BRITE-FL: Blockchain-based Reputation and Incentive Token Enhanced Federated Learning for IoT Ecosystems

ARTICLE INFO

Keywords:
Blockchain
Federated Learning
Internet of Things (IoT)
Reputation Mechanism
Token Incentives

ABSTRACT

With the rapid development of Internet of Things (IoT) technologies, effectively utilizing distributed data while preserving privacy has become a critical challenge. This paper proposes a secure federated learning framework integrating blockchain and reputation token mechanisms to address data privacy protection, security maintenance, and participant incentivization in IoT environments.The framework leverages blockchain's decentralization and immutability to design a dynamic reputation token mechanism via smart contracts, which rewards participants based on their model training contributions and long-term performance. To mitigate potential malicious behaviors, the proposed approach introduces multi-dimensional anomaly detection models that comprehensively assess nodes' historical behaviors and current contributions. Moreover, the framework implements a role-based participation model, enabling devices with diverse computational capabilities to engage and receive corresponding incentives.Experimental results demonstrate the framework's robustness: when confronted with data pollution and label flipping attacks from up to 30% malicious nodes, the method limits model performance degradation within 5% and achieves a malicious node detection rate exceeding 94%. The innovative reputation evaluation mechanism systematically distinguishes participant behaviors, causing honest nodes' reputations to progressively increase while malicious nodes' reputations exponentially decline with each malicious act. These findings suggest significant potential for secure and efficient federated learning in IoT and edge computing domains.

1. Introduction

The rapid development and widespread application of Internet of Things (IoT) technology have led to an unprecedented scale of data generation and collection. It is predicted that by 2025, the number of global IoT devices will reach 75 billion [1]. These IoT devices, deployed across smart homes, industrial manufacturing, transportation, and healthcare, continuously generate diverse real-time data[2]. Effectively utilizing these distributed data resources has become crucial in advancing Artificial Intelligence (AI) and Machine Learning (ML) technologies. IoT data-driven AI applications are expected to bring revolutionary changes in smart cities, personalized services, and predictive maintenance, promoting economic development and social progress[3–5].

However, two major challenges have emerged as critical bottlenecks limiting the integrated development of IoT and AI. First, the explosive growth of data has raised significant concerns regarding privacy protection and data security. Traditional centralized machine learning paradigms require aggregating dispersed data to a central server for processing, but this approach faces substantial challenges in IoT environments. The data generated by IoT devices is massive in volume, high in dimensionality, and time-sensitive [6], making large-scale cross-network transmission costly and potentially harmful to devices' limited resources [7]. Second, centrally stored user data becomes vulnerable to network attacks, posing severe risks to user privacy[8]. These challenges are further complicated by strict data protection regulations, such as the European Union's General Data Protection Regulation (GDPR) [9], which imposes specific compliance requirements on personal data handling, exacerbating data silos and limiting cross-regional data sharing.

ORCID(s):

Federated Learning (FL) has emerged as a promising solution to these challenges, offering a novel distributed machine learning paradigm that enables collaborative model training without raw data sharing[10]. In the FL framework, participants (such as IoT devices) train models using local data and only share model updates with a central server for aggregation[11], significantly reducing data transmission volume while protecting privacy[12]. However, several critical challenges remain in practical applications. The heterogeneous nature of IoT devices in hardware configurations, computational capabilities, and network conditions creates an imbalanced distribution of capabilities and data quality among participants[13]. Moreover, in open and dynamic IoT environments, the FL process is vulnerable to malicious nodes that may submit false or harmful model updates (as shown in Figure 1), necessitating robust mechanisms for anomaly detection and participant incentivization[14–16].

While existing research has made progress in addressing these issues, current solutions remain inadequate[10]. Many FL frameworks rely heavily on central server coordination, introducing risks of single-point failure and privacy leakage[10, 17, 18]. Furthermore, these studies often overlook the inherent heterogeneity of IoT environments by assuming uniform computational capabilities across participating nodes[19]. Particularly concerning is the lack of sophisticated mechanisms for handling malicious nodes, as existing reputation-based incentive systems fail to effectively address scenarios where high-reputation nodes engage in malicious activities[20].

Blockchain technology, with its decentralized and immutable characteristics, offers a promising approach to addressing these limitations [21]. By incorporating blockchain into federated learning, systems can achieve peer-to-peer

collaboration without third-party intervention while leveraging consensus mechanisms to suppress malicious behavior [22]. Blockchain's distributed ledgers can securely record model updates, while smart contracts can automate reward and punishment mechanisms, enhancing system transparency and credibility[23–25].

This paper proposes a reputation-driven blockchain federated learning framework specifically designed for IoT environments. Our framework makes several key contributions:

1) Introduces a novel architecture integrating consortium blockchain with federated learning, establishing trust between IoT federations through smart contracts and consensus mechanisms.
2) Introduces a dynamic reputation token system that comprehensively evaluates node contributions based on both current performance and historical behavior, providing targeted incentives for high-quality participation.
3) Designs an innovative role separation mechanism, dividing participating nodes into training nodes and validation nodes to accommodate the heterogeneity of IoT devices. This separation mechanism enhances system security and reliability while providing participation opportunities for devices with varying computational capabilities.
4) Proposes a robust malicious behavior identification method combining multiple anomaly detection models with smart contract-based punishment mechanisms.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 presents the preliminaries of Blockchain Technology, Federated Learning, and Isolation Forest. Section 4 details the proposed framework, including the system model, operation workflow, reputation-based participant selection, and token mechanisms. Section 5 evaluates the framework's performance through comprehensive experiments, and Section 6 concludes the paper.

2. Related work

The integration of blockchain technology with federated learning has emerged as a promising approach to address security and privacy challenges in distributed IoT environments. McMahan et al.[10] introduced the foundational FedAvg algorithm, which enables decentralized model training by aggregating locally computed updates without central data logging. This work established the foundation for subsequent blockchain-FL integration efforts. Building upon this, several researchers have explored blockchain-based FL architectures to enhance system security and trustworthiness. Recent advances include specialized applications such as Zuo et al.[26] who developed Federated TrustChain for blockchain-enhanced large language model training and unlearning, and Zhou et al.[27] who addressed efficient asynchronous federated learning in heterogeneous edge environments.

Awan et al.[28] proposed a blockchain-based privacy-preserving FL framework to secure model updates through

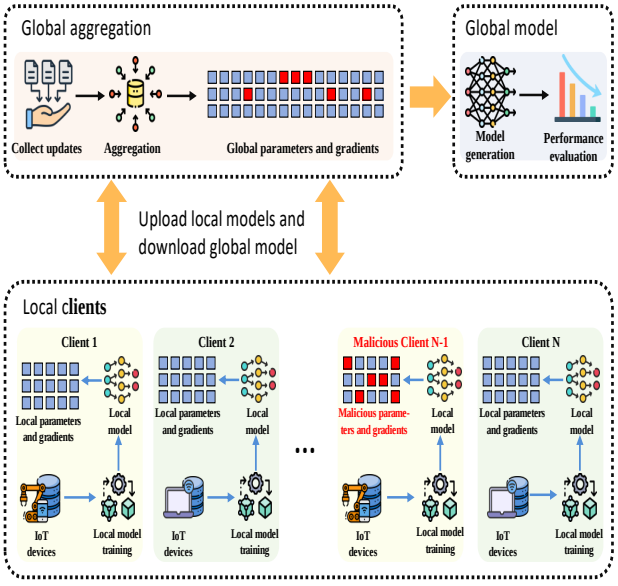

Figure 1: Illustration of FL framework in IoT scenario

smart contracts. Their approach addresses trust issues in centralized FL systems but does not thoroughly consider the computational overhead introduced by blockchain operations. Similarly, Samuel et al.[29] developed a blockchain-based FL framework for healthcare applications, incorporating differential privacy mechanisms. However, their system lacks comprehensive mechanisms for handling malicious participants and long-term participant motivation.

Privacy preservation in FL systems has been extensively studied through various technical approaches. Zhao et al.[30] developed AdaDpFed, an adaptive differential privacy protocol to address attacks like model inversion. While their approach provides privacy guarantees, the noise injection inherently reduces model performance. Lu et al.[31] combined blockchain, FL, and differential privacy by adding calibrated noise to model parameters before blockchain storage. This approach reduced model accuracy due to interference in training, highlighting the fundamental trade-off between privacy and utility in noise-based methods.

Alternative privacy-preserving techniques have also been explored in the context of blockchain-FL integration. Kalapaaking et al.[32] proposed secure aggregation using Trusted Execution Environment (TEE) to provide hardware-based privacy protection. However, their approach suffers from inefficiency due to limited GPU support in TEE environments. Jin et al.[20] addressed privacy concerns through homomorphic encryption, enabling computation on encrypted data. While mathematically sound, homomorphic encryption introduces substantial computational and communication overhead, making it challenging for resource-constrained IoT devices. Recent blockchain infrastructure improvements, such as Yang et al.[33] who developed L2chain for high-performance and confidential layer-2 blockchain solutions, provide new opportunities for efficient privacy-preserving FL deployments. Notably, Wu et al.[34] proposed

a privacy-preserving serverless federated learning scheme for IoT based on secure multi-party computation, eliminating central server dependencies while achieving fault tolerance through secret sharing. Their approach demonstrates formal security guarantees against collusion attacks and maintains accuracy without performance loss, though it focuses primarily on structural privacy rather than dynamic malicious behavior detection.

Incentivizing participant engagement represents another critical challenge in federated learning systems. Sun et al.[23] introduced FLcoin, a cryptocurrency-based reward system to compensate participants based on their contributions. While their approach addresses the free-rider problem, it introduces significant computational costs and long-term motivation challenges remain problematic. Li et al.[35] proposed a blockchain-based crowdsourcing framework that alleviates centralization risks but overlooks FL-specific challenges like model aggregation complexity and heterogeneous data distributions. Recent developments in this area include Wu and Seneviratne[36] who proposed a blockchain-based framework specifically addressing scalable and incentivized federated learning, and Cui et al.[37] who introduced BagChain, a dual-functional blockchain leveraging bagging-based distributed learning for enhanced participant engagement.

The detection and mitigation of malicious behavior in FL systems has received increasing attention as these systems are deployed in open and untrusted environments. Cai et al.[20] systematically analyzed the threat of malicious updates degrading global model quality, particularly in open IoT environments where participants cannot be fully trusted. Zhang et al.[32] proposed clustering-based methods to identify and exclude such malicious nodes from the training process. Xiong et al.[38] developed a blockchain-based reputation system for decentralized participant selection, utilizing blockchain's immutability to maintain participant behavior records. Recent work by Alagha et al.[39] explored blockchain-assisted demonstration cloning for multi-agent deep reinforcement learning, providing new perspectives on collaborative learning security. In domain-specific applications, Dai et al.[40] proposed a personalized federated learning approach for traffic flow prediction that incorporates client reputation mechanisms to evaluate participant credibility and customize model weights accordingly. Their method demonstrates the potential of reputation-based systems in identifying dishonest clients who provide corrupted or incomplete data. However, these approaches often employ static detection mechanisms that lack adaptability to evolving attack strategies in dynamic IoT networks.

Despite these advances, existing approaches face several significant limitations. Traditional FL frameworks rely heavily on central servers, introducing risks of single-point failures and potential privacy leakage. Many blockchain-based FL methods overlook the computational and communication constraints characteristic of IoT devices, leading to impractical resource requirements. Current reputation and incentive mechanisms typically lack the dynamic adaptability needed to effectively handle evolving malicious behaviors and long-term participation challenges. Furthermore, the inherent heterogeneity of IoT devices and the resulting disparities in data quality remain largely underexplored, with most existing frameworks assuming relatively homogeneous participant capabilities.

To address these limitations, our proposed BRITE-FL framework integrates a dynamic reputation system with a novel token-based incentive mechanism specifically designed for heterogeneous IoT ecosystems. Unlike existing approaches that treat security, privacy, and incentivization as separate concerns, our framework provides a comprehensive solution that combines multi-dimensional anomaly detection with adaptive reputation management, effectively isolating malicious participants while maintaining computational efficiency suitable for resource-constrained IoT environments.

3. Preliminaries

3.1. Blockchain Technology

Blockchain technology has revolutionized data management and transaction processing by providing a decentralized, immutable, and transparent framework. At its core, a blockchain is a distributed ledger maintained by a network of participants, each holding a copy of the ledger. The ledger is composed of a series of blocks, each containing a set of transactions, linked together using cryptographic hashes to form a tamper-proof chain [23]. In a permissioned blockchain, such as a consortium blockchain, the network is managed by a group of pre-selected participants responsible for validating transactions and appending new blocks to the chain [21]. This type of blockchain is well-suited for applications that require a controlled environment, such as federated learning, where only authorized participants can join the network and contribute to the process.

One of the key features of blockchain technology is the use of smart contracts, which are self-executing programs stored on the blockchain that can be triggered by specific events or conditions [9]. A smart contract can be formally defined as a tuple $SC = (S, F, R)$, where S represents the state variables, F is the set of functions that can be invoked by the participants, and R denotes the rules and conditions governing the execution of the contract [22]. The execution of a smart contract is triggered by a transaction sent to the blockchain, which invokes one of the contract's functions. The transaction is then validated by the network participants through the consensus mechanism, and if deemed valid, the corresponding function is executed, resulting in a change of the contract's state.The combination of blockchain technology and smart contracts offers several advantages, such as increased security, transparency, and automation. By leveraging the security and immutability of the blockchain, along with the self-executing nature of smart contracts, we can create a trustworthy and efficient environment for collaborative model training and aggregation in the context of federated learning. Blockchain and smart contracts can be utilized to ensure the integrity and confidentiality of

the learning process, while also providing a mechanism for incentivizing participants to contribute high-quality data and model updates[18].

3.2. Federated Learning

Federated learning (FL) is a distributed machine learning paradigm that enables multiple participants to collaboratively train a shared model while keeping their data locally, thereby preserving data privacy [4, 16]. In this setting, a set of participants $\mathcal{P} = \{P_1, P_2, ..., P_N\}$, each holding a private dataset D_i, cooperate to learn a global model with parameters θ. The objective is to minimize the global loss function, which is a weighted average of the local loss functions:

$$\min_{\theta} F(\theta) = \sum_{i=1}^{N} \frac{n_i}{n} F_i(\theta) \tag{1}$$

where N is the number of participants, $n_i = |D_i|$ is the size of the local dataset belonging to participant P_i, $n = \sum_{i=1}^{N} n_i$ is the total number of data samples across all participants, and $F_i(\theta)$ is the local objective function of participant P_i.

The Federated Averaging (FedAvg) algorithm [2] is a popular approach to solve this optimization problem. In each round t of FedAvg, the server selects a random subset of participants $S_t \subseteq \mathcal{P}$ and broadcasts the current global model θ_t to them. The selected participants then perform local training on their respective datasets to update the model parameters:

$$\theta_i^{t+1} = \theta_i^t - \eta \nabla F_i(\theta_i^t) \tag{2}$$

where η is the learning rate. After local training, the participants send their updated models θ_i^{t+1} back to the server, which aggregates them to obtain the new global model:

$$\theta_{t+1} = \sum_{i=1}^{N} \frac{n_i}{n} \theta_i^{t+1} \tag{3}$$

This process is repeated for a predefined number of rounds or until convergence.

Federated learning provides a principled framework for enabling collaborative learning while preserving data privacy. However, it faces challenges such as the potential presence of malicious participants who may attempt to undermine the learning process by sending incorrect or manipulated updates [18]. To mitigate these challenges, various techniques have been proposed to enhance the robustness and security of federated learning systems [19, 20].

3.3. Isolation Forest

Isolation Forest (iForest) is an unsupervised anomaly detection algorithm proposed by Liu et al. [41]. Unlike traditional density or distance-based methods that rely on profiling normal behavior, iForest operates on the principle that anomalous instances are fundamentally different from normal instances and can be more easily separated or "isolated" from the majority of data points [42, 43].

The core mechanism of iForest involves constructing an ensemble of isolation trees (iTrees), where each iTree is built by recursively partitioning the data space through random attribute selection and random split-point determination. The algorithm constructs these trees by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature. This process continues recursively until each data point is isolated in its own leaf node or a predefined tree height is reached.

The fundamental insight underlying iForest is that anomalies exhibit significantly shorter average path lengths in iTrees compared to normal data points. For a dataset of size n, normal instances require approximately $O(\log n)$ steps to be isolated, as they follow patterns similar to the majority of the data. In contrast, anomalous instances can be isolated with substantially fewer partitions due to their distinct characteristics that differentiate them from the dense regions of normal data.

The anomaly score for each data point x is calculated based on its average path length $E(h(x))$ across multiple iTrees in the ensemble:

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}} \tag{4}$$

where $E(h(x))$ represents the average path length of data point x over all iTrees in the forest, and $c(n)$ is the average path length of unsuccessful search in a Binary Search Tree (BST), defined as $c(n) = 2H(n-1) - \frac{2(n-1)}{n}$ for $n > 2$, where $H(i)$ is the harmonic number. This normalization factor enables meaningful comparison of anomaly scores across datasets of different sizes.

The anomaly score $s(x, n)$ ranges from 0 to 1, where values close to 1 indicate strong anomalous behavior, values significantly smaller than 0.5 suggest normal behavior, and values around 0.5 indicate that the data point cannot be clearly distinguished as normal or anomalous. The ensemble approach, utilizing multiple iTrees with different random partitioning strategies, effectively reduces the impact of randomness and improves the robustness and reliability of anomaly detection.

iForest demonstrates several advantageous properties: linear time complexity with low constant factors, minimal memory requirements, and the ability to handle high-dimensional datasets without requiring distance calculations or density estimations. These characteristics distinguish it from traditional anomaly detection methods such as k-nearest neighbors or support vector machines, which typically suffer from the curse of dimensionality and require substantial computational resources for distance or similarity calculations.

In distributed computing environments, iForest's design supports natural parallelization and distributed processing. Model parameters from machine learning systems can be vectorized by flattening weight matrices and bias

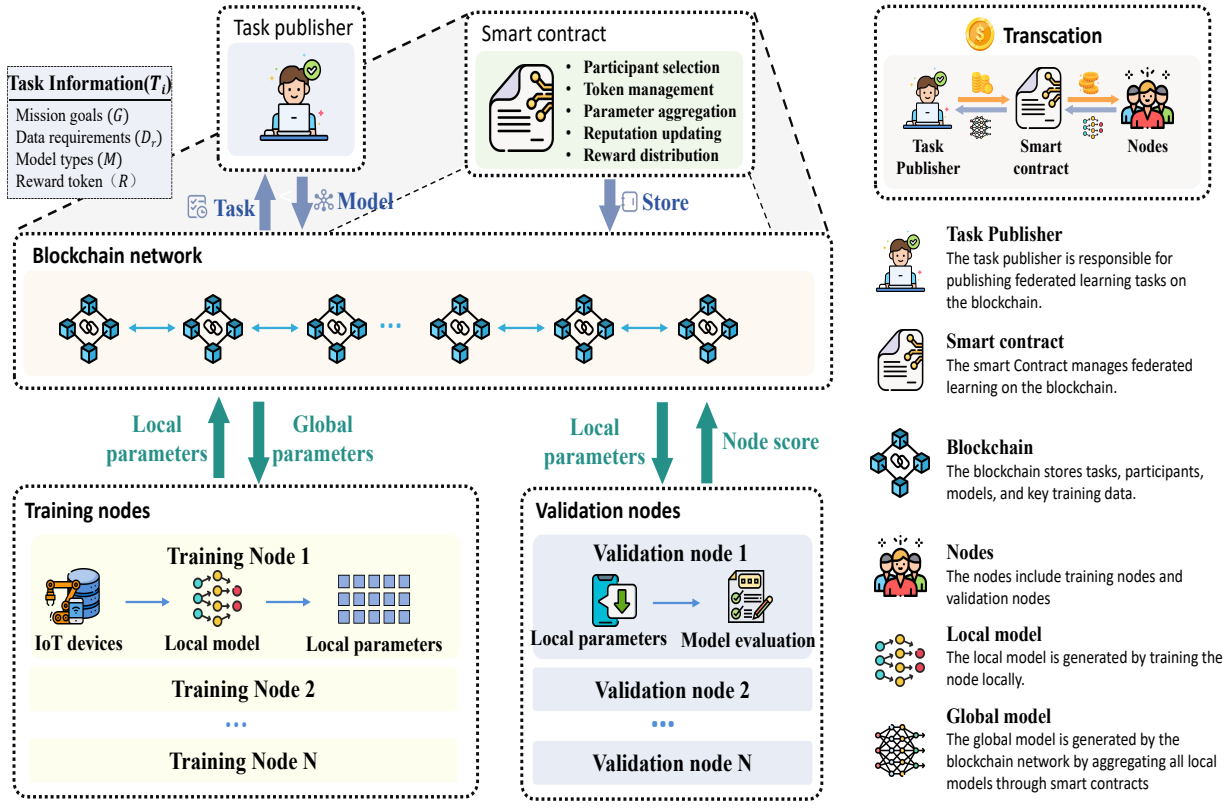

Figure 2: System Model for a Blockchain-based Federated Learning Framework in the Internet of Things (IoT)

vectors into high-dimensional feature vectors, making them amenable to anomaly detection analysis. When multiple validation entities are involved, each can independently construct isolation forests using the same parameter vectors, and their anomaly scores can be aggregated through consensus mechanisms to enhance detection consistency. The algorithm's robustness to noise and its capacity to identify anomalous patterns without requiring prior knowledge of attack signatures make it suitable for detecting malicious model updates, where adversarial modifications often exhibit systematic deviations from legitimate parameter distributions.

4. Proposed protocol

This section proposes a blockchain-based Internet of Things (IoT) federated learning framework designed to promote the participation of high-quality nodes and encourage honest behavior through a reputation token incentive mechanism. The framework primarily consists of four key components: task publishers, participating nodes, blockchain network, and smart contracts. Figure 2 illustrates the overall architecture of the system and the interactions among various components. Table 1 lists the main parameters used in this paper and their definitions.

4.1. System Model

The core components of this framework and their main functions are as follows:

1) Task Publisher (T): As the initiator of the system, responsible for publishing federated learning tasks on the blockchain. Task information includes task objectives (G), data requirements (D_r), model type (M), and total reward tokens (R_{total}). This information is publicly released through the blockchain (B), which ensures that all potential participants can access and understand the task requirements.
2) Participating Nodes: key entities executing federated learning tasks, divided into two roles: training nodes and validation nodes.
   (a) Training Nodes ($n_{ti} \in N_t$): Responsible for providing local data and executing model training. Each training node possesses a local dataset D_{ti}, performs model training based on global model parameters θ and local data, and uploads the updated parameters θ_t to the blockchain network.
   (b) Validation Nodes ($n_{vi} \in N_v$): Responsible for validating the effectiveness of training results. Validation nodes retrieve parameter updates θ_t submitted by training nodes from the blockchain network and evaluate them using a specified test dataset D_v. Evaluation results are uploaded to the smart contract as scores, used to update the reputation values of training nodes.

Table 1
System Parameters Definition

Parameter	Definition
T	Task Publisher
G	Task Goal
D_r	Data Requirement
M	Model Type
R_{total}	Total Reward Tokens
B	Blockchain Network
SC	Smart Contract
N_t	Set of Training Nodes
N_v	Set of Validation Nodes
n_{ti}	The i-th Training Node
n_{vj}	The j-th Validation Node
D_{ti}	Local Dataset of Training Node n_{ti}
D_v	Public Dataset for Validation
$\theta_{ti}^{(t)}$	Local Model Parameters of Training Node n_{ti} in the t-th Round
$\theta_g^{(t)}$	Global Model Parameters in the t-th Round
C	Amount of Staked Tokens
R_i	Reputation value of participant i
$R_{init,i}$	The i-th Node Initial Reputation Value of Participant
S_i	Comprehensive Score of Participant i
L	Loss Function Used for Model Training
f	Aggregation Function
δ_d	Reputation Decay Factor
C_{avg}	Average Amount of Tokens Staked by All Participants
E	Number of Local Update Rounds
R_{min}	Minimum Reputation Threshold
S_j	Final Score of Training Node j
A_j^i	Accuracy of Model Parameters from Training Node j Evaluated by Validation Node i
Z_j^i	Z-Score of Model Accuracy for Training Node j by Validation Node i
s_j^i	Anomaly Score for Training Node j by Validation Node i
$\mathcal{M}_j^{(t)}$	Malicious Behavior Indicator for Node j in Round t
τ	Score Threshold
$N_j^{(t)}$	Cumulative Number of Potential Malicious Behaviors by Node j up to Round t
W_{ti}	Reward for Training Node n_{ti}
W_{vi}	Reward for Validation Node n_{vi}

Participants can choose their roles based on their resources and willingness. They need to stake a certain amount of tokens C during registration. These tokens are locked by the smart contract and converted into initial reputation values R_{init}, serving as proof of qualification for task participation.

3) Blockchain Network (B): As the system's infrastructure, responsible for storing task information, participant information, and model parameters, and recording key data during each training round. The blockchain network design ensures transparency and traceability of the entire federated learning process, avoiding the risk of single-point failure associated with centralized storage.
4) Smart Contract(SC): An automatically executed program deployed on the blockchain network, responsible for managing the entire federated learning process. Main functions include participant selection, token management, training process coordination, parameter aggregation, reputation value updates, and reward distribution. The smart contract selects the final participating nodes (N_t' and N_v') based on participants' reputation values R_i and staked token amounts C_i. After the training process ends, the smart contract performs model parameter aggregation, generating global model updates $\theta = f(\theta_t)$, and distributes the updated model parameters to training nodes to begin a new round of training.

4.2. System Operation Workflow

Figure 3 illustrates the overall system operation process, detailing the interactions between different components. The entire process can be divided into the following steps:

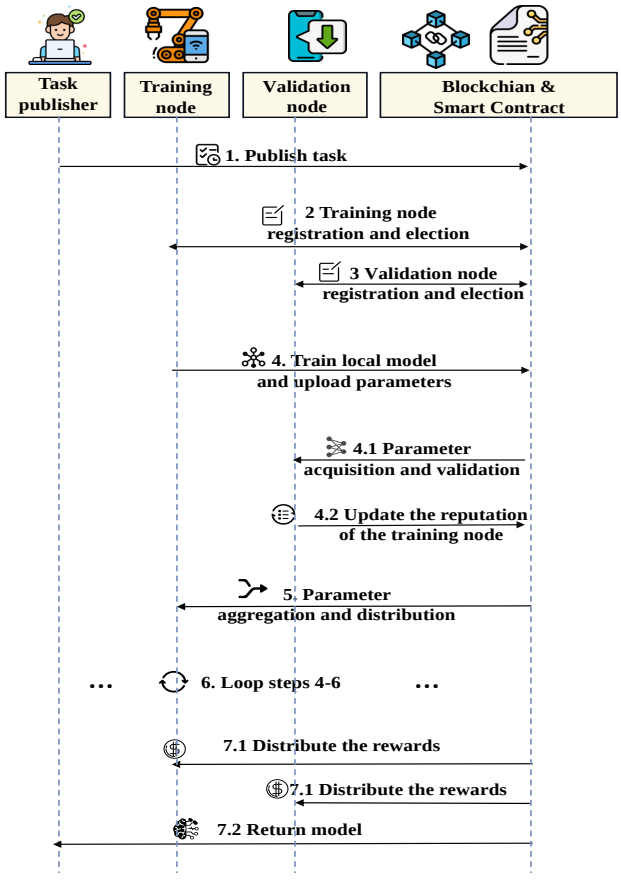

Figure 3: System Workflow Diagram

Step 1. Publish task.The task publisher T releases the requirements for the federated learning task (including the task objective G, data requirements D_r, model type M, and

total reward tokens R_{total}) onto the blockchain B, making this information accessible to all potential participants.

Step 2. Training node registration and election.Participants in the system choose to become either training nodes n_{ti} or validation nodes n_{vi} based on their resources and willingness, and proceed with registration. During the registration process, participants must stake a certain amount of tokens C, which will be locked by the smart contract SC and converted into an initial reputation value R_{init}, serving as proof of qualification for participating in the task.

Step 3. Validation node registration and election.The smart contract SC determines the final nodes N_t' and N_v' that will participate in the task, based on the participants' reputation values R_i and staked tokens C_i. Participants with higher reputation values R_i and more staked tokens C_i are given priority in obtaining participation rights.

Step 4. Local model training and verification.The selected training nodes $n_{ti} \in N_t'$ perform model training based on the global model parameters θ and the local datasets D_{ti}, and upload the trained model parameter updates θ_t to the blockchain network B. Subsequently, the validation nodes $n_{vi} \in N_v'$ retrieve these parameter updates and evaluate them using the validation dataset D_v specified by the task publisher. The evaluation scores are fed back to the smart contract, which are used to update the corresponding training nodes' reputation values R_i.

Step 5. Global model aggregation and distribution.Once the smart contract has collected a sufficient number of validated parameter updates, it executes parameter aggregation to generate new global model parameters $\theta = f(\{\theta_t\})$. The aggregated model parameters are then distributed to the training nodes, initiating a new round of training.

Step 6. Iterate the loop Steps.4-6 are repeated until the predetermined termination conditions C_{end} are met. These conditions may include a preset number of training rounds, a model accuracy threshold, etc.

Step 7. Return the results.The smart contract SC distributes rewards based on participants' contributions and reputation values R_i, and refunds any staked tokens C_i that were not deducted. After task completion, participants can receive corresponding token rewards W_i based on their performance in the task.

4.3. Reputation-Based Participant Selection Management Mechanism

The participant selection and reputation value management mechanism is a key component of the blockchain-based federated learning framework. This mechanism aims to ensure that high-quality participants are selected more often for tasks while effectively constraining the selection of low-quality or malicious participants. Algorithm 1 describes the participant selection process in detail.

First, the node selection mechanism is primarily based on participants' reputation values and the amount of staked tokens. The smart contract calculates participants' comprehensive scores S_i, ranks them, and selects top-ranking nodes to participate in the federated learning task.

Algorithm 1: Participant Selection and Reputation Management

Input: Set of all nodes N, staked tokens $\{C_i\}$, decay factor δ_d, weight coefficients $\alpha, \beta, \gamma, \delta_r$, number of training nodes k_t, number of validation nodes k_v

Output: Selected training nodes N_t', validation nodes N_v'

$C_{avg} = \frac{1}{|N|} \sum_{i \in N} C_i$;
for *each node* $i \in N$ **do**
  if *i is a new node* **then**
    $R_{init,i} = \frac{C_i}{C_{avg}} \cdot (1 + \log(1 + C_i))$;
    $R_i = R_{init,i}$;
  else
    $R_{i,t} = R_{i,t-1} \cdot \delta_d$;
  $\Delta R_i = R_{i,t-1} - R_{i,t}$;
  $S_i = \alpha \cdot R_i + \beta \cdot (\frac{C_i}{C_{avg}})^{\gamma} + \delta_r \cdot \Delta R_i$;
Sort nodes in descending order based on S_i;
$N_t' =$ Top k_t nodes from sorted list;
$N_v' =$ Next k_v nodes from sorted list;
return N_t', N_v';

For participants joining the system, their initial reputation value R_{init} is determined by the amount of staked tokens C. Staked tokens represent both the participant's sincerity and a benchmark for measuring initial reputation in the system. The formula for calculating the initial reputation value is as follows:

$$R_{init,i} = f(C_i) = \frac{C_i}{C_{avg}} \cdot \left(1 + \log(1 + C_i)\right) \tag{5}$$

where C_{avg} represents the average number of tokens staked by all participants. This design ensures that participants who stake more tokens can obtain higher initial reputation values, but the growth rate gradually slows as the staked amount increases, preventing single large stakes from having an excessive impact on initial reputation values.

To prevent high-reputation nodes from monopolizing system resources over extended periods, this study introduces a mechanism for reputation value decay over time. The dynamic adjustment formula for reputation values is as follows:

$$R_{i,t} = R_{i,t-1} \cdot \delta_d \tag{6}$$

where $R_{i,t}$ represents the reputation value of node n_i at time t, and δ_d is the decay factor, $0 < \delta_d < 1$. The value of the decay factor δ_d can be dynamically adjusted according to the system's activity level. For example, when there are many new nodes joining the system, the value of δ_d can be appropriately reduced to accelerate the decay rate of reputation values, thereby providing more opportunities for new nodes to participate in tasks.

Finally, the calculation of the comprehensive score S_i considers multiple factors, including reputation value R_i,

staked tokens C_i, and reputation value change ΔR_i. The formula for calculating the comprehensive score is as follows:

$$S_i = \alpha \cdot R_i + \beta \cdot \left(\frac{C_i}{C_{avg}} \right)^{\gamma} + \delta_r \cdot \Delta R_i \tag{7}$$

where α, β, γ, and δ_r are weight coefficients for different factors and can be adjusted according to system requirements. ΔR_i is the change in reputation value for node n_i, calculated as

$$\Delta R_i = R_{i,t-1} - R_{i,t} \tag{8}$$

After completing the comprehensive score calculation, the system ranks all participants in descending order of S_i and selects the top-ranking nodes as the final training nodes N_t' and validation nodes N_v'.

4.4. Model Update and Validation

This section elaborates on the node training process and validation mechanism in the blockchain-based federated learning framework with reputation token incentives. These mechanisms aim to ensure model quality and system security, including a comprehensive evaluation system and multiple anomaly detection methods to prevent potential malicious behavior from training nodes.

4.4.1. Local Model Update

Algorithm 2: Local Model Update

Input: Training node set N_t', local datasets $\{D_{ti}\}$, model type M, learning rate η, local update rounds E, reputation threshold R_{min}

Output: Updated model parameters $\{\theta_{ti}^{(t)}\}$

for *each* $n_{ti} \in N_t'$ **do**
  Preprocess local dataset D_{ti};
  Initialize $\theta_{ti,0}$ based on model type M;
  $t = 0$;
  while *not converged* **do**
    $t = t + 1$;
    Retrieve latest global model parameters $\theta_g^{(t-1)}$ from blockchain B;
    $\theta_{ti}^{(t-1)} = \theta_g^{(t-1)}$;
    $\theta_{ti}^{(t)} = \theta_{ti}^{(t-1)} - \eta \nabla L(\theta_{ti}^{(t-1)}, D_{ti})$;
    if $t \mod E == 0$ **then**
      $\Delta\theta_{ti}^{(t)} = \theta_{ti}^{(t)} - \theta_{ti}^{(t-E)}$;
      if $R_{ti} \geq R_{min}$ **then**
        Upload $\Delta\theta_{ti}^{(t)}$ to blockchain B;
      else
        Reject $\Delta\theta_{ti}^{(t)}$;
        Record node's behavior;
return $\{\theta_{ti}^{(t)}\}$;

When the smart contract SC determines the final set of training nodes N_t', training nodes $n_{ti} \in N_t'$ begin model training using their local data D_{ti}. Algorithm 2 describes the node training process, with the following specific steps:

1) Data Preparation and Model Initialization: Each training node n_{ti} prepares its local dataset D_{ti} and preprocesses the data. Based on the model type M provided by the task publisher T, training nodes initialize local model parameters $\theta_{ti,0}$.
2) Model Training and Parameter Update: At the beginning of each training round, training nodes n_{ti} first retrieve the latest global model parameters $\theta_g^{(t-1)}$ from the blockchain B and initialize their local model parameters $\theta_{ti}^{(t-1)}$ with $\theta_g^{(t-1)}$. Training nodes perform model training locally and update model parameters:

$$\theta_{ti}^{(t)} = \theta_{ti}^{(t-1)} - \eta \nabla L(\theta_{ti}^{(t-1)}, D_{ti}) \tag{9}$$

   where $\theta_{ti}^{(t)}$ represents the model parameters of training node n_{ti} in the t-th round, η is the learning rate, and L is the loss function.
3) Parameter Upload: After every E rounds of training (E is a predefined local update rounds), training nodes upload the updated model parameters $\theta_{ti}^{(t)}$ to the blockchain B. The uploaded parameters are in the form of parameter updates $\Delta\theta_{ti}^{(t)} = \theta_{ti}^{(t)} - \theta_{ti}^{(t-E)}$, which represent the difference between the current parameters and the parameters from E rounds ago. Before accepting the uploaded parameters, the smart contract SC checks the reputation value R_{ti} of the submitting node n_{ti}. If R_{ti} is lower than a predefined reputation threshold R_{min}, the smart contract rejects the submitted parameters and records the node's behavior. This reputation-based parameter acceptance mechanism can be described as follows:

$$\text{Accept } \Delta\theta_{ti}^{(t)} \text{ if } R_{ti} \geq R_{min}, \text{ else reject} \tag{10}$$

   where R_{min} is a system-defined reputation threshold that can be adjusted according to the security requirements and the overall reputation distribution of the network.

4.4.2. Validation and Reputation Management

Validation nodes play a crucial role in this framework, ensuring system security and reliability. This study designs a multiple validation mechanism and a non-linear reputation value evaluation model. The validation process and reputation update mechanism are summarized in Algorithm 3 and Algorithm 4, respectively. The validation process is as follows:

1) Validation Node Assignment: Randomly select k validation nodes to form a validation group $V = \{v_1, v_2, ..., v_k\}$.
2) Independent Validation: Each validation node $v_i \in V$ independently executes the following steps:
   a) Retrieve model parameters $\{\theta_1^{(t)}, \theta_2^{(t)}, ..., \theta_n^{(t)}\}$ submitted by all training nodes from the blockchain B.
   b) Calculate the accuracy of each model using the validation dataset D_v:

Algorithm 3: Validation Process

Input: Validation group $V = \{v_1, ..., v_k\}$, training nodes' parameters $\{\theta_j^{(t)}\}$, validation dataset D_v, weights w_1, w_2, score threshold τ
Output: Final scores $\{S_j\}$ for each training node, Malicious Behavior Indicators $\{\mathcal{M}_j^{(t)}\}$

Initialize $\{\mathcal{M}_j^{(t)}\} = 0$ for all training nodes;
for *each* $v_i \in V$ **do**
  Retrieve $\{\theta_j^{(t)}\}$ from blockchain B;
  for *each training node* j **do**
    $A_j^i = \text{Accuracy}(\theta_j^{(t)}, D_v)$;
    $Z_j^i = \frac{A_j^i - \mu_i}{\sigma_i}$;
    $s_j^i = 2^{-\frac{E(h(\theta_j^{(t)}))}{c(n)}}$;
    $S_j^i = w_1 Z_j^i + w_2 s_j^i$;
for *each training node* j **do**
  $S_j^{\text{median}}, \sigma_{S_j} = \text{MedianAndStdDev}(\{S_j^i\})$;
  $S_j^{\text{valid}} = \{S_j^i : |S_j^i - S_j^{\text{median}}| \le 2\sigma_{S_j}\}$;
  if $|S_j^{valid}| \ge \lceil 2k/3 \rceil$ **then**
    $S_j = \text{Average}(S_j^{\text{valid}})$;
    if $S_j < \tau$ **then**
      $\mathcal{M}_j^{(t)} = 1$;
  else
    $\mathcal{M}_j^{(t)} = 0$;
    Repeat validation process;
return $\{S_j\}, \{\mathcal{M}_j^{(t)}\}$;

$$A_j^i = \text{Accuracy}(\theta_j^{(t)}, D_v), \quad j = 1, 2, ..., n \tag{11}$$

c) Perform Z-Score standardization on accuracies:

$$Z_j^i = \frac{A_j^i - \mu_i}{\sigma_i} \tag{12}$$

where μ_i and σ_i are the mean and standard deviation of all accuracies calculated by validation node v_i, respectively.

d) Execute isolation forest anomaly detection and calculate anomaly scores:

$$s_j^i = s(\theta_j^{(t)}, \mathcal{F}_t) = 2^{-\frac{E(h(\theta_j^{(t)}))}{c(n)}} \tag{13}$$

where $h(\theta_j^{(t)})$ represents the average path length of parameters $\theta_j^{(t)}$ in the isolation forest $\mathcal{F}_t$, and $c(n)$ is a constant related to sample size n.

e) Calculate comprehensive scores:

$$S_j^i = w_1 Z_j^i + w_2 s_j^i \tag{14}$$

where w_1 and w_2 are weight coefficients.

3) Consistency Check: The smart contract SC calculates the median score S_j^{median} and score standard deviation σ_{S_j} for each training node j. If $|S_j^i - S_j^{\text{median}}| > 2\sigma_{S_j}$, the score is discarded.
4) Consensus Formation: If the number of remaining valid scores is $\ge \lceil 2k/3 \rceil$, the average of these scores is taken as the final score S_j for training node j. Otherwise, the validation round fails, validation nodes are reassigned, and the validation process is repeated.
5) Validation Node Reputation Update: Based on the proximity of provided scores to the final scores, update the reputation value of validation nodes:

$$R_{vi}^{(t+1)} = R_{vi}^{(t)} + \lambda_v \left(1 - \frac{1}{n}\sum_{j=1}^{n} \frac{|S_j^i - S_j|}{S_j}\right) \tag{15}$$

where λ_v is an adjustment factor.

To evaluate the reputation value of training nodes, this study proposes a simplified non-linear reputation value evaluation model. The reputation value $R_j^{(t+1)}$ of training node j after the t-th round of training is updated as follows:

$$R_j^{(t+1)} = R_j^{(t)} + \Delta R_j^{(t)} \tag{16}$$

where $\Delta R_j^{(t)}$ is the change in reputation value, defined as:

$$\Delta R_j^{(t)} = \lambda_h (1 - S_j) - \lambda_m (e^{\kappa N_j^{(t)}} - 1) S_j \tag{17}$$

where $\lambda_h > 0$ controls the growth rate of honest nodes' reputation values, $\lambda_m > 0$ controls the base rate of decline for malicious nodes' reputation values, and $\kappa > 0$ controls the intensity of the cumulative effect of malicious behavior. $N_j^{(t)}$ is the cumulative number of malicious behaviors by node j up to round t, recursively defined as:

$$N_j^{(t)} = N_j^{(t-1)} + \mathcal{M}_j^{(t)} \tag{18}$$

where $\mathcal{M}_j^{(t)}$ is the malicious behavior indicator for node j in round t, taking a value of 0 or 1. The initial value is $N_j^{(0)} = 0$.

This model aims to allow honest nodes' reputations to gradually increase, while malicious nodes' reputations decrease exponentially with the number of malicious acts, thereby incentivizing honest behavior and punishing persistent malicious behavior.

4.5. Token Reward Distribution and Penalty Mechanism

This section introduces the token reward distribution and penalty mechanism in the blockchain-based federated learning framework with reputation token incentives. This mechanism, through multi-dimensional assessment, aims to

Algorithm 4: Reputation Update Mechanism

Input: Current reputations $\{R_j^{(t)}\}, \{R_{vi}^{(t)}\}$, scores $\{S_j\}, \{S_j^i\}$, malicious behavior indicators $\{\mathcal{M}_j^{(t)}\}$, parameters $\lambda_h, \lambda_m, \kappa, \lambda_v$

Output: Updated reputations $\{R_j^{(t+1)}\}, \{R_{vi}^{(t+1)}\}$

for *each training node* j **do**
- $N_j^{(t)} = N_j^{(t-1)} + \mathcal{M}_j^{(t)}$;
- $\Delta R_j^{(t)} = \lambda_h(1 - S_j) - \lambda_m(e^{\kappa N_j^{(t)}} - 1)S_j$;
- $R_j^{(t+1)} = R_j^{(t)} + \Delta R_j^{(t)}$;

for *each validation node* v_i **do**
- $R_{vi}^{(t+1)} = R_{vi}^{(t)} + \lambda_v \left(1 - \frac{1}{n}\sum_{j=1}^{n} \frac{|S_j^i - S_j|}{S_j}\right)$;

return $\{R_j^{(t+1)}\}, \{R_{vi}^{(t+1)}\}$;

incentivize participants to provide high-quality contributions while suppressing potential malicious behavior. The validation process and the deposit and penalty mechanism are summarized in Algorithm 5 and Algorithm 6, respectively.

At the beginning of the task, the publisher sets the total reward token amount R_{total}, and specifies the reward ratio λ_t for training nodes and λ_v for validation nodes, where $\lambda_t, \lambda_v \in (0, 1)$ and $\lambda_t + \lambda_v = 1$. Based on this, the training node reward pool is defined as $R_{train} = \lambda_t \cdot R_{total}$ and the validation node reward pool as $R_{verify} = \lambda_v \cdot R_{total}$. Deposits confiscated due to the penalty mechanism will be redistributed to the corresponding reward pools according to the ratio of λ_t and λ_v.

The reward calculation for training nodes is based on their contribution to the model and reputation value:

$$W_{ti} = R_{train} \cdot \frac{C_{ti} \cdot R_{ti}}{\sum_j (C_{tj} \cdot R_{tj})} \tag{19}$$

where R_{ti} reflects the node's historical performance, and C_{ti} represents the comprehensive contribution of node n_{ti}:

$$C_{ti} = \frac{1}{T}\sum_{t=1}^{T} \frac{\|\theta_{ti}^{(t)} - \theta_{ti}^{(t-1)}\|_2}{\|\theta_g^{(t)} - \theta_g^{(t-1)}\|_2} \tag{20}$$

Where, T is the total number of training rounds, $\theta_{ti}^{(t)}$ is the model parameters of node n_{ti} in round t, and $\theta_g^{(t)}$ is the global model parameters in round t.

The reward for validation nodes is similarly based on their accuracy and reputation value:

$$W_{vi} = R_{verify} \cdot \frac{A_{vi} \cdot R_{vi}}{\sum_j (A_{vj} \cdot R_{vj})} \tag{21}$$

Validation accuracy A_{vi} is calculated by comparing the evaluation results of the validation node with the consistency of other nodes:

$$A_{vi} = 1 - \frac{1}{|N_t'|}\sum_{j \in N_t'} |\mathrm{S}_{vi}(n_{tj}) - \overline{\mathrm{S}}(n_{tj})| \tag{22}$$

Here, $\mathrm{S}_{vi}(n_{tj})$ is the score given by validation node n_{vi} to training node n_{tj}, and $\overline{\mathrm{S}}(n_{tj})$ is the average score from all validation nodes.

This framework introduces a deposit mechanism. Participating nodes need to stake a certain amount of tokens as a deposit, the quantity of which is determined based on the task scale and the node's historical behavior:

$$C_i = \max(C_{min}, \mu \cdot R_{total} \cdot (1 - R_i)) \tag{23}$$

Here, C_{min} ensures a minimum deposit requirement, while μ acts as a deposit coefficient to adjust the deposit amount. This design allows high-reputation nodes to enjoy lower deposit requirements, while low-reputation nodes need to bear higher risks.

When malicious behavior is detected, the system will confiscate part or all of the deposit. The confiscation ratio ρ is dynamically adjusted according to the severity of the behavior:

$$\rho = \min(1, \rho_{base} + \nu \cdot S_i) \tag{24}$$

where ρ_{base} is the base confiscation ratio, ν is the adjustment coefficient, and S_i is the severity score (ranging from 0 to 1).

Algorithm 5: Token Reward Distribution

Input: Total reward R_{total}, reward ratios λ_t, λ_v, training nodes N_t', validation nodes N_v', model parameters $\{\theta_{ti}^{(t)}\}$, global parameters $\{\theta_g^{(t)}\}$, node reputations $\{R_{ti}\}, \{R_{vi}\}$, validation scores $\{\mathrm{S}_{vi}(n_{tj})\}$

Output: Node rewards $\{W_{ti}\}, \{W_{vi}\}$

$R_{train} = \lambda_t \cdot R_{total}$; $R_{verify} = \lambda_v \cdot R_{total}$;

for *each* $n_{ti} \in N_t'$ **do**
- $C_{ti} = \frac{1}{T}\sum_{t=1}^{T} \frac{\|\theta_{ti}^{(t)} - \theta_{ti}^{(t-1)}\|_2}{\|\theta_g^{(t)} - \theta_g^{(t-1)}\|_2}$;
- $W_{ti} = R_{train} \cdot \frac{C_{ti} \cdot R_{ti}}{\sum_j (C_{tj} \cdot R_{tj})}$;

for *each* $n_{vi} \in N_v'$ **do**
- $\overline{\mathrm{S}}(n_{tj}) = \frac{1}{|N_v'|}\sum_{k \in N_v'} \mathrm{S}_{vk}(n_{tj})$;
- $A_{vi} = 1 - \frac{1}{|N_t'|}\sum_{j \in N_t'} |\mathrm{S}_{vi}(n_{tj}) - \overline{\mathrm{S}}(n_{tj})|$;
- $W_{vi} = R_{verify} \cdot \frac{A_{vi} \cdot R_{vi}}{\sum_j (A_{vj} \cdot R_{vj})}$;

return $\{W_{ti}\}, \{W_{vi}\}$;

5. Performance Analysis

This section provides a detailed description of the experimental evaluation process for our proposed reputation-based federated learning scheme. We designed a series of

Algorithm 6: Deposit and Penalty Mechanism

Input: Total reward R_{total}, minimum deposit C_{min}, deposit coefficient μ, node reputations $\{R_i\}$, base confiscation ratio ρ_{base}, adjustment coefficient ν, severity scores $\{S_i\}$

Output: Required deposits $\{C_i\}$, confiscation ratios $\{\rho_i\}$

for *each node* $i \in N'_t \cup N'_v$ **do**
  $C_i = \max(C_{min}, \mu \cdot R_{total} \cdot (1 - R_i))$;
for *each detected malicious node* i **do**
  $\rho_i = \min(1, \rho_{base} + \nu \cdot S_i)$;
  Confiscate $\rho_i \cdot C_i$ tokens;
  Redistribute confiscated tokens to reward pools;
return $\{C_i\}, \{\rho_i\}$;

experiments to assess the framework's performance in terms of model performance, malicious node identification rate, and changes in node reputation.

5.1. Experimental Environment and Configuration

The hardware environment used for the experiments includes a workstation equipped with an Intel i7-10700K CPU, 32GB RAM, and an NVIDIA RTX 3080 GPU, running Ubuntu 20.04 LTS as the operating system. The programming environment for the experiments is Python 3.8, primarily utilizing the PyTorch framework for model training and testing. The blockchain component simulation was implemented using Hyperchain 2.12.0[44], managed with Docker containers to simulate the distributed ledger operating environment. Table 2 lists the configuration of various parameters in our scheme as well as the experimental environment setup. We conducted experiments on the MNIST and Fashion-MNIST (FMNIST) datasets, both comprising 60,000 training and 10,000 test samples. Each image is a 28×28 grayscale representation of a digit (MNIST) or a fashion item (FMNIST). Prior to training, all pixel values were normalized to the range [0,1] without additional data augmentation. We randomly and uniformly distributed the training data across 100 training clients, ensuring that each client received an equal portion of the training set. An additional 50 validation clients were designated to evaluate the submitted model parameters.For the model, we adopted a standard Convolutional Neural Network (CNN) that includes two convolutional-pooling layers followed by two fully connected layers, a common lightweight architecture suitable for MNIST-like datasets. The training process used the Adam optimizer with an initial learning rate of 0.01 and a batch size of 32. We ran a total of 250 training epochs, where in each global communication round, clients performed local updates on their assigned data before uploading model parameters for aggregation.

5.2. Comparison Schemes

To comprehensively evaluate the performance of our proposed scheme, we compared it with the following four approaches:

1) Privacy-Preserving Serverless FL for IoT (PSFL-IoT)[34]: This scheme proposes a serverless federated learning framework for IoT environments based on secure multi-party computation. The approach eliminates central server dependencies, achieves fault tolerance through secret sharing, and provides formal security guarantees against collusion attacks while maintaining accuracy without additional loss.
2) Personalized Federated Learning with Client Reputation (PFL-CR)[40]: This scheme employs a reputation-based approach for traffic flow prediction using Graph Neural Networks and Gated Recurrent Units. It incorporates client reputation mechanisms to defend against malicious attacks while optimizing personalized model training through reputation-guided aggregation strategies.
3) Adaptive Differential Privacy Federated Learning (ADPFL)[30]: This scheme protects data privacy by introducing noise to the uploaded model parameters, adaptively adjusting noise levels to balance model performance and privacy protection.
4) Federated Learning in IoT with Malicious Nodes (FedIoT-Malicious): This baseline represents standard federated learning implementations in IoT environments following the FedIoTs framework[34]. We introduce malicious participants in this setting to evaluate system robustness under adversarial conditions, providing a representative baseline for assessing the effectiveness of security-enhanced federated learning approaches.

Regarding attack scenarios, we considered two main types: data pollution attacks and label flipping attacks. For data pollution attacks, we set malicious client proportions at 10%, 20%, and 30%. These malicious clients provide completely polluted data. For label flipping attacks, we set malicious client proportions at 10%, 15%, and 20%, where malicious nodes intentionally mislabel part of the data to mislead the model.

5.3. Experimental Results and Analysis

5.3.1. Model Performance Comparison

First, we compared the global model performance of our proposed framework with baseline schemes in the absence of malicious nodes. Table 3 shows the accuracy of each scheme on MNIST and FMNIST datasets.

As shown in Table 3, the PSFL-IoT scheme achieves 92.8% accuracy on the MNIST dataset and 78.9% on the FMNIST dataset, demonstrating the effectiveness of serverless architecture in maintaining model performance. PFL-CR, which employs reputation-based personalized federated learning, achieved 91.6% and 77.8% accuracy on MNIST and FMNIST respectively, showing competitive performance with its personalized approach. ADPFL, incorporating differential privacy, demonstrated the lowest accuracy at

Table 2
Experimental Environment and Key Parameter Configurations

Parameter	Configuration/Value
Hardware Environment	Intel i7-10700K CPU, 32GB RAM, NVIDIA RTX 3080 GPU
Operating System	Ubuntu 20.04 LTS
Programming Environment	Python 3.8, PyTorch
Blockchain Platform	Hyperchain 2.12.0
Blockchain Consensus Mechanism	RBFT
Dataset	MNIST, Fashion-MNIST (60,000 train / 10,000 test)
Image Size	28×28 grayscale
Number of Training Clients	100
Number of Validation Clients	50
Model Architecture	2 convolution-pooling layers + 2 fully connected layers
Optimizer	Adam
Initial Learning Rate	0.01
Batch Size	32
Total Training Epochs	250
Loss Function	Cross-Entropy
Reputation Update Params ($\lambda_v, \lambda_h, \lambda_m, \kappa, R_{min}$)	(0.05, 0.05, 0.1, 0.15, 3)
Scoring Weights (w_1, w_2)	(0.6, 0.4)

Table 3
Model Accuracy of Different Schemes Without Malicious Nodes

Scheme	MNIST	FMNIST
PSFL-IoT[34]	92.8%	78.9%
ADPFL[30]	90.7%	76.2%
PFL-CR[40]	91.6%	77.8%
Proposed Scheme	92.4%	78.3%

90.7% and 76.2% for MNIST and FMNIST, respectively, due to the noise introduced to preserve privacy. In contrast, our proposed scheme achieved 92.4% and 78.3% accuracy for MNIST and FMNIST datasets, respectively, indicating minimal compromise in model performance while incorporating additional security mechanisms. These results confirm that our framework's reputation-based selection and anomaly detection mechanisms do not adversely affect performance in benign environments.

However, when malicious nodes participate and submit harmful model updates, the performance of each scheme shows significant differences. Figure 4 and Figure5 show the experimental results of each scheme under data pollution attack scenarios on MNIST and FMNIST datasets, respectively.

In data pollution attacks, malicious nodes inject corrupted or completely randomized data to degrade the global model's performance. The experimental results reveal distinct vulnerability patterns across different schemes. On the MNIST dataset, FedIoT-Malicious experiences performance degradation under data pollution attacks, with accuracy dropping from 92.8% to 88%, 75%, and 66.9% under 10%, 20%, and 30% malicious node proportions respectively—representing performance losses of 5.2%, 19.2%, and 27.9%. The serverless architecture, while providing privacy guarantees, lacks specialized mechanisms for identifying malicious participants. PFL-CR demonstrates improved resilience, achieving 89.3%, 76.8%, and 68.5% accuracy under the same attack intensities, effectively utilizing its reputation mechanisms to defend against malicious participants.

ADPFL exhibits mixed performance under data pollution attacks. While it achieves 85.2% accuracy with 10% malicious nodes, it performs better than FedIoT-Malicious at 20% malicious nodes (79% vs 75%), likely due to the noise injection masking some attack effects. However, at 30% malicious nodes, ADPFL suffers the most severe degradation (58.9%), as the combination of attack effects and differential privacy noise compounds the performance deterioration. In contrast, our proposed scheme maintains stability with accuracies of 92.1%, 90.8%, and 88.7%, corresponding to minimal degradations of only 0.8%, 2.1%, and 4.0% respectively.

The FMNIST dataset presents more challenging attack scenarios due to its higher visual complexity compared to MNIST. FedIoT-Malicious shows more significant accuracy drops from 78.9% to 73.7%, 66%, and 62.5% (degradations of 6.6%, 16.4%, and 20.8%), indicating that serverless architectures face greater challenges in complex data environments. PFL-CR achieves better resilience with accuracies of 74.9%, 67.4%, and 64.1% (degradations of 3.7%, 13.4%,

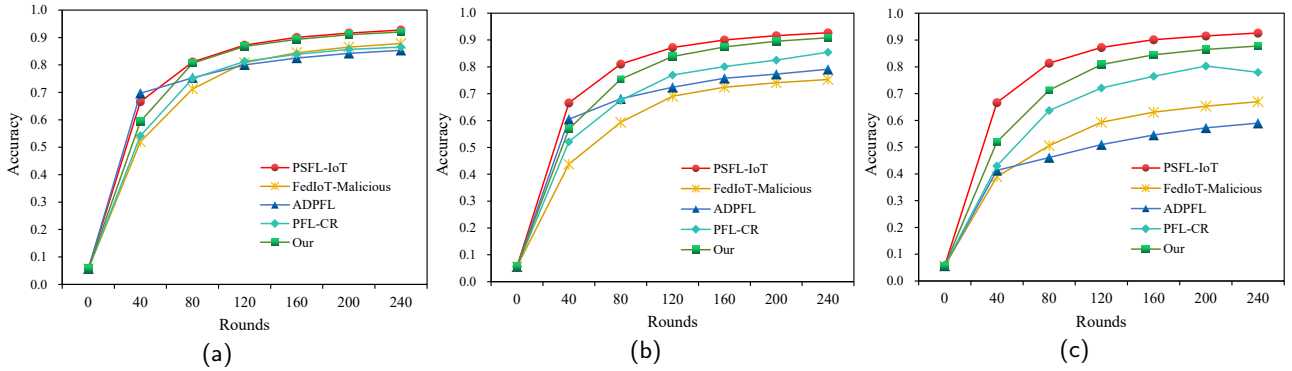

Figure 4: Comparison of Model Performance Under Data Pollution Attacks on MNIST Dataset.
(a)Malicious node proportion: 10% .(b) Malicious node proportion: 20%.(c)Malicious node proportion: 30%

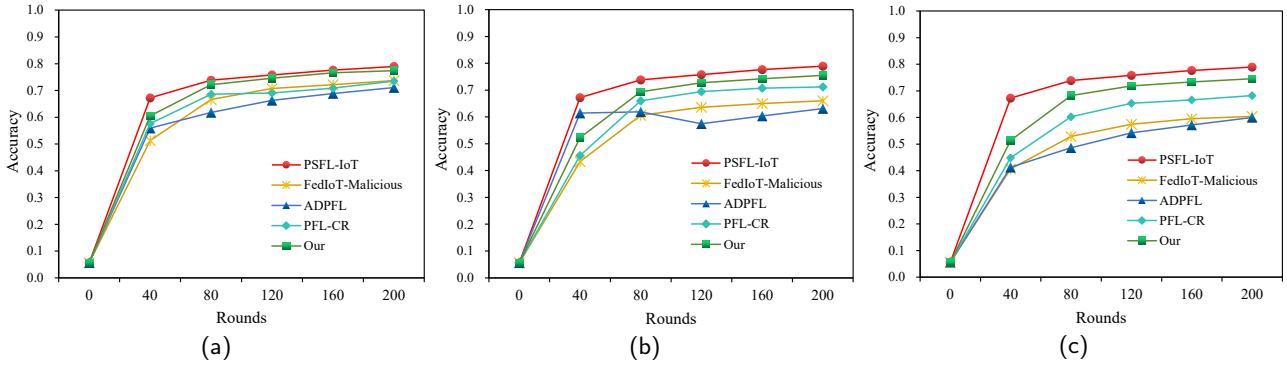

Figure 5: Comparison of Model Performance Under Data Pollution Attacks on FMNIST Dataset.
(a)Malicious node proportion: 10% .(b) Malicious node proportion: 20%.(c)Malicious node proportion: 30%

and 17.6%), confirming that its malicious attack defense capabilities provide consistent protection across different data complexities. ADPFL shows degradations of 6.7%, 17.2%, and 21.3% respectively, with performance becoming increasingly unstable as attack intensity grows. Our proposed scheme maintains superior stability with accuracies of 77.4%, 75.5%, and 74.7% (degradations of only 1.1%, 3.6%, and 4.6%), highlighting the robustness of security-oriented reputation design.

In label flipping attacks, malicious nodes intentionally mislabel part of the data to mislead the model. Figure 6 and Figure 7 show the experimental results under label flipping attack scenarios on MNIST and FMNIST datasets, respectively. On the MNIST dataset, FedIoT-Malicious achieves test accuracies of 82.4%, 76.7%, and 66.2% under 10%, 15%, and 20% malicious node proportions. The serverless architecture's distributed nature makes it challenging to detect coordinated label manipulation attacks across multiple nodes. PFL-CR performs better at 84.1%, 78.5%, and 68.9% respectively, leveraging its reputation mechanism to identify and mitigate the impact of malicious clients, though its approach primarily targets traffic flow prediction scenarios. ADPFL shows the poorest performance at 72.3%, 68.6%, and 60.1%, as the added noise amplifies the confusion caused by incorrect labels. Our scheme maintains test accuracies of 91.8%, 90.4%, and 88.6% in these three scenarios, benefiting from its specialized anomaly detection and security-focused reputation updates.

On the FMNIST dataset, the impact of label flipping attacks becomes more pronounced due to the increased complexity of fashion item classification. FedIoT-Malicious achieves test accuracies of 70.5%, 66.7%, and 58.3% under 10%, 15%, and 20% malicious node proportions, showing larger performance degradation compared to MNIST. PFL-CR performs slightly better at 72.1%, 68.9%, and 60.8% respectively, demonstrating that its malicious attack defense mechanisms provide improved resilience across different data complexities. ADPFL maintains similar performance patterns at 72.3%, 68.6%, and 60.1%, with the differential privacy mechanism showing consistent but limited effectiveness against label manipulation. Our scheme maintains test accuracies of 77.0%, 75.9%, and 73.2% in these scenarios, demonstrating that security-oriented design principles become increasingly important as data complexity grows.

5.3.2. Malicious Node Identification Rate

Beyond model performance evaluation, we assess the framework's capability to identify and isolate malicious participants. Our detection mechanism operates by monitoring reputation score changes and anomaly patterns in model updates. A node is flagged as malicious when its

Table 4
Malicious Node Detection Rate Comparison

Attack Type	Malicious	Proposed Scheme	PFL-CR
Data Poisoning	10%	98.5%	85.3%
	20%	97.8%	78.9%
	30%	96.8%	72.4%
Label Flipping	10%	96.2%	76.8%
	15%	95.4%	71.5%
	20%	94.1%	68.2%

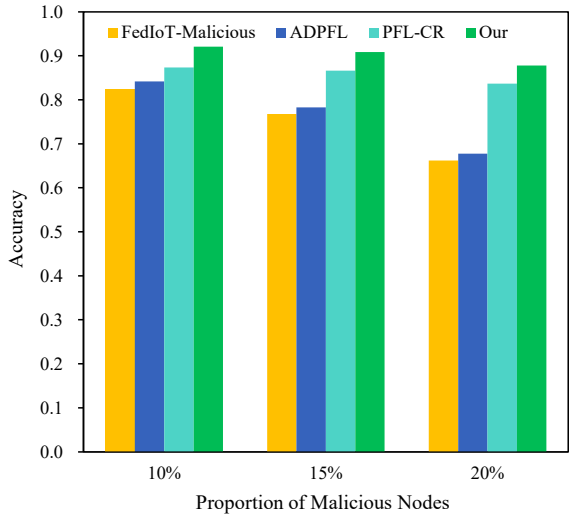

Figure 6: Comparison of Model Performance Under Label Flipping Attacks on MNIST Dataset

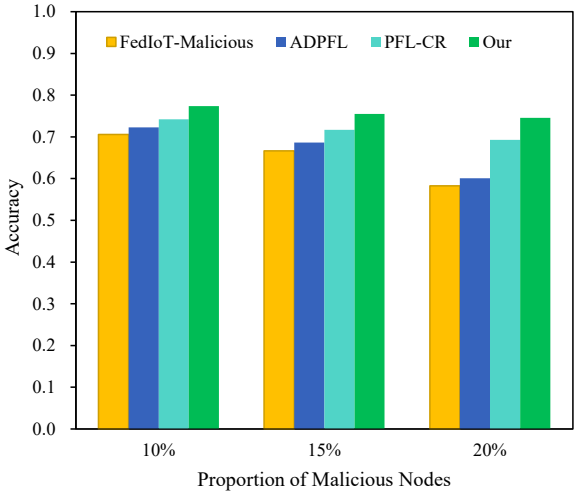

Figure 7: Comparison of Model Performance Under Label Flipping Attacks on FMNIST Dataset

reputation falls below a threshold of 3.0 or exhibits consistent anomalous behavior patterns over multiple rounds. For comparison, PFL-CR identifies dishonest clients based on their credibility evaluation and removes clients whose data quality significantly impacts the personalized model performance. Table 4 presents the detection performance comparison between the two reputation-based schemes.

As shown in Table 4, our proposed framework achieves detection rates above 94% across all attack scenarios, while PFL-CR achieves detection rates ranging from 68-85%. Both schemes utilize reputation mechanisms to identify and handle problematic participants, but differ in their detection sophistication and security focus.

Our detection mechanism combines reputation-based evaluation with anomaly detection and blockchain-based tracking. When malicious participants repeatedly submit harmful updates, their reputation values decline, leading to their eventual removal from training rounds. The multi-layered approach enables robust identification of various attack types including coordinated data poisoning and label flipping scenarios. PFL-CR also employs reputation-based client evaluation and can successfully identify dishonest participants, but our framework incorporates additional security-oriented features such as anomaly pattern analysis and blockchain-based tamper-resistant tracking, enabling more comprehensive threat detection in security-critical environments.

5.3.3. Reputation Value Change Analysis

To analyze the dynamic changes of node reputation values, we recorded the reputation value change curves of typical honest nodes and malicious nodes during the training process. In our experiment, we set the initial reputation value of all nodes to 10 for better comparison. Figure 8 shows the reputation value changes of honest and malicious nodes under 20% data pollution attacks in the MNIST experiment.

Figure 8 clearly shows that the reputation values of honest node gradually increase with the number of training rounds. In contrast, the reputation values of malicious node decrease rapidly with the number of malicious actions until reaching zero. These results demonstrate that our scheme's reputation mechanism can effectively distinguish between honest and malicious nodes, further enhancing system security and stability through dynamic adjustments of reputation values.

6. Conclusion

We presented BRITE-FL, a blockchain-based secure federated learning framework that addresses IoT device heterogeneity through role-based participation and mitigates

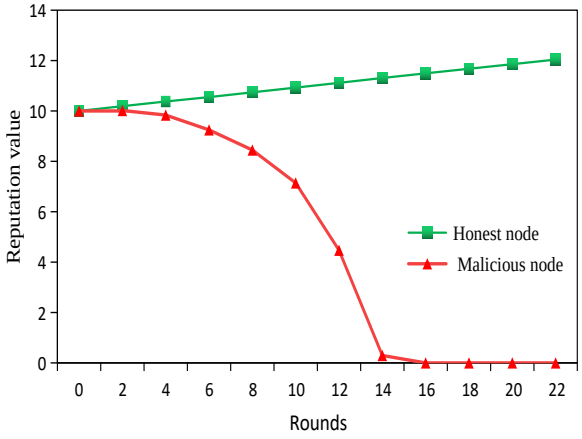

Figure 8: Reputation value change curves of typical honest nodes and malicious nodes

malicious attacks via reputation token incentives. Simultaneously, the framework utilizes blockchain's decentralized characteristics and smart contract technology to achieve dynamic management and reputation assessment of node behavior, identifying and suppressing malicious nodes while incentivizing honest nodes to contribute continuously. Experimental results demonstrate that when facing data pollution and label flipping attacks, even with malicious node proportions as high as 30%, the framework can limit model performance degradation to within 5% and achieve over 90% malicious node identification rate, exhibiting significant attack resistance. This research provides a new technical approach to enhance the security and reliability of federated learning systems and lays a foundation for constructing robust distributed machine learning systems in complex IoT environments. Future work will focus on optimizing the framework's computational efficiency and scalability, exploring the introduction of privacy protection mechanisms, and extending the framework to other distributed learning scenarios.

References

[1] Juncen Zhu, Jiannong Cao, Divya Saxena, Shan Jiang, and Houda Ferradi. Blockchain-empowered federated learning: Challenges, solutions, and future directions. *ACM Computing Surveys*, 55(11):1–31, 2023.

[2] Yang Zhao, Jun Zhao, Mengmeng Yang, Teng Wang, Ning Wang, Lingjuan Lyu, Dusit Niyato, and Kwok-Yan Lam. Local differential privacy-based federated learning for internet of things. *IEEE Internet of Things Journal*, 8(11):8836–8853, 2020.

[3] Mansoor Ali, Hadis Karimipour, and Muhammad Tariq. Integration of blockchain and federated learning for internet of things: Recent advances and future challenges. *Computers & Security*, 108:102355, 2021.

[4] Yuzheng Li, Chuan Chen, Nan Liu, Huawei Huang, Zibin Zheng, and Qiang Yan. A blockchain-based decentralized federated learning framework with committee consensus. *IEEE Network*, 35(1):234–241, 2020.

[5] Dapeng Man, Fanyi Zeng, Wu Yang, Miao Yu, Jiguang Lv, and Yijing Wang. Intelligent intrusion detection based on federated learning for edge-assisted internet of things. *Security and Communication Networks*, 2021(1):9361348, 2021.

[6] Weishan Zhang, Qinghua Lu, Qiuyu Yu, Zhaotong Li, Yue Liu, Sin Kit Lo, Shiping Chen, Xiwei Xu, and Liming Zhu. Blockchain-based federated learning for device failure detection in industrial iot. *IEEE Internet of Things Journal*, 8(7):5926–5937, 2020.

[7] Jiaxiang Zhang, Yiming Liu, Xiaoqi Qin, Xiaodong Xu, and Ping Zhang. Adaptive resource allocation for blockchain-based federated learning in internet of things. *IEEE Internet of Things Journal*, 10 (12):10621–10635, 2023.

[8] Liang Gao, Li Li, Yingwen Chen, ChengZhong Xu, and Ming Xu. Fgfl: A blockchain-based fair incentive governor for federated learning. *Journal of Parallel and Distributed Computing*, 163:283–299, 2022.

[9] Unal Tatar, Yasir Gokce, and Brian Nussbaum. Law versus technology: Blockchain, gdpr, and tough tradeoffs. *Computer Law & Security Review*, 38:105454, 2020.

[10] H. Brendan Mcmahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. 2016.

[11] Stacey Truex, Ling Liu, Ka-Ho Chow, Mehmet Emre Gursoy, and Wenqi Wei. Ldp-fed: Federated learning with local differential privacy. In *Proceedings of the third ACM international workshop on edge systems, analytics and networking*, pages 61–66, 2020.

[12] Gaoyang Liu, Chen Wang, Xiaoqiang Ma, and Yang Yang. Keep your data locally: Federated-learning-based data privacy preservation in edge computing. *IEEE Network*, 35(2):60–66, 2021.

[13] Jingcheng Song, Weizheng Wang, Thippa Reddy Gadekallu, Jianyu Cao, and Yining Liu. Eppda: An efficient privacy-preserving data aggregation federated learning scheme. *IEEE Transactions on Network Science and Engineering*, 10(5):3047–3057, 2022.

[14] Zhe Yang, Kan Zheng, Kan Yang, and Victor C. M. Leung. A blockchain-based reputation system for data credibility assessment in vehicular networks. *IEEE*, 2017.

[15] Qipeng Xie, Siyang Jiang, Linshan Jiang, Yongzhi Huang, Zhihe Zhao, Salabat Khan, Wangchen Dai, Zhe Liu, and Kaishun Wu. Efficiency optimization techniques in privacy-preserving federated learning with homomorphic encryption: A brief survey. *IEEE Internet of Things Journal*, 11(14):24569–24580, 2024.

[16] Xin Wang, Haoji Zhang, Haoyu Wu, and Hongnian Yu. Dual-blockchain based multi-layer grouping federated learning scheme for heterogeneous data in industrial iot. *Blockchain: Research and Applications*, page 100195, 2024.

[17] Chunghun Baek, Sungwook Kim, Dongkyun Nam, and Jihoon Park. Enhancing differential privacy for federated learning at scale. *IEEE Access*, 9:148090–148103, 2021.

[18] Md Ashraf Uddin, Andrew Stranieri, Iqbal Gondal, and Venki Balasubramanian. A survey on the adoption of blockchain in iot: Challenges and solutions. *Blockchain: Research and Applications*, 2(2):100006, 2021.

[19] Noora Mohammed Al-Maslamani, Mohamed Abdallah, and Bekir Sait Ciftler. Reputation-aware multi-agent drl for secure hierarchical federated learning in iot. *IEEE Open Journal of the Communications Society*, 4:1274–1284, 2023.

[20] Weizhao Jin, Yuhang Yao, Shanshan Han, Carlee Joe-Wong, Srivatsan Ravi, Salman Avestimehr, and Chaoyang He. Fedml-he: An efficient homomorphic-encryption-based privacy-preserving federated learning system. *arXiv preprint arXiv:2303.10837*, 2023.

[21] Mohd Javaid, Abid Haleem, Ravi Pratap Singh, Shahbaz Khan, and Rajiv Suman. Blockchain technology applications for industry 4.0: A literature-based review. *Blockchain: Research and Applications*, 2 (4):100027, 2021.

[22] Yunhua He, Mingshun Luo, Bin Wu, Limin Sun, Yongdong Wu, Zhiquan Liu, and Ke Xiao. A game theory-based incentive mechanism for collaborative security of federated learning in energy blockchain environment. *IEEE Internet of Things Journal*, 10(24):21294–21308, 2023.

[23] Huaqun Guo and Xingjie Yu. A survey on blockchain technology and its security. *Blockchain: research and applications*, 3(2):100067, 2022.

[24] Zhilin Wang, Qin Hu, and Xu Zehui Xiong. Incentive mechanism design for joint resource allocation in blockchain-based federated learning. *IEEE Transactions on Parallel and Distributed Systems: A Publication of the IEEE Computer Society*, 34(5):1536–1547, 2023.

[25] Umer Majeed, Latif U Khan, Sheikh Salman Hassan, Zhu Han, and Choong Seon Hong. Fl-incentivizer: Fl-nft and fl-tokens for federated learning model trading and training. *IEEE Access*, 11:4381–4399, 2023.

[26] Xuhan Zuo, Minghao Wang, Tianqing Zhu, Lefeng Zhang, Dayong Ye, Shui Yu, and Wanlei Zhou. Federated trustchain: Blockchain-enhanced llm training and unlearning. *arXiv preprint arXiv:2406.04076*, 2024.

[27] Yajie Zhou, Xiaoyi Pang, Zhibo Wang, Jiahui Hu, Peng Sun, and Kui Ren. Towards efficient asynchronous federated learning in heterogeneous edge environments. In *IEEE INFOCOM 2024-IEEE conference on computer communications*, pages 2448–2457. IEEE, 2024.

[28] Yunlong Lu, Xiaohong Huang, Yueyue Dai, Sabita Maharjan, and Yan Zhang. Blockchain and federated learning for privacy-preserved data sharing in industrial iot. *IEEE Transactions on Industrial Informatics*, 16(6):4177–4186, 2019.

[29] Omaji Samuel, Akogwu Blessing Omojo, Abdulkarim Musa Onuja, Yunisa Sunday, Prayag Tiwari, Deepak Gupta, Ghulam Hafeez, Adamu Sani Yahaya, Oluwaseun Jumoke Fatoba, and Shahab Shamshirband. Iomt: A covid-19 healthcare system driven by federated learning and blockchain. *IEEE Journal of Biomedical and Health Informatics*, 27(2):823–834, 2022.

[30] Zirun Zhao, Yi Sun, Ali Kashif Bashir, and Zhaowen Lin. Adadpfed: A differentially private federated learning algorithm with adaptive noise on non-iid data. *IEEE Transactions on Consumer Electronics*, 2023.

[31] Yunlong Lu, Xiaohong Huang, Yueyue Dai, Sabita Maharjan, and Yan Zhang. Blockchain and federated learning for privacy-preserved data sharing in industrial iot. *IEEE Transactions on Industrial Informatics*, 16(6):4177–4186, 2019.

[32] Aditya Pribadi Kalapaaking, Ibrahim Khalil, Mohammad Saidur Rahman, Mohammed Atiquzzaman, Xun Yi, and Mahathir Almashor. Blockchain-based federated learning with secure aggregation in trusted execution environment for internet-of-things. *IEEE Transactions on Industrial Informatics*, 19(2):1703–1714, 2022.

[33] Zihuan Xu and Lei Chen. L2chain: Towards high-performance, confidential and secure layer-2 blockchain solution for decentralized applications. *Proceedings of the VLDB Endowment*, 16(4):986–999, 2022.

[34] Changti Wu, Lei Zhang, Lin Xu, Kim-Kwang Raymond Choo, and Liangyu Zhong. Privacy-preserving serverless federated learning scheme for internet of things. *IEEE Internet of Things Journal*, 11 (12):22429–22438, 2024.

[35] Ming Li, Jian Weng, Anjia Yang, Wei Lu, Yue Zhang, Lin Hou, Jia-Nan Liu, Yang Xiang, and Robert H Deng. Crowdbc: A blockchain-based decentralized framework for crowdsourcing. *IEEE transactions on parallel and distributed systems*, 30(6):1251–1266, 2018.

[36] Bijun Wu and Oshani Seneviratne. Blockchain-based framework for scalable and incentivized federated learning. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 1761–1767, 2025.

[37] Zixiang Cui, Xintong Ling, Xingyu Zhou, Jiaheng Wang, Zhi Ding, and Xiqi Gao. Bagchain: A dual-functional blockchain leveraging bagging-based distributed learning. *arXiv preprint arXiv:2502.11464*, 2025.

[38] Jiawen Kang, Zehui Xiong, Dusit Niyato, Yuze Zou, Yang Zhang, and Mohsen Guizani. Reliable federated learning for mobile networks. *IEEE Wireless Communications*, 27(2):72–80, 2020.

[39] Ahmed Alagha, Jamal Bentahar, Hadi Otrok, Shakti Singh, and Rabeb Mizouni. Blockchain-assisted demonstration cloning for multiagent deep reinforcement learning. *IEEE Internet of Things Journal*, 11(5): 7710–7723, 2023.

[40] Guowen Dai, Jinjun Tang, Jie Zeng, Chen Hu, and Chuyun Zhao. Road network traffic flow prediction: A personalized federated learning method based on client reputation. *Computers and Electrical Engineering*, 120:109678, 2024.

[41] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE, 2008.

[42] Youyang Qu, Longxiang Gao, Yong Xiang, Shigen Shen, and Shui Yu. Fedtwin: Blockchain-enabled adaptive asynchronous federated learning for digital twin networks. *IEEE Network*, 36(6):183–190, 2022.

[43] Sahand Hariri, Matias Carrasco Kind, and Robert J Brunner. Extended isolation forest. *IEEE transactions on knowledge and data engineering*, 33(4):1479–1489, 2019.

[44] Liang Cai, Qilei Li, and Xiubo Liang. Hyperchain application development fundamentals. In *Advanced Blockchain Technology: Frameworks and Enterprise-Level Practices*, pages 273–292. Springer, 2022.