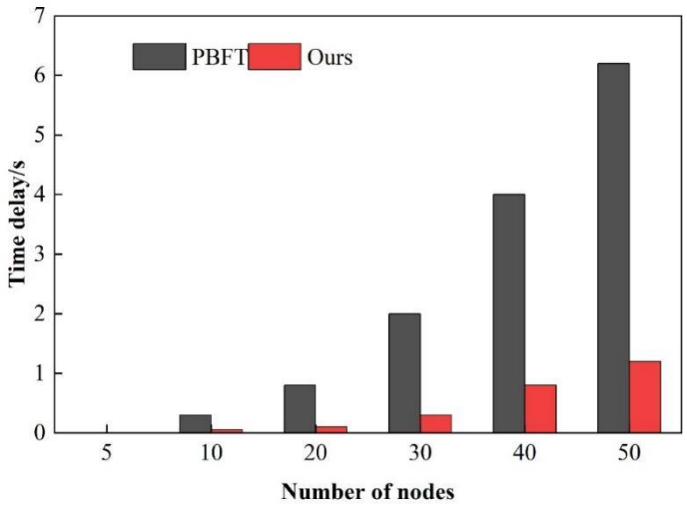


Figure 7. Delay of the system at different numbers of nodes

2) Bandwidth Analysis

When block dimensions remain constant, increasing the node count would concurrently expand the required bandwidth. Within the conventional PBFT consensus framework, the three primary stages—pre-preparation, preparation, and submission—are executed. Each cycle necessitates nodes transmitting block data, thus reaching one consensus requires network throughput approximately threefold greater.

Nodes do not require broadcasting blocks throughout the consensus procedure for the algorithm suggested here. Instead, they are disseminated from the primary node to secondary ones, or vice versa, involving a total of five phases. Consequently, the bandwidth utilization equals 5 times (N-1) $\times$ BlockSize. Given a block volume of 4kB, the bandwidth consumption ratio between the two techniques as the node count grows is illustrated in Figure 8. In contrast to traditional PBFT, the resource demands of the proposed consensus mechanism increase merely linearly with the node quantity and remain significantly less than those of PBFT. According to these experimental findings, it is evident that, under identical circumstances, a framework utilizing the consensus protocol introduced herein possesses a reduced communication bandwidth requirement compared to a system employing PBFT.

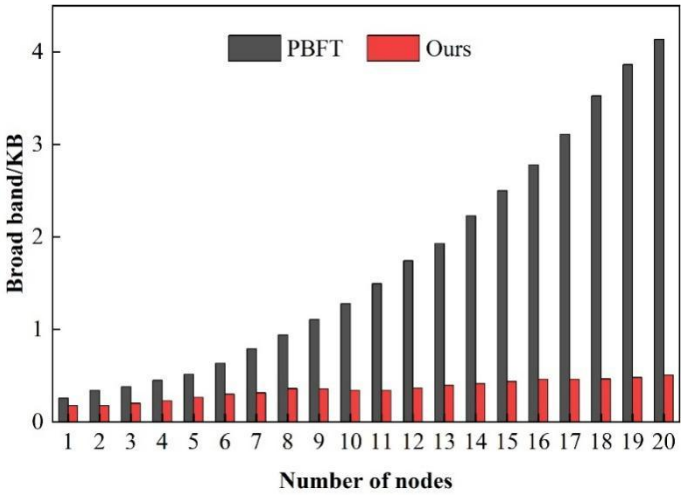


Figure 8. Displays the contrast regarding bandwidth usage among nodes between the two.

5. Conclusion

Owing to the emphasis on safeguarding information privacy, corporations have been compelled to maintain data independently, thereby creating data silos. Considering these specific conditions, this study suggests a framework for cross-organizational fiscal data confidentiality maintenance and cooperative verification utilizing federated learning architectures, blockchain systems and differential privacy techniques.

Without considering deliberate assaults, the learning precision regarding the MNIST data increased consistently and peaked at 90.78% following the 10th epoch; upon completion of optimization, it attained 96.33%. Concerning the CelebA data, the network's performance began to stabilize by the 15th epoch at 88.98% and eventually achieved 92.12%.

Regarding protection, the overall framework's precision remains nearly identical to the collective model's score for the clean dataset, reaching 91.39%. This general scheme's performance drops to 90.82% when facing contamination, confirming that this evaluation strategy is highly efficient for addressing toxic financial information.

3) This auditing mechanism decreases the global model's training loss metric modestly, and such a minor gap is insufficient for identifying a free-riding threat. Free-riding intrusions are more pronounced and simpler to recognize within batch aggregation processes.

4) In contrast to conventional PBFT, as the node count approaches 50, the network's transaction delay remains roughly 1.2 seconds, while bandwidth usage grows linearly alongside the node quantity; this is markedly less than the performance of the PBFT consensus protocol.

To safeguard the confidentiality and safety of individuals, further investigations are required to enhance the training effectiveness and precision of federated learning in reality, and superior approaches must be created.

References

1. Chen, Y., & Xu, P. (2024, April). The Research on Cross-enterprise Collaborative Information System Based on NGIT. In Proceedings of the 5th International Conference on Computer Information and Big Data Applications (pp. 329-333).
2. Hijazi, N. M., Aloqaily, M., Guizani, M., Ouni, B., & Karray, F. (2023). Secure federated learning with fully homomorphic encryption for iot communications. IEEE Internet of Things Journal, 11(3), 4289-4300.
3. Park, J., & Lim, H. (2022). Privacy-preserving federated learning using homomorphic encryption. Applied Sciences, 12(2), 734.
4. Pan, Y., Chao, Z., He, W., Jing, Y., Hongjia, L., & Liming, W. (2024). FedSHE: privacy preserving and efficient federated learning with adaptive segmented CKKS homomorphic encryption. Cybersecurity, 7(1), 40.
5. Wen, J., Zhang, Z., Lan, Y., Cui, Z., Cai, J., & Zhang, W. (2023). A survey on federated learning: challenges and applications. International journal of machine learning and cybernetics, 14(2), 513-535.

6. Zhang, L., Xu, J., Vijayakumar, P., Sharma, P. K., & Ghosh, U. (2022). Homomorphic encryption-based privacy-preserving federated learning in IoT-enabled healthcare system. IEEE transactions on network science and engineering, 10(5), 2864-2880.
7. Cai, Y., Ding, W., Xiao, Y., Yan, Z., Liu, X., & Wan, Z. (2023). SecFed: A secure and efficient federated learning based on multi-key homomorphic encryption. IEEE Transactions on Dependable and Secure Computing, 21(4), 3817-3833.
8. Alluri, R. K. (2023). Federated Learning for Collaborative Fraud Detection Across Institutions. International Journal of Core Engineering & Management, 7(6), 280-292.
9. Alazab, M., Rm, S. P., Maddikunta, P. K. R., Gadekallu, T. R., & Pham, Q. V. (2021). Federated learning for cybersecurity: Concepts, challenges, and future directions. IEEE Transactions on Industrial Informatics, 18(5), 3501-3509.
10. Priyadharshini, D., Yamini, G., Kiruthika, N., Ashin, A., & Thangarajan, N. (2026, January). A Privacy-Aware Federated Learning Framework for Collaborative Performance Analytics Across Distributed Enterprise Management Environments. In 2026 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES) (pp. 1-9). IEEE.
11. Valiki, D. (2025). Federated AI and Cloud Computing Models for Cross-Enterprise Risk Intelligence. International Journal of Advanced Engineering Science and Information Technology (IJAESIT), 8(6), 17772.
12. Suryadevara, S. S. K., & Nakirikanti, S. (2023). Privacy-Preserving Personalization Using Federated Learning in AEM. International Journal of AI, BigData, Computational and Management Studies, 4(4), 190-199.
13. Guo, K., Chen, D., Huang, Q., Li, F., Guo, C., Wu, D., ... & Chen, K. (2023). Privacy-preserving multi-label propagation based on federated learning. IEEE Transactions on Network Science and Engineering, 11(1), 886-899.
14. Chen, L., Guo, Y., Ge, C., Zheng, B., & Gao, Y. (2023). Cross-source Data Error Detection Approach Based on Federated Learning. International Journal of Software & Informatics, 13(1).

About the Author

Fengjuan Yu received the Bachelor of Management degree in Accounting and the Master of Auditing degree in Auditing from the Southwest University of Political Science & Law, China, in 2017 and 2019, respectively. She is currently working in the Tianfu College of the Southwestern University of Finance and Economics as the director of Experimental Teaching Center. Her research interests include artificial intelligence, statistical and deep learning applications in modern accounting and auditing, based on which she has spearheaded the design and development of two provincial level platforms, which are the experimental teaching & research center of accounting and the intelligent virtual reality experimental teaching & research center in finance and artificial intelligence.

Lei Chen (born in 1994) is a native of Chengdu, Sichuan Province, P.R. China. She obtained her Ph.D. from Southeast Asia University in Thailand. Her research interests include ESG and corporate management, as well as accounting and finance teaching and talent development.

Shuai Zeng was born in 1994 in Deyang, Sichuan Province, China. He graduated from Dongbei University of Finance and Economics and obtained a master's degree. His research interests lie in corporate governance and management.

Zhuo Yang was born in 1997 in Deyang City, Sichuan Province, China. She graduated from City University of Malaysia with a master's degree. Her research area is business administration.

Li Yue was born in 1995 in Chengdu City, Sichuan Province, China. She graduated from Southwest University of Political Science & Law with a master's degree.