Hindawi
Security and Communication Networks
Volume 2022, Article ID 4202084, 18 pages
https://doi.org/10.1155/2022/4202084

Research Article

Towards Fair and Decentralized Federated Learning System for Gradient Boosting Decision Trees

Shiqi Gao,[1] Xianxian Li,[1,2] Zhenkui Shi,[1,2] Peng Liu,[1,2] and Chunpei Li[1]

[1]*Guangxi Key Lab of Multi-source Information Mining and Security, Guangxi Normal University, Guilin, China*
[2]*College of Computer Science and Engineering, Guangxi Normal University, Guilin, China*

Correspondence should be addressed to Xianxian Li; lixx@gxnu.edu.cn and Zhenkui Shi; shizhenkui@gxnu.edu.cn

Received 9 April 2022; Accepted 21 June 2022; Published 2 August 2022

Academic Editor: Andrea Michienzi

At present, gradient boosting decision trees (GBDTs) has become a popular machine learning algorithm and has shined in many data mining competitions and real-world applications for its salient results on classification, ranking, prediction, etc. Federated learning which aims to mitigate privacy risks and costs, enables many entities to keep data locally and train a model collaboratively under an orchestration service. However, most of the existing systems often fail to make an excellent trade-off between accuracy and communication. In addition, they overlook an important aspect: fairness such as performance gains from different parties' datasets. In this paper, we propose a novel federated GBDT scheme based on the blockchain which can achieve constant communication overhead and good model performance and quantify the contribution of each party. Specifically, we replace the tree-based communication scheme with the pure gradient-based scheme and compress the intermediate gradient information to a limit to achieve good model performance and constant communication overhead in skewed datasets. On the other hand, we introduce a novel contribution allocation scheme named split Shapley value, which can quantify the contribution of each party with a limited gradient update and provide a basis for monetary reward. Finally, we combine the quantification mechanism with blockchain organically and implement a closed-loop federated GBDT system FGBDT-Chain in a permissioned blockchain environment and conduct a comprehensive experiment on public datasets. The experimental results show that FGBDT-Chain achieves a good trade-off between accuracy, communication overhead, fairness, and security under large-scale skewed datasets.

1. Introduction

Machine learning (ML) has achieved extensive success in many practical applications. However, a well-trained ML model heavily depends on massive data. In reality, there may be sensitive information in the data sets which may lead to growing concerns about personal privacy and even national security. And data is considered as a valuable asset and a critical strategic resource increasingly. All these constraints greatly motivate federated learning (FL) [1], which enables multiple entities to collaboratively train a model under an orchestration service for immediate aggregation and store data locally. The data in FL may be generated at different contexts. This may lead the data distribution to be unbalanced or Non-IID. The data sets' scale and quality may be different. These may lead to different intermediate computation and communication cost for different parties. And data is a significantly important asset to organizations, so a nice FL scheme could stimulate and incent the parties with high-quality datasets to join the training to form a better model and guarantee their rewards that match their contribution in addition to privacy preservation. In this context, it is necessary to consider factors such as privacy protection, unbalanced/skewed data distribution, fairness, to form a closed-loop federated learning system (FLS) [2]. On the other hand, gradient boosting decision trees (GBDTs) has become a popular machine learning algorithm and has shined in many machine learning and data mining competitions [3, 4] as well as real-world applications for its salient results on classification, ranking, prediction, etc., (especially for tabular data mining task) [5]. And several works have studied the horizontal federated GBDT system

[6, 7]. They focus on training and publishing a single decision tree among multiple federated parties to compose the global ensemble model. But in these systems, there are still some challenges as follows:

(i) Balance of efficiency, learning accuracy, and privacy-preserving. In most of the existing schemes, each party trains a single decision tree, and then shares the tree with the next participating party [6, 8]. And the global communication cost of building each tree is a multiple of the corresponding trainer's data. Other schemes may adopt cryptographic methods or differential privacy [7]. Cryptographic methods may bring prohibitive overhead. And the accuracy is relatively lower in the existing federated GBDT scheme with differential privacy in skewed data distribution.

(ii) Contribution quantification. Many data owners may not actively participate in federated learning, especially when the data owners are enterprises rather than devices [9]. As mentioned previously, a nice FL scheme could stimulate parties with high quality datasets to join the training to train a better model and guarantee their rewards that match their contribution. It is also essential to prevent participants from inflating their contributions. Most of the existing schemes overlooked this and failed to provide an outstanding quantifying mechanism.

(iii) Accuracy measurement and verification. In the FL setting, there is no guarantee that all parties are honest and trusted. To tackle these issues, [6] proposed to use MAE to measure the accuracy, and [8] adopted the blockchain for verification. However, it leads to additional communication overhead to achieve higher accuracy. It is necessary to consider two factors in accuracy measurement: (1) whether the feature with the most information gain is correctly selected; (2) whether the samples are in the correct sorting position [10]. To the best of our knowledge, there is no effective solution to measure and verify the accuracy contribution of each party.

In response to the above challenges, we propose a closed-loop federated GBDT system FGBDT-Chain which consists of two components: FV-tree and FQ-chain. More specifically, FV-tree is our federated GBDT framework. And we combine FV-tree with blockchain organically and design FQ-chain to quantify the contribution logic on the smart contract to attain a decentralized verification and auditability. Our scheme can achieve a relatively better balance of efficiency, learning accuracy, and privacy-preserving in skewed distribution of data. Particularly, it can also quantify parties' contribution for the global model, provide a value-driven incentive mechanism that encourages parties with different data sets to be honest, and suit to large-scale datasets.

Our contributions can be summarized as follows:

(1) We propose FV-tree, a federated GBDT framework that can achieve constant communication cost and less precision loss in skewed distributed data. FV-tree is based on the data-parallel algorithm of the decision tree to find the global top-2 candidate features and utilizes private spatial decomposition (PSD) to capture other parties' distribution and refits gradients to vote on the local most informative feature. We also design a scalable differential privacy mechanism in this process to enhance privacy-preserving.

(2) We design a contribution quantifying mechanism with a metric, namely, *split Shapley value* and a decentralized verification endorsement mechanism, namely, FQ-chain, which can reach a relatively fair and auditable federated GBDT. It can encourage and incent organizations with different datasets to train a better model.

(3) We implement the system FGBDT-Chain in a permissioned blockchain environment and conduct a comprehensive experiment on public datasets. The results show that FGBDT-Chain has high performance and can meet the practical application, especially for large-scale datasets.

The rest of the paper is organized as follows. Section 2 reviews the related work about federated GBDT systems. Section 4 introduces the design outline of our system. The technical details of FV-tree and FGBDT-Chain are introduced in Section 5. Section 6 presents the performance evaluation of our system in terms of accuracy and fairness. We give a brief discussion and analysis in Section 7. Section 8 summarizes the paper and puts forward the potential research directions in the future.

2. Related Work

In this section, we review the literature on the federated GBDT and fairness in federated learning.

2.1. Federated Gradient Boosting Decision Tree. Gradient boosting decision tree (GBDT) and its effective implementations such as XGBoost [3] and LightGBM [4] are widely used machine learning both in industry and academic applications [5, 11, 12]. In distributed GBDT, the training data is located in different machines and should be partitioned according to the sample level. Generally, the local histograms of features are broadcasted to all the parties to obtain the global distribution. Then each party chooses the most informative splitting points [13]. Among them, the parallel voting decision tree (PV-tree) [14] is a representative scheme. It performs full-granular histogram communication according to the features selected by each machine, then calculates the global split point. PV-tree can achieve a very low communication cost (independent of the total number of features/samples) in the context of uniform data distribution and has great scalability in the context of large datasets.

In recent years, with the growing concerns about data security and privacy, several horizontal federated GBDT systems have been developed. [6] designed a distributed GBDT scheme, in which each party trains a differential

privacy decision tree and uses Mean Absolute Error (MAE) to evaluate the accuracy of each decision tree. [8] took a similar approach and extended this learning process to the blockchain. However, in these tree-based sharing schemes, the quality of the shared tree is low. To solve this problem, [7] proposed Sim-FL, in which, each instance gathers similar instances' gradients of other parties through a local sensitive hash (LSH) to learn the distribution of other parties. This weighted gradient boosting strategy can significantly improve the accuracy of each decision tree, and achieve a primary level of privacy protection. Unfortunately, the communication overhead in each iteration is proportion to the number of local instances in the training party, which is not feasible in large-scale datasets learning. Intuitively, we summarize the existing federated GBDT system and compare them with our scheme in Table 1.

2.2. Fairness in Federated Learning. Many data owners may not actively participate in federated learning, especially when the data owners are enterprises rather than devices [9]. Therefore, the fairness of the federated learning system needs to be taken into account. In the existing federated learning research, fairness is mainly realized through an incentive mechanism. There are two main ideas: (i). All parties enjoy a global model; (ii). According to the contribution of parties, parties get different model rewards [15].

The goal of incentive mechanism is to make the party get a reward commensurate with its contribution. A number of literature focused on designing incentive mechanisms by clients' resources [16] and reputation [17]. Whereas, we concentrate on the incentive mechanism based on the contribution of data quality. Because data quality is a key factor that affects the model. In the scheme based on data quality contribution, Shapley value [18] has a wide range of applications, and [15, 19, 20] studied the Shapley Value of the data point contribution during ML training. In the training process of federated learning, [21] proposed to record the intermediate results (i.e. gradients and models), and then use them to reconstruct the model for approximate the contribution indexes. This approach is efficient and feasible in horizontal federated learning. Unfortunately, there is an essential difference between gradient-based distributed GBDT and Gradient Descent-based algorithms. Because reconstructed models are not always useful and internal nodes will not affect the prediction score. Therefore, we need a new contribution measurement mechanism for the scenario without an intermediate model.

In addition, some works use blockchain technology to record the training milestones of clients and ensure the security of the incentive mechanism [22–24]. These works do not promise a good balance of privacy-preserving, efficiency, and learning accuracy to form a practical federated GBDT.

3. Preliminaries

3.1. GBDT. GBDT is an ensemble model of sequential training for several decision trees. In each iteration, the following objective function is minimized to fit the residual of previous learners [25]:

$$\tilde{\mathscr{L}}^{(t)} = \sum_{i}^{n}\left[g_i f_t(\mathbf{x}_i) + \frac{1}{2} f_t^2(\mathbf{x}_i)\right] + \Omega(f_t), \tag{1}$$

where $g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$ is first-order gradient and $\Omega(f)$ is a regularization term. Let $I = I_L \cup I_R$, where I is the instance set of the father node, I_L and I_R are the instance sets of left and right nodes after a split. The gain of a split point is given by:

$$G(I_L, I_R) = \left(\frac{\left(\sum_{i\in I_L} g_i\right)^2}{|I_L| + \lambda}\right) + \left(\frac{\left(\sum_{i\in I_R} g_i\right)^2}{|I_R| + \lambda}\right). \tag{2}$$

To reduce the computational complexity of traversing all feature values, histogram-based algorithms like [4, 26] use discrete bins to find the approximate optimal split. The detail of the histogram-based algorithm as shown in Algorithm 1.

3.2. Private Spatial Decompositions (PSD). Generally, any dataset with ordered attributes or moderate to high cardinality (e.g. numerical features such as salary) can be considered as spatial data. In addition, if a dataset can be indexed through a tree structure (such as a B-tree, R-tree, kd-tree etc.), it can be implicitly treated as spatial [27]. Formally, a spatial decomposition is a hierarchical (tree) decomposition of a geometric space into smaller areas/hyperspaces, with data points partitioned among the leaves. Indexes are usually computed down to a level where the leaves either contain a small number of points, or have a small enough area, or a combination of the two. There have been many approaches to spatial decompositions. Some are data-independent, such as quadtrees which recursively divide the data space into equal quadrants. Other methods, such as the popular kd-trees, aim to better capture the data distribution, and they are data-dependent. [27] gives a full framework for privately representing spatial data. We use the PSD to share a coarse distribution summary with other data owners. And it is both used in collaborative learning and calculation verification under statistical heterogeneity scenarios.

3.3. Blockchain. Blockchain [28] is a kind of chained data structure that combines data blocks in order according to time sequence. The append-only data are ensured that they are tamperproof and unforgeable through cryptographic primitives. The main advantages of blockchain are decentralization, security, transparency, and traceability. Hyperledger Fabric [29] is a popular and efficient enterprise-level permissioned blockchain framework. And Fabric also realizes the modularization of consensus mechanism, authentication, and other components, which is more suitable for business cooperation between enterprise organizations. In summary, the fabric can provide a decentralized trust environment for a group of organizations to carry out complex business transactions for collaborative GBDT training tasks.

Table 1: Compare with existing federated GBDT systems.

	Accuracy[1]	DP[2]	Shared information[3]	Communication efficiency[4]	Blockchained[5]
[6]	×	✓	Model	✓	×
[8]	✓	×	Model + gradients	×	×
[7]	×	✓	Model	✓	✓
Our scheme	✓	✓	Gradients	✓	✓

[1] The accuracy of federated GBDT model performance well in skewed data distribution. Notice: "×" representative does not meet the requirement, "✓" meets the requirements. [2] The system has differential privacy extensibility. [3] The system's communication architecture, especially the shared training information in federated GBDT training. [4] The communication cost is independent of the number of samples in the local dataset. [5] In the absence of a third-party server (none of the above systems need it), the blockchain is used as an autonomous platform to coordinate the training process.

Input: I: instance set of the current node, F:feature set.
Output: bestSplit.
forall f in F **do**.
H ← new Histogram();
forall x in I **do**.
bin ← x[f].bin;
H[bin].g ← H[bin].$g + x$.gradient;
H[bin].n ← H[bin].$n + 1$;
forall bin in H **do**.
leftSum, rightSum = CalSumFromSplit(bin);
split.gain = SplitGain(leftSum, rightSum);//(2) ;
bestSplit = ChoiceBetterOne(split, bestSplit);
return bestSplit.

Algorithm 1: FindBestSplit.

4. The FGBDT-Chain Framework

This section describes the overall design of FGBDT-Chain, including the design objectives and system overview. We adopt the general assumption of federated learning, in which one model requester publishes a model request and multiple parties participant in the collaborative learning task. The problem description is included in Section 3-A. The system summary is shown in Section 3-B. The main symbols used in this paper are given in Table 2.

4.1. Design Objectives. We assume that there are M parties, and each party is denoted by P_m ($m \in [1, M]$). We use $I_m = \{(x_i^m, y_i^m)\}$ to denote the instance set of P_m, where $x_i^m \in \mathbb{R}^f$, $y_i^m \in \mathbb{R}$. We focus on the collaborative training of GBDT model, in which M parties (data owners) include one requestor cooperate to implement a federated GBDT training task. For example, as shown in Figure 1, due to the different distribution of patients, two private hospitals P_1, P_2 may prefer accurate test predictions for female and young patients, respectively [15]. Without relying on unrealistic public datasets and third-party central servers, they hope to achieve peer-to-peer collaborative learning and obtain high-quality models in a trusted environment. More importantly, they need to be guaranteed that they can get rewards corresponding to their own contributions. Out of this assumption, our federated GBDT system tries to meet the following three objectives:

(i) Model accuracy and efficiency. It is the basic requirement of all parties to build a high-quality global model in multiple skewed data sets. In addition, the geographical distance between parties may be far away, and the intermediate process can be stored in blockchain for the sake of fairness and security. The communication cost should be strictly reasonable. For this reason, we propose FV-tree, which can reduce the communication to a small range, and obtain good model performance in the case of skewed data distribution.

(ii) Fairness: As mentioned previously, data is considered a valuable asset and a critical strategic resource increasingly. In addition, participants need to invest tremendous of computation and storage in FL. Without any revenue, data owners may not voluntarily provide data and training resources. To encourage more parties to participate in a collaborative learning program, it is necessary to accurately calculate the cooperative contribution of each participant. We use the split gain generated by the party's updated gradients to calculate the split Shapley value of each party. In this way, we can fairly quantify the contribution of each party in the whole process, and provide the mechanism for the monetary reward of delayed payment.

(iii) **Security:** We assume that parties are curious, and they will not maliciously attack the federated model

Table 2: List of symbols.

Symbols	Meaning
P_m	m-th party in federated learning task;
M	Number of participating party;
I_m	Instance set of party P_m;
T	Number of decision trees in GBDT;
d	Maximum depth of decision tree;
Q	Total number of ensemble model split;
h_m	P_m's histogram of ordered gradients;
$\text{bin}_g, \text{bin}_n$	The sum of gradients and counts of each bin in one histogram;
gain^q	The split gain of q-th split in the GBDT model;
split^q	The split point of q-th split in the GBDT model, which includes the split feature and split threshold;
psd_m	Privacy spatial decomposition structure of P_m;
c_q^m, C_m	They represent the P_m's contribution index of the q-th split and the contribution index of total splits respectively;
ϕ_m^q	P_m's split indexes (split Shapley value) during the $q-th$ split;
κ_m^q	P_m's voting contribution indexes during the $q-th$ split;
$\mathbf{W^m}$	The P_m' distribution weight matrix;
$\mathbf{w^{m*}}$	The P_m' global distribution weight vector;
pk_m, sk_m	The P_m' key pair for signing and verification respectively;

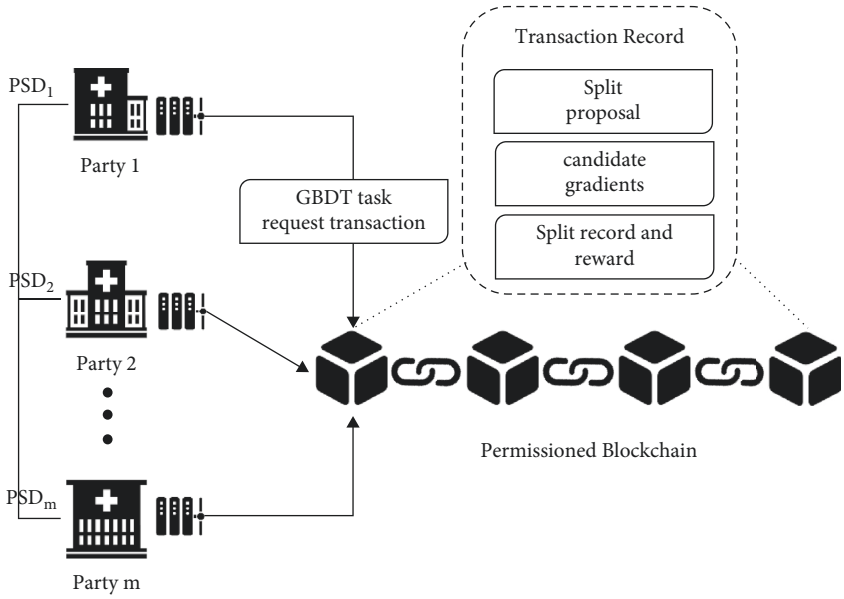


Figure 1: FGBDT-Chain system overview.

unless they can get higher income. This means that our system not only needs to avoid leaking the original data in the learning process but also needs to provide a necessary verification mechanism. We also have to eliminate the potential that greedy participants deliberately exaggerate contribution through updated information. Therefore, we propose FGBDT-Chain which can provide an extension of differential privacy, and a decentralized endorsement mechanism to filter distorted update information.

4.2. The Proposed Architecture. Our proposed system consists of two modules: permissioned blockchain module and federated GBDT module. The permissioned blockchain establishes secure connection channels among all nodes. FGBDT-Chain is based on the FV-tree training framework, which includes three stages: distribution preprocessing, features voting, and gradient histogram aggregation. Permissioned blockchain module includes four types of transactions: model request transaction, feature voting transaction, gradient histogram upload transaction, and contribution indexes allocation transaction. The contribution indexes assignment is implemented by smart contracts according to historical transactions. The stored information in the permissioned blockchain is shown in Figure 2.

Step 1. In the beginning, a model requester initializes the permissioned blockchain and specifies the requirements of the learning task, such as dataset requirements and model

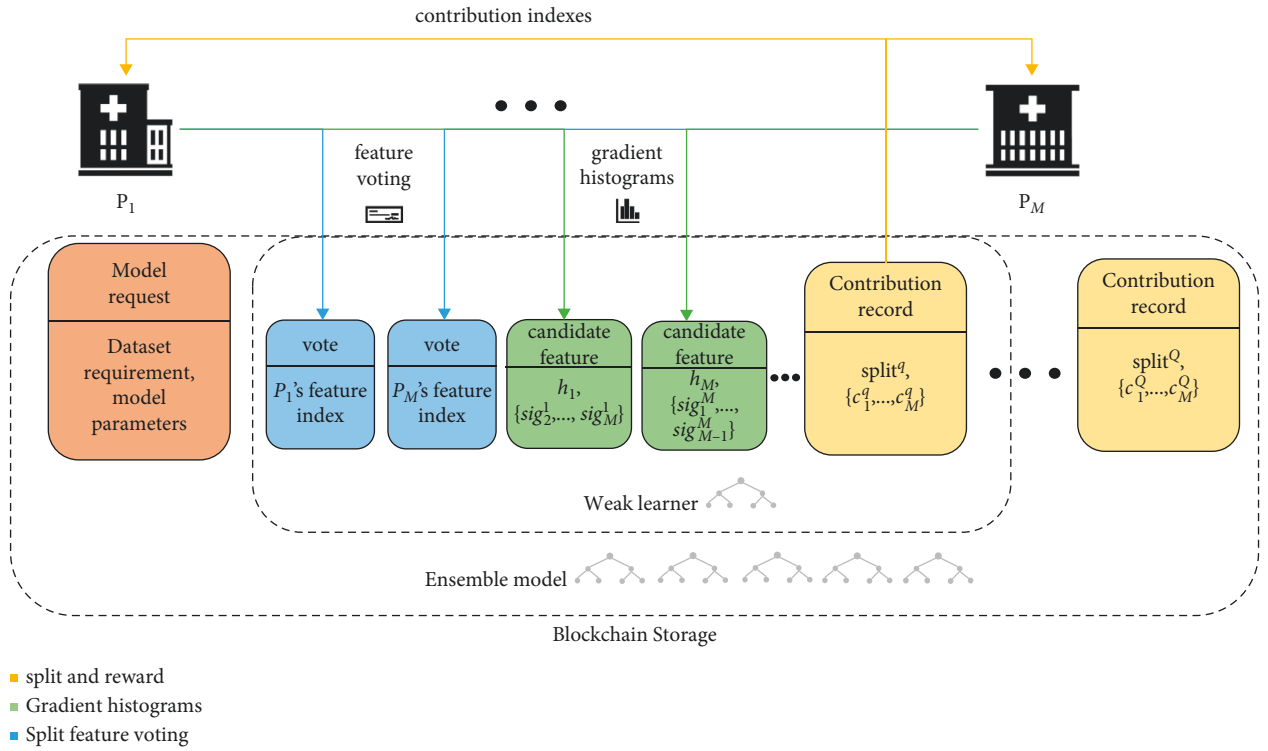


Figure 2: Blockchain-based ledger storage of FGBDT-Chain system.

parameters. Parties that wish to join the learning task or receive a request should be authenticated, then upload the rough distribution summary (i.e., PSD) of their datasets. The model requester has the right to refuse a party to become a federation member according to the observation of the distribution summary.

Step 2. After a specified number of organizations join the federated learning task, each party downloads all PSDs, and establishes the distribution matrix and global distribution vector. So far, the initialization work is completed.

Step 3. In the stage of collaborative training, each party uses the local dataset I_m and the global distribution vector to calculate the local most informative features and uploads the feature index through the voting transaction. At the same time, all parties can calculate the top-2 features with the highest number of votes as candidate features according to on-chained transactions.

Step 4. Parties broadcast the local original gradient histograms of candidate features. After one party receives most signatures corresponding to his histogram, the histograms and signature set are written into the transaction. With the help of the distribution matrix, the verification algorithm can detect malicious updates in skewed data distribution (Malicious update refers to the gradient histogram stretched by greedy participants to improve their contribution indicators).

Step 5. The smart contract will calculate the best split point and allocate contribution indexes according to the historical transactions. These two sub operations can be parallelized and the complexity is low. In addition, since the update records are stored in transactions, the contribution indexes can be calculated after the emergency task training process is completed.

The above 3–5 steps will form a loop that continues to execute until the stop training condition is met. When the learning task is finished, the federated GBDT model and parties' update/contribution records are stored in the blockchain's transactions. The whole learning process does not depend on any single party. In addition, because all the records created during the training of the decision tree are tamper-proof, the federated member can be audited at any time.

5. The Design Detail of FGBDT-Chain

FGBDT-Chain is a collaborative learning framework based on blockchain for GBDT. We will introduce the framework in two parts: FV-tree and FGBDT-Chain. Firstly, we will introduce the PSD-based preprocessing phase, which provides the basis for our framework (Section 5-A). Secondly, we will describe the GBDT training framework FV-tree in detail, which includes tree growth processes based on feature voting, gradient histograms publishing, and the expansion of differential privacy (Section 5-B). Finally, we introduce FGBDT-Chain's fairness assurance, including the fair guaranteed incentive mechanism based on a novel

contribution measurement algorithm, and the decentralized verification scheme on the blockchain (Section 5-C).

5.1. Preprocessing Stage. When a party receives the model request transaction, it first checks the dataset requirements and filters out the instances that meet the task description in the local instance, which is expressed as I_m. Then it starts the preprocessing operations. The main idea is to capture the data distribution of all other parties by generating a rough distribution matrix $W^m \in \mathbb{R}^{N_m \times M}$ and a global distribution vector $w^{m*} \in \mathbb{R}^{N_m}$. Where W_{ij}^m is the distribution weight of P_m's instance x_i^m in party P_j's instance set I_j, and w_i^{m*} is the distribution weight of the instance x_i^m in the global instance set I. In our scheme, w^{m*} is an optional term. When distributions are badly skewed, it will be used in the voting stage to select the most informative local feature (Section 5-B1), and W^m is used for verification subsequently (Section 5-C2).

More specifically, party P_m firstly calculates the psd_m by I_m, which has been well studied in previous research [27]. Let V_l^m be the value of l-th leaf in psd_m. Intuitively, the psd_m is a tree model represents the rough data distribution summary of P_m, where the value V_l^m is the number of instances corresponding to the hyper-space represented by the leave node l, and the count value V_l^m has been perturbed by differential privacy. Party P_m can upload psd_m with the blockchain's transaction, and download other parties' psd in the collaborative learning task. Then P_m maintains the distribution weight matrix W^m and the global distribution weight vector w^{m*}. The detail is shown in Algorithm 2. After party P_m downloads psd_j from P_j, it uses a local instance set I_m to query psd_j. Assuming that the query result of i-th instance (x_i^m, y_i^m) is l-th leaf in psd_j, then P_m pushes index i into the set S_l^j, where S_l^j is the set of P_m's instances falling in the hyperspace psd_l^j. After all instances have been queried, W^m can be assigned, where $W_{ij}^m = (|S_l^j|_{i \in S_l^j})/V_l^j$. Finally, after calculating the distribution vectors $W_1^m, \ldots, W_j^m$ of all other participants, P_m will further assign the global distribution vector w^{m*}, as follows:

$$w_i^{m*} = \delta \sum_{j=1}^{M} \left(W_{ij}^m \times \frac{N_j}{N} \right), = \delta \sum_{j=1}^{M} \left(W_{ij}^m \times \frac{\sum_{l=1}^{L_j} V_l^j}{\sum_{k=1}^{M} \sum_{l=1}^{L_k} V_l^k} \right), \tag{3}$$

where δ is a parameter of fitting distribution degree, N and N_j denote the number of instances of global and party P_j respectively, which is got from the accumulated leaves' value of different psd s. In addition, N_j/N represents a fitting budget of P_j. The more instances a party has, the larger fitting budget needs to be allocated. For Algorithm 2, we have the following observations. Firstly, the calculation of PSD only needs one time, and the distributed structure of tree model will greatly reduce the communication cost compared with the approach of sending each sample hash [7]. Secondly, the structure of psd s can be different, which means parties do not need to communicate in advance to use a unified structure of psd. In other words, parties can choose any tree model or inner nodes, whether it is a quad-tree or a kd-tree. It will not affect other parties to generate their weight matrix.

5.2. FV-Tree. When the local weight matrix W^m and global weight vector w^{m*} are established, parties can start to enter the training stage. In the training phase, each party does not train a complete tree, instead, it sends minimal update information. There are two types of update information: (i) parties' split feature voting and (ii) gradient histogram of candidate feature which is used to calculate global split points. In each node split, parties calculate the split feature with the most informative gain locally and vote on it. The top-2 features with majority votes in the global voting will become candidate features, and then parties send the gradient histograms of them. According to the above two kinds of update information, each party can update the global GBDT model synchronously.

However, this method may produce errors due to the split feature may be not globally optimal, especially in the context of decentralized data owners with different distributions/sizes. So, we consider *gradient refit* to alleviate this problem. The basic idea of gradient refit is to adjust gradients according to the global weights of the instances, then calculate the most informative feature according to the refitted gradients. When the global candidate features are selected, the two local original histograms are sent. The details of FV-tree are shown below.

At the beginning of an iteration, party P_m has a local instance set I_m, and the global distribution weight vector w^{m*}. First, P_m updates gradients and synchronizes the split information of each new node. Details are shown in the Algorithm 3 and Figure 3. For each new node generated in the decision tree, P_m calculates the local split gain of all the split points. The split gain is calculated as follows:

$$G_m(I_L, I_R) = \frac{\left(\sum_{i \in I_L} g_i^m w_i^{m*} \right)^2}{\sum_{i \in I_L} w_i^{m*} + \lambda} + \frac{\left(\sum_{i \in I_r} g_i w_i^{m*} \right)^2}{\sum_{i \in I_R} w_i^{m*} + \lambda}. \tag{4}$$

When the local split point with the highest split gain is selected, party P_m will publish the corresponding feature's index as a vote. And after receiving all the local votes, every party can sort features according to the number of votes. So far, each party can get the ranking of the same features, then select the top-2 features as candidate features, and upload the corresponding gradient histograms. It should be noted that the original uploaded gradients histogram is not the fitted one. After receiving the histograms from other parties, each party will traverse all the split points in the aggregated histograms to find the best split with the highest split gain. The gain of each split point is calculated as follows:

$$G_{\text{global}} = \frac{1}{2} \left[\frac{\left(\sum_{m=1}^{M} \sum_{i \in I_L^m} g_i^m \right)^2}{\sum_{m=1}^{M} \left| I_L^m \right| + \lambda} + \frac{\left(\sum_{m=1}^{M} \sum_{i \in I_R^m} g_i^m \right)^2}{\sum_{m=1}^{M} \left| I_R^m \right| + \lambda} \right], \tag{5}$$

where, $\sum_{i \in I_L^m} g_i^m$, $\sum_{i \in I_R^m} g_i^m$, $\sum_{m=1}^{M} |I_L^m|$, and $\sum_{m=1}^{M} |I_R^m|$ are calculated from the aggregated histograms. When the node

Input: PSD model set psd_1, psd_2..., psd_M, instance set I_m
Output: distribution weight matrix: W; global distribution vector: w^*
//establish distribution weight matrix
for $j \leftarrow 1$ to M **do**
for $i \leftarrow 1$ to N_m **do**
S $\leftarrow \text{psd}_j$.getLeafNode((x_i, y_i));
S.push(i);
//set hyperspace′s weight to matrix W
for $l \leftarrow 1$ to L_j **do**
S_j.weight $\leftarrow |S_l^j|_{i \in S_l^j} / V_l^j$;
forall i in S_l^j **do**
W [i][j] $\leftarrow S_l^j$.weight;
//establish global distribution vector
for $i \leftarrow 1$ to N_m **do**
for $j \leftarrow 1$ to M **do**
w^* [i] += W [i][j] $\times N_j / N$;
return W, w^*;

Algorithm 2: FVtree:DistributionMatrixEstablish.

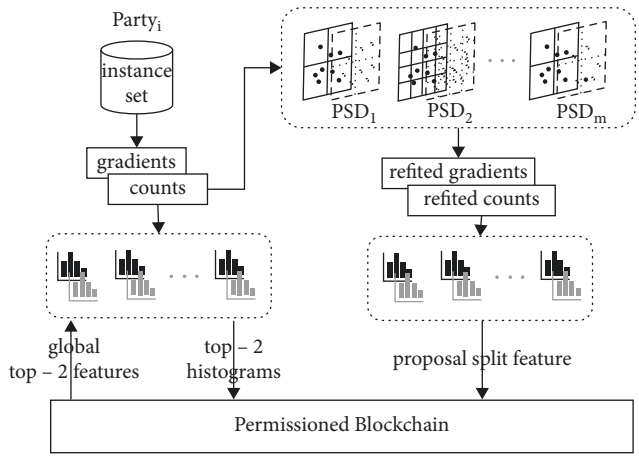


Figure 3: Training process of FV-tree.

Input: local gradients $g_1, \ldots, g_{N_m}$, global distribution weight vector w^{m*}
Output: bestSplit
localHistograms = ConstructHistograms($g_1, \ldots, g_{N_m}$);
localRefittedHistograms = ConstructHistograms($g_1, \ldots, g_{N_m}$, w^{m*});
//Local Voting
forall H in localRefittedHistograms **do**
splits.Push(H.FindBestSplit())//For details in Algorithm 1;
localVote = Max(splits).getFeatureID();
uploadVote(localVote);
//Global Voting
featureRanking ← gather other parties' localVote;
globalCandidate = featureRanking.Top2ByMajority();
uploadHistograms(globalCandidate, localHistograms);
//Merge global histograms
globalHistograms ← gather other parties' localHistograms;
bestSplit = globalHistograms.FindBestSplit();
return bestSplit;

Algorithm 3: FV-tree:FindBestSplit.

reaches the max depth, it becomes a leaf node and the value is calculated through the following equation:

$$\text{Value} = -\left(\frac{\sum_{m=1}^{M}\sum_{i\in I_m} g_i^m}{\sum_{m=1}^{M}|I_m| + \lambda}\right). \tag{6}$$

In the training process of FV-tree, a participant needs to update information from other parties to split none-leaf node, and the value of a leaf node is directly generated by the histograms of its parent node. So, we only need to allocate the privacy budget to the none-leaf nodes. In the communication process of FV-tree, local feature voting and histograms aggregation may lead to privacy leakage. For the local best split point selection, the information gain is used as the utility function, and the exponential mechanism is used to return the split point with the largest gain value. Let g^* be the gradient with the largest absolute value. By introducing the conclusion of previous work [13], the sensitivity is $\Delta G = ((3\lambda + 2)/((\lambda + 1)(\lambda + 2)))g^*$. Before updating histograms, the count of each bin is perturbed by Laplace noise [14]. The sensitivity of the gradient histogram is $2g^*$, and the sensitivity of the count histogram is 1. To maintain the effectiveness of boosting, we use the two-level boosting structure (EOE) to allocate the privacy budget for multiple decision trees [13], and our method satisfies the ϵ-differential privacy.

Proof. Assume that the privacy budget of a tree is ϵ_t, and the max depth of a decision tree is d. Since the nodes in one depth have disjoint inputs according to the parallel composition, each instance will go through at most $d - 1$ times node split. Further, each split will be regarded as five queries, namely, the best split feature voting and twice gradient histograms and count histograms updating respectively. The privacy budget for each split is $\epsilon_{\text{split}} = (\epsilon_t/5(d-1))$. Thence, the privacy budget of a single decision tree satisfies ϵ_t-differential privacy. In EOE, if there are a total of E ensembles, the privacy budget of each tree is $\epsilon_t = \epsilon/E$, and the whole FV-tree training process satisfies ϵ-differential privacy.

In summary, our scheme leverages voting split features and updating gradient histogram to make a tradeoff between accuracy, communication cost and security, and we give a brief discussion in section 7-A. □

5.3. FGBDT-Chain. To attract more institutions with high-quality data into the federal learning task, it is necessary to quantify the contribution of each party fairly and provide incentive mechanisms according to the contribution index. A widely used approach is to quantify the contribution of each participant's local model [9]. However, it is infeasible when the local model does not exist. For example, in our FV-tree scheme, there is no local model, and split points are decided by all parties. We should design a new approach and mechanism to quantify the contribution of federated parties. We first define the fairness of the federated GBDT task.

Definition 1. (Collaborative fairness in GBDT) In a collaborative GBDT learning task, multiple parties train a global model together. The party that provides more valuable information for the global model will get a higher contribution index. Specifically, fairness can be measured by the parties' split gain.

We define what is valuable information as follows.

Definition 2. (Valuable information in gradient-based collaborative GBDT): Suppose party P and P' participate in distributed GBDT learning. Once the global best split point is determined, we can informally say that party P provides more valuable information than P', if the gradients submitted by P bring more split gain than the gradients submitted by P' on the global split point.

The growing process of decision tree is to constantly find the split point which can bring the maximum split gain. The split gain provided by party's update information for the global model can reflect the corresponding contribution because split gain represents the reduced uncertainty in the selection process of the split point. Formally, let $C \triangleq \{P_1, \ldots, P_M\}$ denote a set of M parties. We call a subset B a coalition of parties if $B \subseteq C$. The histogram vector of $P_m \in B$ is represented by h_m, coalition B's histogram set is denoted by HB. And we denote the best splitting point as split^q, the global gain of split^q is G^q. Then, we define the utility function U_B:

$$U_B \triangleq G(\text{HB}; \text{split}_q) = \frac{\left(\sum_{m\in B}\sum_{bin\in h_L^m}\text{bin}_g\right)^2}{\sum_{m\in B}\sum_{\text{bin}\in h_L^m}\text{bin}_n + \lambda} + \frac{\left(\sum_{m\in B}\sum_{bin\in h_R^m}\text{bin}_g\right)^2}{\sum_{m\in B}\sum_{bin\in h_R^m}\text{bin}_n + \lambda}. \tag{7}$$

The above equation is the histogram form transformed from (5). Where h_L^m/ h_R^m denote the set of bins on the left/right parts segmented by split^q, bin_g and bin_n denote the sum of gradients and counts in the corresponding bin respectively. According to the observation of (7), two properties fulfill the standard assumptions of cooperative game theory:

Property 1. Histogram of the empty coalition has no utility: $U_\varnothing = 0$;

Property 2. Histogram of any coalition $B \subseteq C$ has nonnegative value: $\forall B \subseteq C, U_B \geq 0$;

Proof. The above two properties can be proved simply. For Property 1, when $B \subseteq \varnothing$, each bin in HB equals 0, so the $G(\text{HB}; \text{split}_q)$ equals 0. For Property 2, because $(\sum_{m\in B}\sum_{\text{bin}\in h_L^m}\text{bin}_g)^2 \geq 0$, and n_{bin} is a natural number, the minimum value of U_B is $(0/\lambda) + (0/\lambda) = 0$.

To guarantee that the histograms' contribution measurement is fair to all M parties, we use Shapley Value, which is the unique value division scheme that satisfies symmetry, null player, additivity, and efficiency properties. Next, we define the contribution of a federated party in a single split: □

Definition 3. (Split Shapley value) In the q-th node split split^q of federated GBDT model, given a utility function $U \triangleq G$ where G is the split gain function of GBDT algorithm,

and a histogram set $HC \triangleq \{h^m\}_{m\in\{1,M\}}$, the split Shapley value of a federated party $P_m \in C$ is defined as:

$$\phi(h_m; U, \mathrm{HC}) \triangleq \frac{1}{M}\sum_{j=1}^{M}\frac{1}{\binom{M-1}{j-1}}\sum_{\mathrm{HB}\subseteq \mathrm{HC}\setminus\{h_m\}:\ |\mathrm{HB}|=j-1}\left(U\left(\mathrm{HB}\cup\{h_m\};\mathrm{split}^q\right)-U\left(\mathrm{HB};\mathrm{split}^q\right)\right). \tag{8}$$

For simplicity, we use ϕ_m^q denotes the split Shapley value of P_m at the q-th splitting, it can be called as split contribution index.

In addition to the split contribution, the voting contributions are required to encourage parties to choose the most informative features. In the q-th split, the voting contribution κ of P_m is defined as:

$$\kappa_m^q = \begin{cases} 0, & \text{If } P_m\text{'s vote hits the split feature,} \\ G_q, & \text{If } P_m\text{'s vote does not hit the split feature,} \end{cases} \tag{9}$$

Finally, the party P_m's total contribution index of the q-th splitting of the federated GBDT model is defined as c_m^q:

$$c_m^q = \alpha\kappa_m^q + \phi_m^q, \tag{10}$$

where κ_m^q is the voting contribution, $\alpha \in (0, 1]$ is a variable parameter that controls the voting contribution, and ϕ_m^q is the split contribution comes from Equation[eq_split]. When the federated GBDT model training is complete, the contribution of party P_m is $C_m = \sum_{q=1}^{Q} c_m^q$, where Q is the total number of split (number of nonleaf nodes).

In the previous section, we described in detail how to quantify the contribution of a party. However, it is a challenge to calculate C when there is no trusted third party because C is directly related to the interests of each participant. To ensure the security of the logic of contribution measurement, we use a smart contract to retrieve historical transactions and record the contribution of each party.

Even smart contract can achieve the security of computing process, due to the sensitivity of split Shapley value, greedy parties can get a higher split contribution ϕ by tampering with the local histograms. As a concrete example, it is shown in Table 3. Suppose two parties P_1, and P_2 submitted their local histogram transactions h_1 and h_2 where $h_1 = \{\{1, 2\}, \{10, 10\}\}$, $h_2 = \{\{-1, 1\}, \{10, 10\}\}$. For simplicity, let $\lambda = 0$, we can get G is 0.45, and split contribution ϕ of P_1 and P_2 was 0.375 and 0.075, respectively. However, if P_2 tampers with its gradient histogram h_2 by doubling the magnification, the global G increases to 0.85. Accordingly, the split contribution ϕ_1, ϕ_2 is changed to 0.275 and 0.575. It can be seen that P_2 has increased his split contribution a lot.

Based on the above analysis, it is necessary to verify the updated information in our system to maintain fairness. In federated GBDT, the only existing verification scheme is to use local datasets to measure the performance of the updated model [6, 8]. Because it is difficult to generate public validation data sets, this scheme is considered as a minimized method in the federated scenario [30]. We inherit this idea of using a local dataset as the basis of verification. However, we cannot directly use the performance of the model, the reasons are as follows: First, updating information in FV-tree is gradients rather than models. Using gradients to reconstruct a model requires additional calculation; Secondly, the verification of model quality cannot fundamentally solve the above problem, because the contribution value of a histogram will be significantly higher after it is stretched proportionally. But the quality of the model using the stretched histogram may not be much different from the original one. In response to the above problems, we take the histogram overlap degree as the verification algorithm, in which the histogram used for verification is constructed by the distribution matrix W and the local histogram h. And we integrate this method into the endorsement mechanism of the permissioned blockchain to implement the FV-tree's decentralized verification scheme.

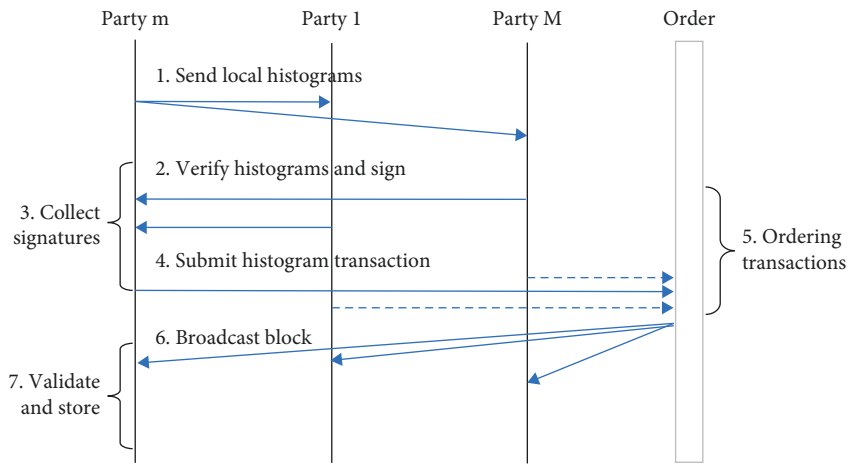
Specifically, as shown in Figure 4, before party P_m submits a histogram transaction, it first needs to broadcast the histogram h_m to other parties for signature. When $P_j \in C\setminus\{P_m\}$ received the signature request of h_m from P_m, the P_j' local gradients and the distribution vector W_m^j will be used to construct the refitted histogram h_m^{j*}, which denotes the histogram constructed by P_j to verify h_m. For P_j, there is only its histogram, which can simply denote as h_m^*. The details of this process are similar to Algorithm 1, except that $x \cdot \mathrm{gradient}, 1$ are replaced by $gW_{m,i}^j$ and $W_{m,i}^j$ in line 5 and line 6 respectively. Then h_m^* is used to calculate the overlapping degree with h_m:

$$\mathrm{Ver}\left(h_m, h_m^{j*}\right) = \sum_{bin\in h_m}\left(\frac{\left|\mathrm{bin}_g^m - \mathrm{bin}_g^{m*}\right|}{\max\left(\mathrm{bin}_g^m, \mathrm{bin}_g^{m*}\right)} + \sum_{bin\in h_m}\frac{\left|\mathrm{bin}_n^m - \mathrm{bin}_n^{m*}\right|}{\max\left(\mathrm{bin}_n^m, \mathrm{bin}_n^{m*}\right)}\right), \tag{11}$$

where bin_g^m, bin_n^m denote cumulative gradients and count respectively. The overlapping degree can verify the correlation of bin values and whether they are stretched. When the overlapping degree is less than the threshold, P_j will sign the histogram h_m, and send $\mathrm{sig}(h_m, sk_j)$ to P_m, where sk_j is a private key of P_j. When P_m obtains the signatures of most parties, it will write the histogram and signature set into the transaction and sends it to orderers, then the histogram transaction will be packaged into block.

Table 3: An example of the influence of local histogram h on split contribution ϕ.

	No tampering with histogram	P_2 tampered With his histogram
Local histogram h_1	$\{\{1, 2\}, \{10, 10\}\}$	$\{\{1, 2\}, \{10, 10\}\}$
Local histogram h_2	$\{\{-1, 1\}, \{10, 10\}\}$	$\{\{-2, 2\}, \{10, 10\}\}$
Gain of global split	0.45	0.85
P_1' split contribution ϕ_1	0.375	0.275
P_2' split contribution ϕ_2	0.075	0.575



3. Add the collected signatures to the histogram transaction, and commit the transaction when the number of signatures meets requirement.
7. When encountering a histogram transaction, verify the signature and check the number of signatures.

Figure 4: Blockchain-based histogram transactions working flow.

The above design is suitable for the overall architecture of our federated GBDT, which can detect the histogram with exaggerated contribution, and will not significantly affect the efficiency of the system. Firstly, to consider the data distribution of parties, we can avoid misjudging the correctly calculated update information as malicious by using the refitted histogram to a certain extent, and the stretched histogram can be easily discovered. For the efficiency of the verification scheme, the whole decentralized verification process is very similar to Fabric's high-level transaction flow [29]. The only difference is that the party uses the local data set under blockchain instead of simulating the execution of the smart contract. In addition, this process is also different from the processing method of Proof of Quality (PoQ) [8], where they suggest checking the quality of all models after block generation. If there is a malicious transaction, the block needs to be repackaged, which means retraining the whole GBDT model. In our scheme, orderers can filter out the transactions that are not recognized by the majority of participants when ordering transactions.

6. Implementation and Evaluation

6.1. Experiment Setup. We implement FV-tree based on LightGBM (https://github.com/microsoft/LightGBM). For PSD, we use a data-independent tree model. Each time of the PSD's node splitting, we randomly select a feature in the unused feature set and divide it according to the average of the global maximum and minimum values (the maximum and minimum values are specified in the task initialization transaction), we also treat the label as a feature. The maximum depth of PSD is 8, the maximum value of each leaf node is 500. Laplace noises are injected into the leaf nodes, where the privacy budget $\epsilon = 1$. For the GBDT model, the maximum depth of each tree is 8, the number of iterations is 500, the regulation parameter λ is set to 0.1, and the maximum number of bin in the feature histogram is 16 (more bin will bring higher accuracy, but this small accuracy difference is not significant for the federated GBDT framework).

We used three public datasets to evaluate our scheme (https://www.csie.ntu.edu.tw/cjlin/libsvmtools/datasets/), as shown in Table 4 And 75% of these datasets are used for training, the rest are used for testing. To allocate skewed local datasets, as the realistic scenario requires, we used the partition method of previous work [31], which allocates the datasets for each party according to the unbalanced ratio $\theta \in \{0, 1\}$. After allocation, half of parties got $(\theta * N_{\text{class0}})/M$ instances of class 0, and $((1 - \theta) * N_{\text{class0}})/M$ of instances of class 1, the other parties are just the opposite. This partition method well represents the data distribution in the federation scene. Specifically, in addition to label skewed, there is also feature skewed between local datasets [32]. As shown in Figure 5, we use kernel density estimation (KDE) to

Table 4: Dataset description.

Dataset	Cardinality	Dimension
a9a	32615	123
SUSY	1000000	18
HIGGS	1000000	28

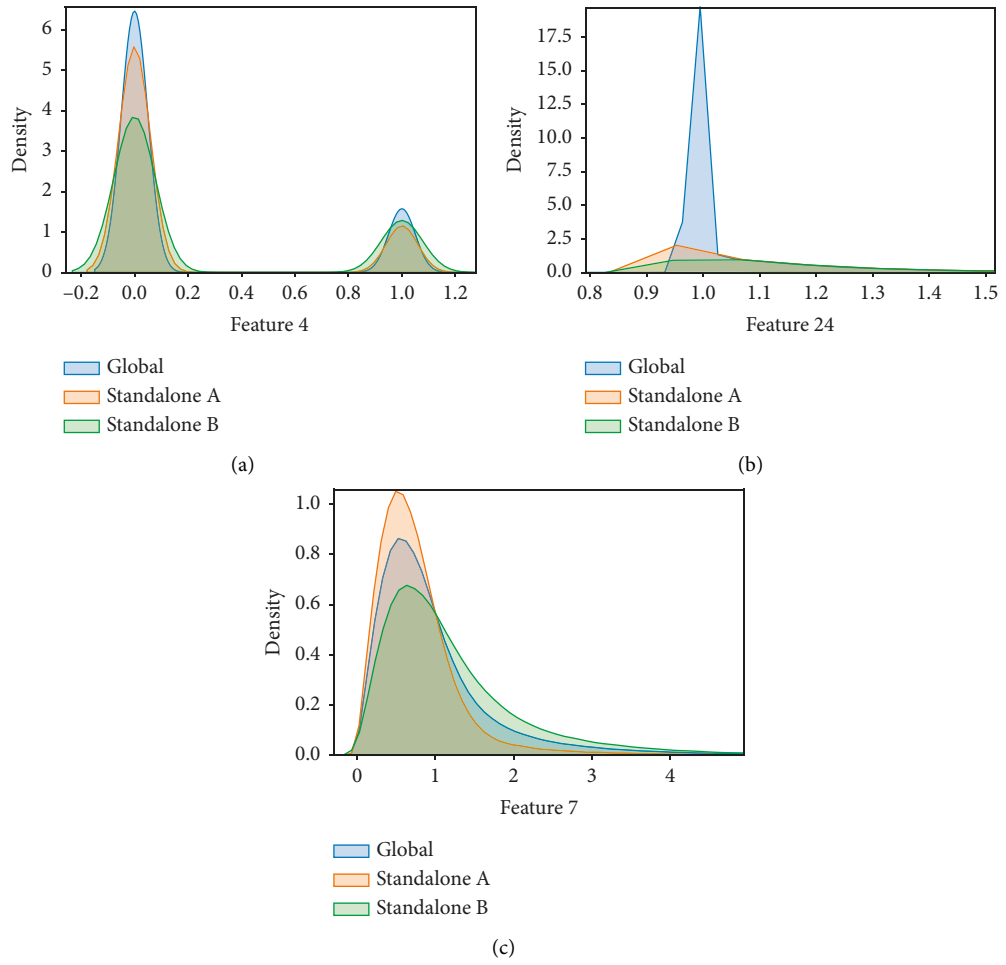


Figure 5: Compare feature distributions between local and global datasets by using Kernel density estimation (KDE). (a) a9a feature 4 (b) HIGGS feature 24 (c) SUSY feature 7.

intuitively show the skew degree of feature distribution between local and global datasets.

We compare our federated GBDT system with the other two frameworks: Standalone framework. This framework assumes that the parties training integration model only use their local dataset. The standalone setting shows the performance of the local training model of the party. In addition, there are two types of local dataset distributions in the unbalanced partition. We represent one part of the parties with more positive samples as Standalone A, and the other part as Standalone B. Centralized framework: This framework assumes that there is a trusted server accessing all parties' data, and uses global data to train the ensemble model without any privacy concerns. The centralized framework is high-precision, but it is hindered to implement in practice due to various restrictions. In addition, we also compare our scheme with other advanced federated GBDT frameworks in several same settings, such as TFL based on tree model communication and SimFL based on both tree model and gradients communication.

6.2. Experimental Results

6.2.1. Voting by Refitted Gradients. We first show the accuracy of FV-tree without considering differential privacy. To evaluate the effect of gradient refit, we compare FV-tree and PV-tree by convergence speed. Without losing generality, the number of parties is set to 4, and the ratio θ is set to 80%. The default parameters are used in all frameworks. The experimental results are shown in Figure 6. We can observe the following points. First, FV-tree performs better than PV-tree and Standalone models in all datasets. And because of

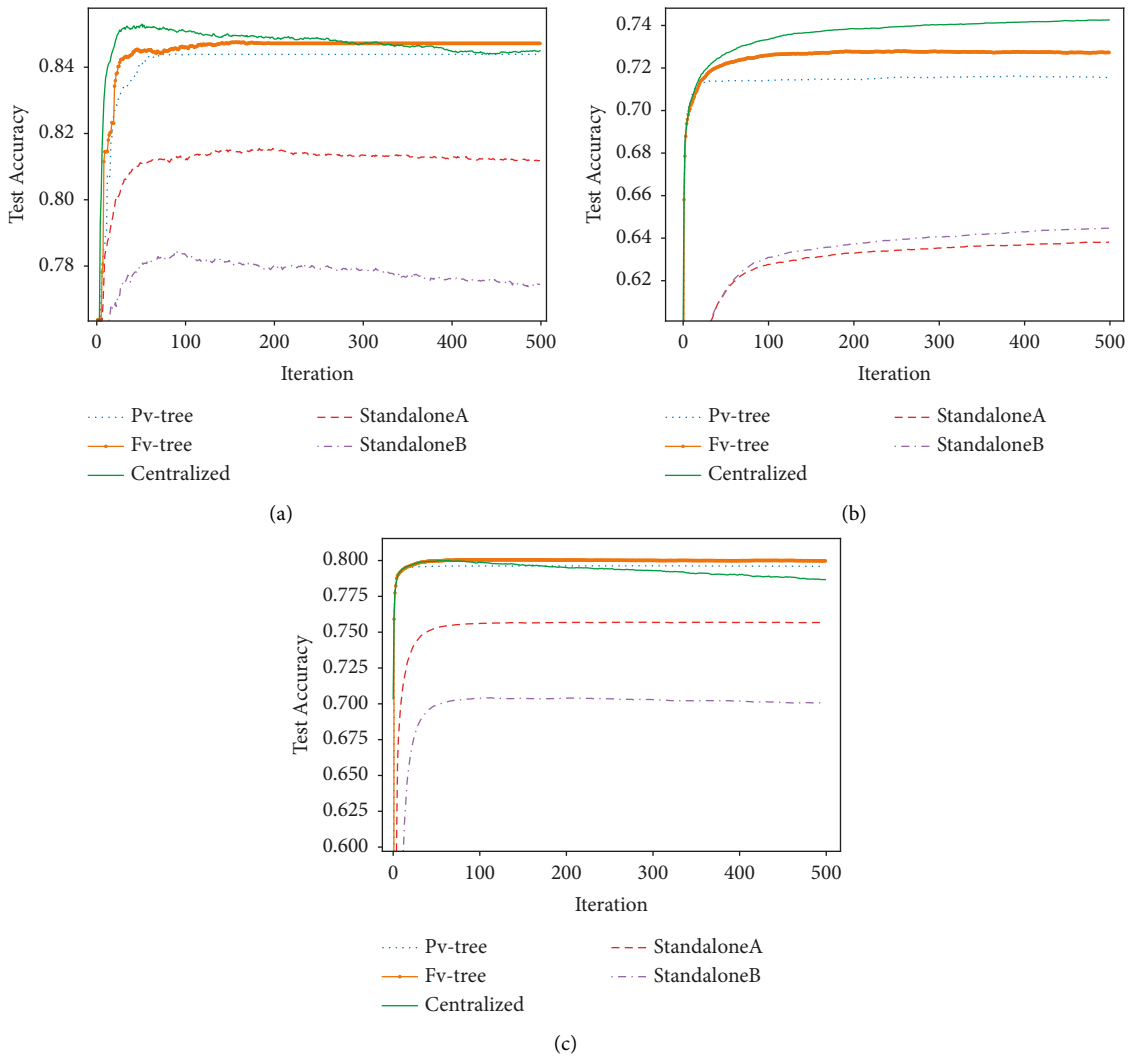


Figure 6: Comparison of the convergence speed, where the number of parties is set to 4, and the ratio θ is set to 80%. (a) a9a (b) HIGGS (c) SUSY.

the data skew, the accuracy of standalone mode is greatly reduced. This is because each party is affected by the data distribution bias in the learning process. And FV-tree uses a gradient to refit through PSDs, so it has a greater probability to select the most informative feature. Second, in the datasets a9a and SUSY, the centralized framework may lead to overfitting, while there is no such problem in the schemes based on FV-tree and PV-tree. Finally, the accuracy of PV-tree is significantly higher than the Standalone mode. This means that when considering differential privacy, we can get a tighter sensitivity without using the gradient refit.

6.2.2. The Impact of Unbalanced Ratio θ. To show the influence of different skew degrees on the FV-tree, we simply set the number of parties to 2. The experimental results are compared with SimFL, an advanced work without differential privacy. We observe the influence of different unbalanced distribution degrees on the prediction accuracy, as shown in Figure 7. We can observe that the accuracy of the standalone model decreases greatly with the skew of distribution. Secondly, although the accuracy of our framework and SimFL can be higher than local training when the unbalanced ratio is greater than 70%, FV-tree is much less affected than SimFL. This may be because the model accuracy is only affected by the feature selection in the FV-tree framework. While SimFL is affected by the feature selection and calculation of leaf weight. This means FV-tree is more suitable for skewed data distribution.

6.2.3. The Impact of the Number of Parties **M**. The number of different parties will also affect the accuracy of the model. We set a different number of parties when the unbalanced ratio θ is set to 80%. The experimental results are shown in Figure 8. Firstly, we can observe that FV-tree outperforms Standalone and SimFL in different number of parties settings, even the test error on dataset SUSY is less than that of over fitted centralized model. Secondly, with increasing number of parties, it does not have too much impact on FV-tree. This advantage may also come from the fact that FV-tree is not affected by the calculation of leaf weight.

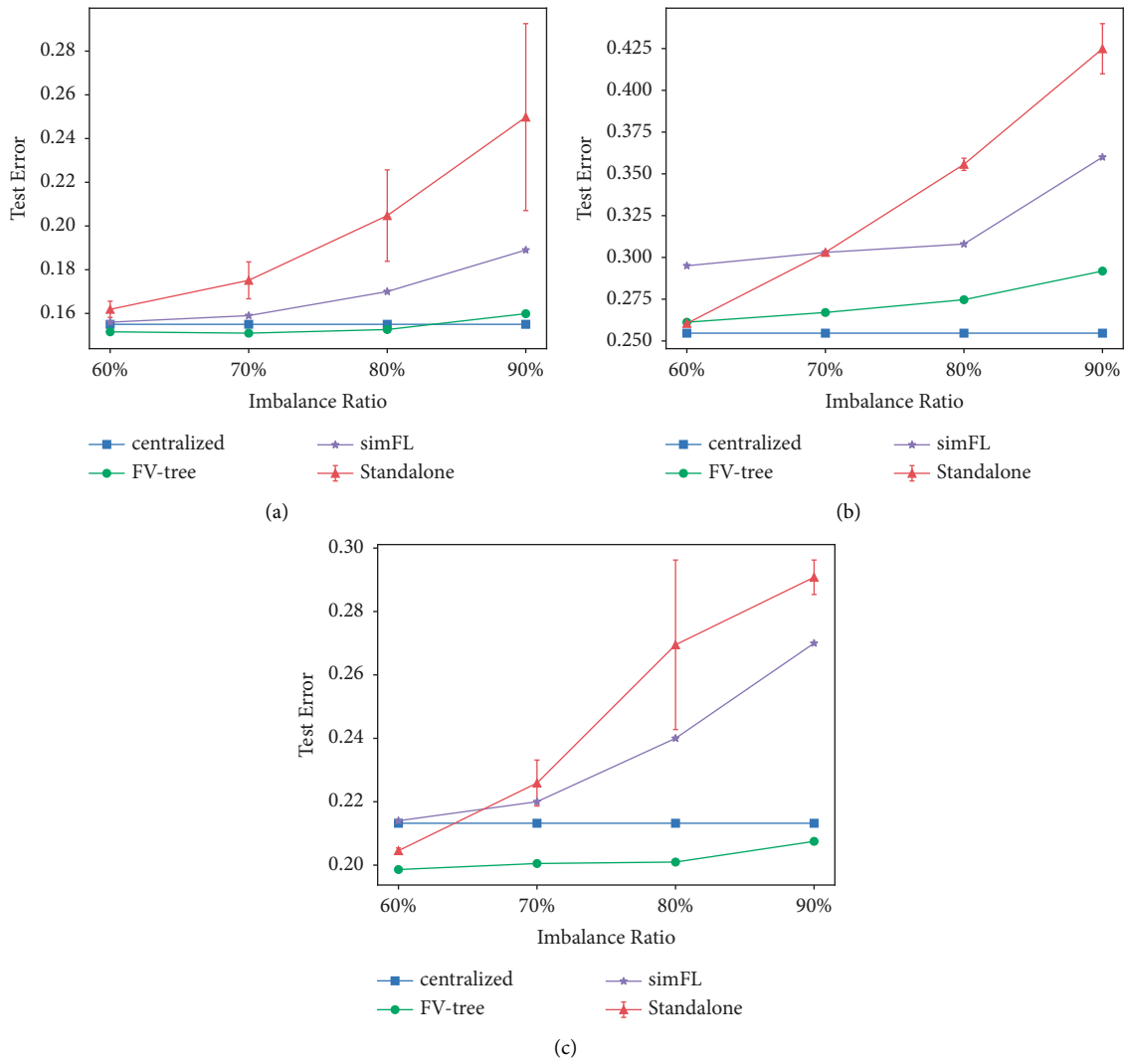


FIGURE 7: Comparison of the test errors given different unbalanced ratio θ, where the number of parties is set to 2. (a) a9a (b) HIGGS (c) SUSY.

6.2.4. The Impact of Differential Privacy. Based on the above experimental evaluation, FV-tree can achieve almost the same accuracy in distributed settings as centralized settings. Then, we test the FV-tree with differential privacy. Generally, we set the number of parties M to 4, and the unbalanced ratio θ is still set to 80%. To control the consumption of privacy budget, we set the maximum depth d of a single decision tree to 3. For dataset a9a, which has a small number of instances, is set as two ensembles, and each ensemble contains 20 trees. Dataset SUSY and HIGGS, which have a large number of instances, are set as one ensemble. To ensure a strict total privacy budget, PSD is not used. We evaluated the test error for different privacy budgets ϵ, as shown in Figure 9. Due to the randomness of differential privacy, we conducted 10 experiments and showed the maximum, minimum and average values (To be fair, the default parameter settings are still used in centralized and standalone models. Because there is no need to consider the consumption of the privacy budget, the iterations T and depth d can be increased to achieve higher accuracy).

We can observe that the accuracy of the FV-tree can still be higher than that of local training after using differential privacy on large-scale HIGGS and SUSY datasets. However, in the a9a dataset, due to the small amount of data, too much noise is added to the histogram, which reduces the accuracy of the model, but it is still comparable to the best training effect of local training. This means that our scheme has a good performance in large-scale datasets, and can meet the needs of practical applications.

7. Discussion

7.1. Accuracy Loss and Communication Overhead

7.1.1. Accuracy Loss. The accuracy loss of the FV-tree comes from the selection of the best split features. In the balanced data partition, we assume that the feature values of each dimension are i.i.d. uniform random variables, and assign the same number of instances to each party. Then, the possibility of selecting the best feature is as same as PV-tree

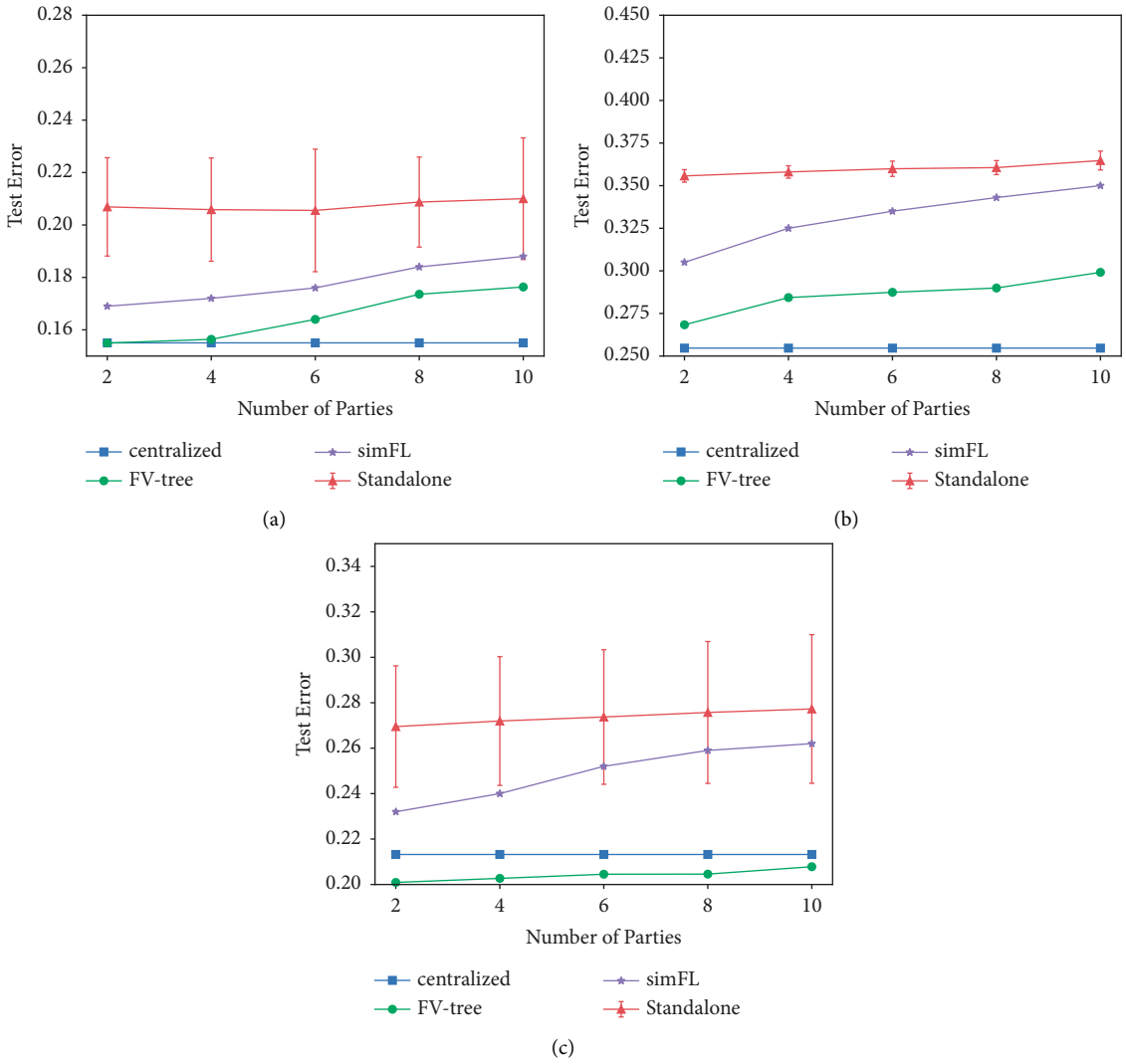


FIGURE 8: Comparison of the test errors given a different number of parties M, where the unbalanced ratio θ is set to 80%. (a) a9a (b) HIGGS (c) SUSY.

[33]. In the scenario of the skewed data partition, the experiment shows that FV-tree still has high accuracy. Moreover, in the case of significantly skewed data distribution, we can use the weight distribution calculated by PSDs to refit feature distribution, which can improve the possibility of selecting the best feature. However, the global distribution weight vector is used may cause high gradient values, which will make the privacy boundary loose. Under these circumstances, gradient cutting may be a feasible choice [34]. In addition, our scheme is not effective for small and continuous feature data sets. This obstacle is mainly due to adding a lot of noise to histograms, which reduces the effectiveness of the gradient histogram. Therefore, in small-scale dataset scenarios, we still need to use other federated GBDT frameworks.

7.1.2. Communication Overhead. The communication cost of our federated GBDT system is constant. First, in the pretraining phase, assuming that the depth of a PSD is d_{psd}, each party has to send one PSD model and receive $M-1$ PSD models, so the cost is $M(2^{d_{\text{psd}}}-1)$. In the training phase, assuming that there are T trees, and the depth of each tree is d, $2^{d-1}-1$ times node splitting is needed. Because each inner node needs to communicate three times, including one voting and two histograms uploading, where the voting communication is a real number. And the cost of a party sending $M-1$ times histogram to communicate histogram is $2(M-1)n_{\text{bin}}$. When two 2/3 of the signatures are received, the transaction can be sent. Let L_{sign} be the length of signature, then the cost of receiving the signatures is $2/3ML_{\text{sign}}$. In addition, they need to receive other parties' histograms and sign them, where the cost is $2(M-1)n_{\text{bin}}+(M-1)L_{\text{Sign}}$. Therefore, the communication overhead of a histogram aggregation is $(4M-3)N_{\text{bin}}+((7/3)M-1)L_{\text{sign}}$. Because there are T trees, the total communication overhead is $(2^{d-1}-1)[(4M-3)N_{\text{bin}}+((7/3)M-1)L_{\text{sign}}]T$, where d, M, L_{Sign}, T, N_{bin} are constants. So total communication cost of FV-tree is $\#O(1)$, which is less than other $\#O(|I_m|)$

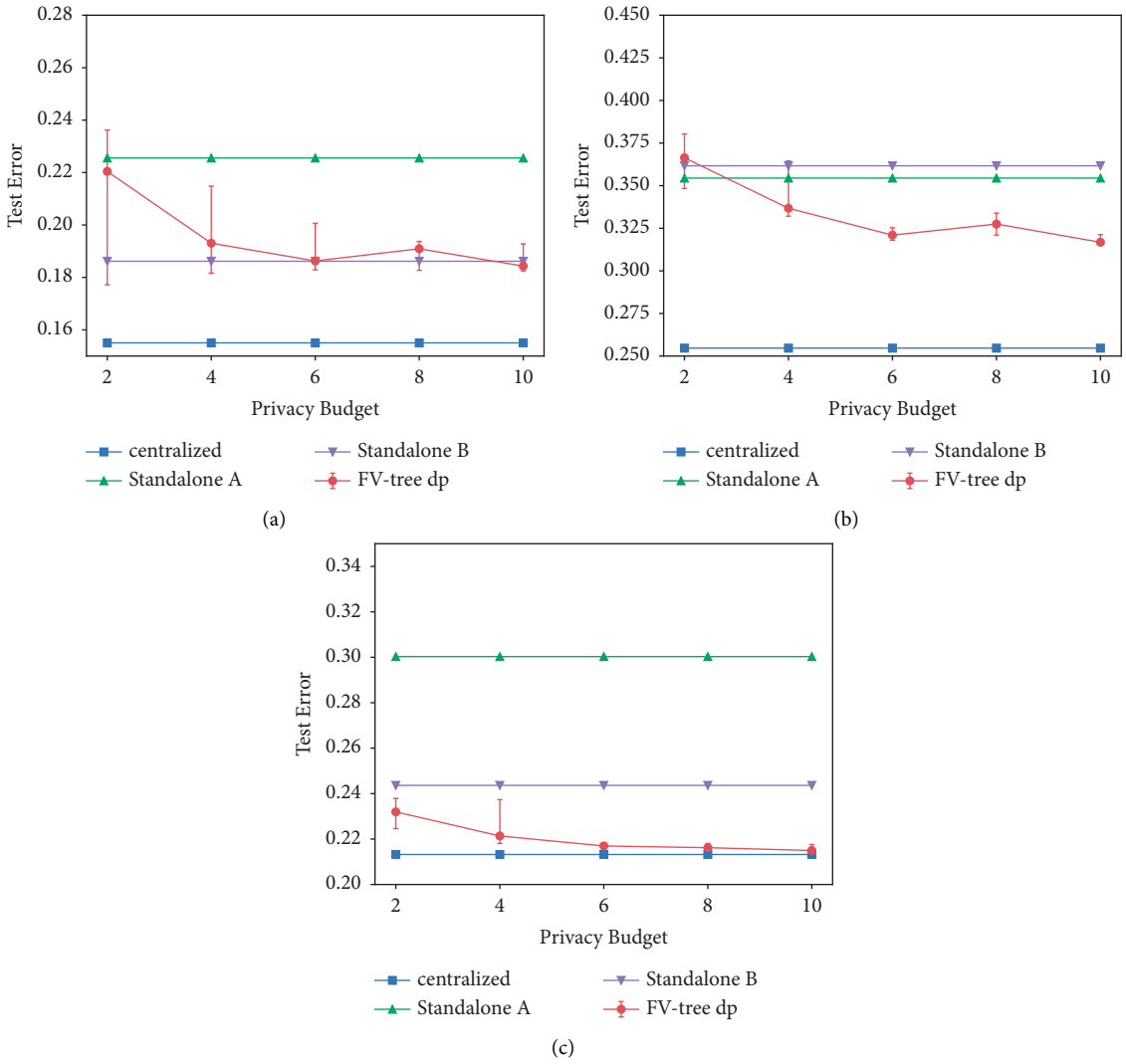


FIGURE 9: Comparison of the test errors given different total privacy budgets ϵ, the unbalanced ratio θ is set to 80%, where the maximum depth d of a single decision tree to 3. Dataset a9a is set as two ensembles, and each ensemble contains 20 trees. Dataset SUSY and HIGGS, are set as one ensemble with 50 trees. (a) a9a (b) HIGGS (c) SUSY.

federated GBDT framework [7]. In addition, the storage cost in the permissioned blockchain can reach an acceptable level to ensure fairness and tamper-proof.

7.2. Fairness and Efficiency. We regard the growth process of the decision tree as multiple cooperative games. Shapley value is used to measure the individual contribution in cooperation, the fairness of Shapley value is widely recognized. In our design, every node segmentation is fair, and the details can be obtained from Section 5-C. In addition, because the benefits obtained by the participants each time directly come from the gain value, it is also fair for the whole training process. For example, in the early stage of training, each split will produce a great gain, and each party will get more contribution value from it. On the other hand, the computational complexity of split Shapley value is acceptable. We can see only M is variable through (8), and in organization-cross federated scenes, M is usually a relatively small value. Besides, we do not need to traverse all the split points in histograms to calculate of U, because the global best split has been determined in split_q.

7.3. Security. It is assumed that all parties will aim at maximizing revenue and act honestly in the stage of voting characteristics because in the absence of any data of other parties, they can only choose the feature with the highest gain value to vote according to their real data to obtain voting awards. Similarly, in the phase of communicating gradient histogram, if the modified gradient histogram is detected, the histogram transaction cannot be published because of the need for a similarity test. Hence, a party can only get the histogram contribution reward if it publishes the real histograms.

Further, if there are malicious participants in the alliance, our system is still robust. Firstly, suppose that in the voting feature stage, if multiple malicious participants

conspire to select a feature f' with less gain to enter the global candidate features. At the same time, as long as one honest party selects another feature f, f' is still likely not to be the split point, because the gain value of f may be greater than it. On the contrary, if the gain value of f is less than f', it means that, f' is a good segmentation feature, and dividing nodes according to f', f' will not cause great harm to the model. Secondly, in the histogram aggregation stage, because the gradient histogram of the malicious party needs to be verified by two-thirds of the parties, it is necessary for the malicious parties involved in the conspiracy to reach two-thirds of the total number to make the histogram of the damage model accepted by the federation.

8. Conclusion

In this paper, we aim to present a closed-loop federated GBDT system. In our scheme, each party can get a good performance model and be allocated to a fair contribution index. At the same time, with the help of blockchain and decentralized verification mechanism, the calculation of the contribution index will remain secure, the results cannot be tampered with, and provide additional functions such as delayed payment or audit for any need. Besides, the communication overhead is constant which enables our method to fit federated GBDT tasks with large-scale datasets very well. Due to privacy constraints, this scheme may not be suitable for small-scale data sets, which is the direction we plan to study in our future work. [35].

Data Availability

The experiment source data used to support the findings of this study have been deposited in the https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/. And the experimental results data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. U21A20474), the Guangxi "Bagui Scholar" Teams for Innovation and Research Project, the Guangxi Science and Technology Plan Projects (no.AD20159039), the Guangxi Young and Middle-aged Ability Improvement Project (no. 2020KY02032), and the Innovation Project of Guangxi Graduate Education (no. YCBZ2021038).

References

[1] H. B. McMahan, E. Moore, D. Ramage, and S. Hampson, B. A. Y. Arcas, Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, pp. 1273–1282, PMLR, 2017.

[2] Q. Li, Z. Wen, Z. Wu et al., "A Survey on Federated Learning Systems: Vision, Hype and Reality for Data Privacy and protection," 2019, https://arxiv.org/abs/1907.09693.

[3] T. Chen and C. Guestrin, "Xgboost: a scalable tree boosting system," in *Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, San Francisco California USA, March 2016.

[4] G. Ke, Q. Meng, T. Finley et al., "Lightgbm: a highly efficient gradient boosting decision tree," *Advances in Neural Information Processing Systems*, vol. 30, pp. 3146–3154, 2017.

[5] A. Callens, D. Morichon, S. Abadie, M. Delpey, and B. Liquet, "Using random forest and gradient boosting trees to improve wave forecast at a specific location," *Applied Ocean Research*, vol. 104, Article ID 102339, 2020.

[6] L. Zhao, L. Ni, S. Hu et al., "Inprivate digging: enabling tree-based distributed data mining with differential privacy," in *Proceedings of the IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pp. 2087–2095, IEEE, Honolulu, HI, USA, April 2018.

[7] Q. Li, Z. Wen, and B. He, "Practical federated gradient boosting decision trees," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 4642–4649, NY, USA, February 2020.

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

[9] Y. Zhan, J. Zhang, Z. Hong, L. Wu, P. Li, and S. Guo, "A survey of incentive mechanism design for federated learning," *IEEE Transactions on Emerging Topics in Computing*, vol. 10, no. 2, pp. 1035–1044, 2022.

[10] L. Zhao, Q. Wang, C. Wang, Q. Li, C. Shen, and B. Feng, "Veriml: enabling integrity assurances and fair payments for machine learning as a service," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 10, pp. 2524–2540, 2021.

[11] J. Mateo, J. Rius-Peris, A. Maraña-Pérez, A. Valiente-Armero, and A. Torres, "Extreme gradient boosting machine learning method for predicting medical treatment in patients with acute bronchiolitis," *Biocybernetics and Biomedical Engineering*, vol. 41, no. 2, pp. 792–801, 2021.

[12] S. Lee, T. P. Vo, H.-T. Thai, J. Lee, and V. Patel, "Strength prediction of concrete-filled steel tubular columns using categorical gradient boosting algorithm," *Engineering Structures*, vol. 238, Article ID 112109, 2021.

[13] R. Kufrin, "Decision Trees on Parallel Processors," *Machine Intelligence and Pattern Recognition*, vol. 20, pp. 279–306, 1997.

[14] Q. Meng, G. Ke, T. Wang et al., "A Communication-Efficient Parallel Algorithm for Decision Tree," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS, Barcelona, Spain, December 2016.

[15] R. H. L. Sim, Y. Zhang, M. C. Chan, and B. K. H. Low, "Collaborative machine learning with incentive-aware model rewards," in *Proceedings of the 37th International Conference on Machine Learning*, pp. 8927–8936, PMLR, New York City, NY, USA, July 2020.

[16] Y. Zhao, J. Zhao, L. Jiang, R. Tan, and D. Niyato, "Mobile Edge Computing, Blockchain and Reputation-Based Crowdsourcing Iot Federated Learning: A Secure, Decentralized and Privacy-Preserving System," pp. 2327–4662, 2019, https://arxiv.org/abs/1906.10893%20.

[17] L. U. Khan, S. R. Pandey, N. H. Tran et al., "Federated learning for edge networks: resource optimization and incentive

mechanism," *IEEE Communications Magazine*, vol. 58, no. 10, pp. 88–93, 2020.

[18] H. W. Kuhn and A. W. Tucker, *Contributions to the Theory of Games*, Princeton University Press, Princeton, New Jersey, 1953.

[19] R. Jia, D. Dao, B. Wang et al., "Towards efficient data valuation based on the shapley value," in *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, pp. 1167–1176, PMLR, New York City, NY, USA, February 2019.

[20] A. Ghorbani, M. Kim, and J. Zou, "A distributional framework for data valuation," in *Proceedings of the International Conference on Machine Learning*, pp. 3535–3544, PMLR, New York City, NY, USA, February 2020.

[21] T. Song, Y. Tong, and S. Wei, "Profit allocation for federated learning," in *Proceedings of the 2019 IEEE International Conference on Big Data (Big Data)*, pp. 2577–2586, IEEE, Los Angeles, CA, USA, December 2019.

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

[23] L. Lyu, J. Yu, K. Nandakumar et al., "Towards fair and privacy-preserving federated deep models," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 11, pp. 2524–2541, 2020.

[24] J. Weng, J. Weng, J. Zhang, M. Li, Y. Zhang, and W. Luo, "Deepchain: Auditable and Privacy-Preserving Deep Learning with Blockchain-Based Incentive," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 5, 2019.

[25] S. Si, H. Zhang, S. S. Keerthi, D. Mahajan, I. S. Dhillon, and C.-J. Hsieh, "Gradient boosted decision trees for high dimensional sparse output," in *Proceedings of the International Conference on Machine Learning*, pp. 3182–3190, PMLR, New York City, NY, USA, August 2017.

[26] K. Alsabti, S. Ranka, and V. Singh, "Clouds: a decision tree classifier for large datasets," in *Proceedings of the 4th knowledge discovery and data mining conference*, AAAI Press, New York, NY, August 1998.

[27] G. Cormode, C. Procopiuc, D. Srivastava, E. Shen, and T. Yu, "Differentially private spatial decompositions," in *Proceedings of the 2012 IEEE 28th International Conference on Data Engineering*, pp. 20–31, IEEE Computer Society, Los Alamitos, CA, USA, April 2012.

[28] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," *Decentralized Business Review*, Article ID 21260, 2008.

[29] E. Androulaki, A. Barger, V. Bortnikov et al., "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*, pp. 1–15, Porto, Portugal, April 2018.

[30] Y. Li, C. Chen, N. Liu, H. Huang, Z. Zheng, and Q. Yan, "A blockchain-based decentralized federated learning framework with committee consensus," *IEEE Network*, vol. 35, no. 1, pp. 234–241, 2021.

[31] M. Yurochkin, M. Agarwal, S. Ghosh, K. H. Greenewald, T. N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks,"vol. 97, pp. 7252–7261, in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, pp. 7252–7261, ICML, Long Beach, California, USA, June 2019.

[32] K. Hsieh, A. Phanishayee, O. Mutlu, and P. Gibbons, "The non-iid data quagmire of decentralized machine learning," in *Proceedings of the International Conference on Machine Learning*, pp. 4387–4398, PMLR, Shanghai, China, November 2020.

[33] Q. Meng, G. Ke, T. Wang et al., "A Communication-Efficient Parallel Algorithm for Decision Tree," 2016, https://arxiv.org/abs/1611.01276.

[34] Q. Li, Z. Wu, Z. Wen, and B. He, "Privacy-preserving gradient boosting decision trees," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 784–791, NY, USA, February 2020.

[35] C. Dwork, "Differential privacy: A Survey of Results," *Theory and Applications of Models of Computation*, vol. 4978, pp. 1–19, 2008.