Hindawi
International Journal of Intelligent Systems
Volume 2023, Article ID 6692995, 14 pages
https://doi.org/10.1155/2023/6692995

Research Article

Enhancing Fairness in Federated Learning: A Contribution-Based Differentiated Model Approach

Tao Wan,[1] **Xianqing Deng**,[1] **Weichuan Liao**,[2] **and Nan Jiang**[1]

[1]*School of Information Engineer, East China Jiaotong University, Nanchang 330013, China*
[2]*School of Science, East China Jiaotong University, Nanchang 330013, China*

Correspondence should be addressed to Tao Wan; wantao217@163.com

Received 1 July 2023; Revised 17 October 2023; Accepted 28 October 2023; Published 14 November 2023

Academic Editor: Yu-an Tan

Federated learning (FL) has emerged as a promising framework for collaborative machine learning, allowing the training of machine learning models on distributed devices without centralizing sensitive data. However, FL falls short in terms of fairness, as each client receives the same model regardless of their individual contributions. This unfairness discourages active client participation in FL. To address this challenge, we propose a contribution-based differentiated global model mechanism. Specifically, we introduce the contribution score as a metric to assess client contributions in FL and utilize deep Q-networks (DQN) to dynamically update the contribution scores. Subsequently, we allocate clients to different clusters based on their contributions by using a clustering algorithm, where each cluster is associated with a distinct global model. This mechanism encourages clients to make greater contributions for improved global models. Experimental results confirm the effectiveness of our approach in enhancing fairness in FL.

1. Introduction

Federated learning (FL) [1] has gained considerable attention due to its effective approach in addressing the challenges associated with training models using distributed data. In FL, each client collaboratively trains a global model while ensuring the privacy of their local data. Specifically, a central server distributes the global model parameters to individual clients, who utilize their respective data for training local models. Once the training process is completed, the local model parameters are transmitted back to the server, instead of the raw data. Subsequently, the server aggregates the received local models to generate an updated global model. The aforementioned process enables FL to effectively safeguard the security of user data.

Research in the field of FL has primarily focused on protecting user data privacy [2] and reducing communication costs while overlooking the issue of fairness. In FL, clients with varying levels of contribution ultimately obtain the same global model [3]. Such inequity in the distribution of benefits can lead to dissatisfaction among high-contributing clients, subsequently impacting their motivation to actively participate in FL and ultimately impeding the sustainable development of FL. To address this problem, it is necessary to establish a reasonable criterion for evaluating the contribution of each client. Furthermore, the development of a fair allocation mechanism that satisfies all stakeholders is crucial [4].

Several methods have been proposed for fairness research in FL. In [5], the authors proposed a smart contract-based data-model provenance registry to enable accountability and a weighted fair data sampler algorithm to enhance fairness in FL. In [6], an algorithm was presented to achieve more fairness and accuracy in horizontal federated learning (FedFa). They proposed an appropriate weight selection algorithm that combines the information quantity of training accuracy and training frequency to measure the weights, thereby assisting clients in aggregating at the server with a more equitable weighting. Another approach, presented in [7], is the pairwise correlated agreement method, which utilizes the idea of peer prediction to evaluate user contributions in FL without requiring a test dataset. This approach was used to design a new incentive mechanism that ensures truthfulness in FL. The work in [8] proposed the federated learning incentivizer (FLI) payoff-sharing scheme. The scheme dynamically divides a given budget in a context-aware manner among data owners in a federation by jointly maximizing the collective utility while minimizing the inequality among the data owners. Additionally, the authors proposed the primal-dual greedy auction mechanism in [9], which utilizes auction theory to evaluate the resource conditions of clients and provide corresponding incentives.

Previous studies have employed trust models [10] or incentive mechanisms to ensure fairness in FL. The server provides corresponding rewards to clients based on their reputation or contributions. However, it becomes challenging for the server to assess the real-time contributions made by each client to the FL training process [11].

Reinforcement learning (RL) [12] is a prominent field in machine learning that focuses on determining optimal actions for agents based on their interaction with the environment, aiming to maximize expected rewards. Alongside supervised and unsupervised learning, RL forms one of the fundamental branches of machine learning and has witnessed continuous advancements over the past few decades. However, traditional RL methods often encounter difficulties when confronted with complex real-world problems. Deep reinforcement learning (DRL) [13] has emerged as a promising approach that combines the powerful perception and comprehension capabilities of deep learning with the decision-making abilities of RL. This integration allows DRL to tackle complex real-world problems effectively, making RL more practical across diverse domains. Moreover, DRL has been successfully applied in the FL domain to address various challenges, including reducing training completion time under resource constraints [14], improving model aggregation rate, and minimizing communication costs [15], as well as maximizing the social welfare of the FL services market [16].

In this article, we propose an evaluation metric called the contribution score to quantify the contributions made by clients to FL. As the FL training iterations progress, the contributions of clients dynamically change, necessitating an adaptive mechanism to account for these variations. This is where deep Q-network (DQN) [13] comes into play, as it effectively addresses problems in discrete action spaces by leveraging deep neural networks to learn the value function that maps states and actions to expected rewards. By employing techniques such as random sampling from a replay buffer and utilizing a fixed target network to reduce the instability of Q-value estimations, DQN provides a robust and efficient solution for dynamically updating the contribution scores.

In this work, we propose a contribution-based differentiated global model mechanism to ensure fairness in FL. We use the contribution scores to measure the contributions of clients to FL. Then, based on these scores, we employ clustering algorithms to divide clients into different clusters, with each cluster corresponding to a specific global model. After the completion of FL iterations, clients receive global models of different performance levels based on their assigned clusters. This mechanism allows clients who make greater contributions to obtain higher-performing global models, thus incentivizing their active participation in FL. Furthermore, DQN dynamically updates the contribution score of each client, ensuring that client grouping is not static. The dynamic updating of contribution scores enables us to adapt to changes in client behavior over time and ensure a fair assessment of client contributions. The main contributions of this work are listed as follows:

(i) We design a contribution scores updating method based on DQN. This method fairly evaluates the contributions of clients to FL and updates their contribution scores.
(ii) We propose a contribution-based differentiated global model mechanism to address the fairness challenge in FL. This mechanism ensures that high-contributing clients obtain better global models compared to low-contributing clients at the end of FL training.
(iii) The final experimental results confirm the effectiveness of our method in improving the fairness of FL, thereby motivating clients to participate in the FL process.

This article is structured as follows. In Section 2, we provide an overview of the existing literature on FL fairness. The system model is presented in Section 3. Subsequently, we explain our proposed contribution scores updating method based on DQN in Section 4. Following that, the performance results of our proposed model are analyzed in Section 5. Finally, we present our concluding remarks in Section 6.

2. Related Work

There are recent research activities related to fairness in FL. The significant and the most relevant publications are presented in three categories, contribution and reputation as metrics, contribution as an incentive in the FL, and contribution update methods.

2.1. Contribution and Reputation as Metrics. Various techniques and methods have been proposed to address fairness issues in FL. These approaches utilized contribution or reputation as metrics to assess client performance. In [17], the authors designed a blockchain-based FL system where reputation validators on the blockchain employed the multi-KRUM algorithm to compute reputation and evaluate dissatisfactory updates. Validators will increase the reputation of a client upon acceptance of its update, while the reputation is diminished in case of rejection. In [18], the authors proposed a blockchain-based reliable FL reputation system to select trustworthy workers. The reputation of workers was calculated by using a multiweight subjective logic model based on their historical performance and recommendations from other workers. However, there is a potential for workers to receive malicious ratings or unfair evaluations. In [19], the authors proposed an FL incentive mechanism based on reputation and reverse auction theory, where each client bids for the opportunity to participate in FL. Their reputation reflects their reliability and data quality. However, this method is only applicable to horizontal FL and may result in high model complexity.

There are other approaches that utilize contribution to ensure fairness in FL. These methods assess the contributions of clients by considering multiple factors. In [20], the authors proposed a blockchain-based FL framework and a protocol to transparently evaluate the contribution of each participant. This framework protects privacy for all parties in the model-building phase and transparently evaluates contributions based on the model updates. In [21], the authors proposed the guided truncation gradient Shapley (GTG-Shapley) approach. It reconstructs FL models from gradient updates for calculating Shapley value instead of repeatedly training with different combinations of FL participants. In [22], the authors proposed a lightweight multidimensional contribution method based on progressive computation. This approach evaluates the contribution of clients by using the performance gain as an indicator, which reflects the degree to which the client model improves the global model in each round. This approach effectively reduces the traffic and computational overhead. However, it requires that clients make positive contributions to the final global model performance in each iteration.

2.2. Contribution as an Incentive in FL.

2.2. Contribution as an Incentive in FL. Contribution scores are widely used in FL to evaluate the contribution of clients to FL training. An effective contribution mechanism can not only optimize the utility of the FL system but also motivate high-quality participants to join the FL. Several incentive mechanisms have been proposed to integrate fairness into FL. In [23], the authors presented a system called federated learning with quality awareness (FAIR). This system utilizes historical learning records to estimate the learning quality of users, considering the freshness of the records. Furthermore, FAIR employs a reverse auction as an incentive mechanism to incentivize the participation of high-quality learning users. In [24], the authors proposed a decentralized fair and privacy-preserving deep learning (FPPDL) framework. This framework incentivizes participants by rewarding them with points based on the amount of gradient information they upload. Participants can subsequently utilize these points to acquire gradient information from other participants. Another incentive mechanism called FMore was proposed in [25], which employs a multidimensional procurement auction of K winners to select participants based on their scores. The scores are determined by the bidding amount and the quality of the model update. This lightweight mechanism encourages high-quality edge nodes to participate in training and ultimately improve the performance of FL. In [26], the authors proposed an incentive mechanism called fairness-aware incentive mechanism for federated learning (FedFAIM), which ensures reward fairness by utilizing an effective Shapley value-based contribution assessment method and a novel reward allocation method based on reputation and distribution of local and global gradients. It is worth noting that these approaches primarily focus on rewarding high-contributing users without adequately considering penalties for malicious users and free-riders. Additionally, participants may provide false gradient information to obtain better incentives in certain situations, which will undermine the fairness and effectiveness of the FL system.

2.3. Contribution Update Methods. In the literature, there are various technologies proposed to measure and update the contribution scores of clients in FL. In [27], the authors introduced the concept of the contribution index, which quantifies the contribution of data providers by considering local datasets and machine learning algorithms. In [28], the authors proposed a new measure called completed federated Shapley value, which offers a fair evaluation of data quality. However, it is only applicable to horizontal FL.

The dynamic updating of contribution scores is essential to accommodate the changing environment of FL. In [29], the authors proposed a DRL-based reputation mechanism for optimal selection and evaluation of reliable FL clients. The reputation of each client is calculated by using the subjective logic model. Clients whose reputation exceeds the reputation threshold are granted the opportunity to participate in FL training. The reputation threshold is dynamically updated through DRL. In [30], the authors proposed a DRL-based incentive mechanism that can automatically learn the optimal pricing strategy in a dynamic network environment. By actively interacting with the environment, the DRL algorithm improves its strategy based on accumulated experience, ultimately approaching an optimal solution. In [31], the authors modeled the total reward of the server and the total revenue of the edge nodes as a Stackelberg game, solved the Nash equilibrium to obtain the optimal solution, and used the DRL algorithm to dynamically adjust the incentive strategy to optimize the profits of all parties. However, this approach relies on the assumption of independent and identically distributed data among edge nodes, which is unrealistic in real FL environments.

3. System Model and Problem Formulation

To address the challenges in FL fairness, we design a framework based on DQN for optimal contribution evaluation and fair reward distribution in FL, as illustrated in Figure 1. Based on the contributions of clients, the server employs clustering algorithms to categorize the clients into three clusters. Each client within a cluster downloads the corresponding global model from the parameter server and utilizes its local dataset to train the model. Once the predetermined number of training rounds is reached, the trained local model parameters are uploaded to the server for aggregation. We utilize contribution scores to quantify the contributions of clients to FL and employ the DQN algorithm to automatically adjust contribution scores. The adaptive updating of contribution scores ensures that high-

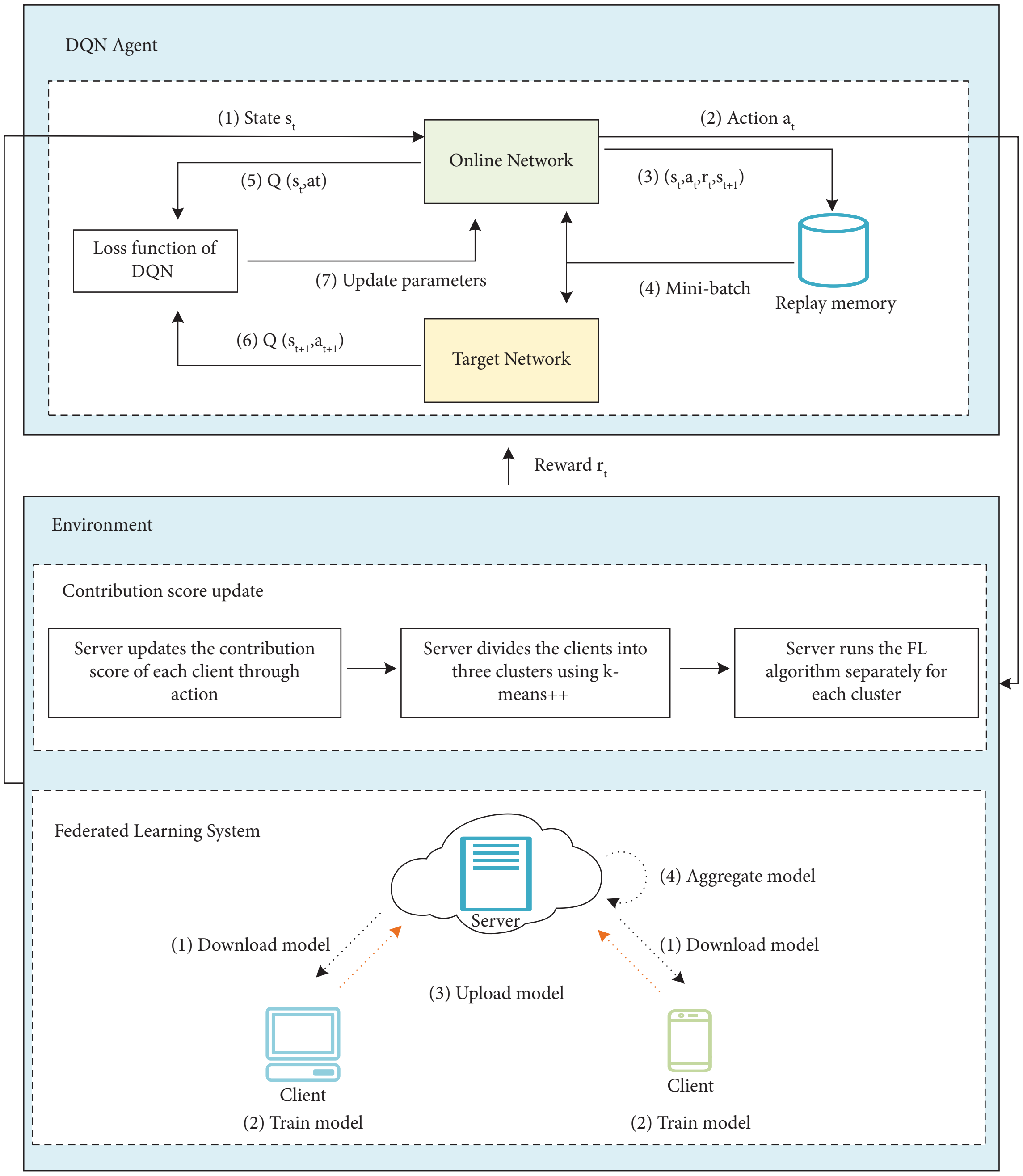

DQN Agent
(1) State s_t
(2) Action a_t
Online Network
(5) Q (s_t,at)
(3) (s_t,a_t,r_t,s_{t+1})
Loss function of DQN
(7) Update parameters
(4) Mini-batch
Replay memory
(6) Q (s_{t+1},a_{t+1})
Target Network
Reward r_t
Environment
Contribution score update
Server updates the contribution score of each client through action
Server divides the clients into three clusters using k-means++
Server runs the FL algorithm separately for each cluster
Federated Learning System
Server
(4) Aggregate model
(1) Download model
(1) Download model
(3) Upload model
Client
Client
(2) Train model
(2) Train model

FIGURE 1: DQN-based contribution scores updating method for federated learning.

contributing clients receive greater scores, thereby enhancing the fairness of FL. The following sections will discuss our proposed DQN-based fair FL solution.

3.1. Federated Learning Model. In our FL model, we divide all clients into three clusters. The clients within each cluster synchronously run the FL algorithm, resulting in three separate FL processes. We denote K as the total number of clients. The objective of FL is to optimize the global loss function $f(w)$ by minimizing the weighted average of each client's local loss function $f(w^k)$, where the contribution score cs^k of each client serves as the weight for the aggregation process.

$$f(w) = \sum_{k=1}^{K} \frac{\mathrm{cs}^k}{\mathrm{CS}} f\left(w^k\right), \tag{1}$$

where CS denotes the aggregate contribution score from K clients, which can be calculated as $CS = \sum_{k=1}^{K} \mathrm{cs}^k$, and w^k represents the local model parameters of the k-th client.

Various optimization algorithms can be used to train an FL model. In this study, we adopt the stochastic gradient descent (SGD) algorithm as our FL training algorithm. The SGD algorithm iteratively selects a batch of training examples to compute the gradients with respect to the current model parameters w and updates the parameters in the direction that minimizes the loss function $f(w)$ [1]. Hence, the objective is to discover the optimal model parameters w^* that minimize $f(w)$.

$$w^* = \operatorname{argmin} f(w). \tag{2}$$

At the beginning of each FL iteration t, the server selects a subset of clients to participate in the task and sends the current global model parameters w_t to the selected clients. The selection of clients can be random or based on specific requirements set by the server, such as data size or computational resources [32]. The selected clients download the global model parameters w_t and use their local data samples to calculate local model updates denoted by w_t^k. To achieve this, each client performs multiple iterations of SGD on its local data samples to compute the gradient g^k. The local update w_t^k of each client is computed as follows:

$$w_t^k = w_t - \eta^k g^k, \tag{3}$$

where η^k is the learning rate used to adjust the impact of the gradient on the local model parameters.

After receiving local model updates from the selected clients, the server aggregates the updates to obtain a new global model w_{t+1}. The aggregation process can be performed using various techniques such as federated averaging (FedAvg) [1] or FedProx [33]. The updated global model w_{t+1} is calculated as follows:

$$w_{t+1} = \sum_{k=1}^{K} \frac{\mathrm{cs}^k}{\mathrm{CS}} w_t^k. \tag{4}$$

In traditional FL, the clients and server repeat the above process in subsequent training iterations until the global model achieves a specific accuracy or a predetermined number of iterations set by the server. Clients with a larger amount of data typically exhibit higher local model accuracy, which can aid in the convergence of the local model w^k in (3) and the global model w in (4) towards the target value with fewer iterations. These high-contributing clients play a crucial role in enhancing the convergence speed and overall performance of the FL process, demonstrating the importance of their active involvement in FL.

3.2. Problem Formulation. Federated learning is a decentralized machine learning approach in which multiple parties collaborate to train a shared machine learning model using their respective local datasets without sharing raw data. However, the decentralized nature also brings about the threat of free-riders [34].

In this work, we consider the varying contributions of individual clients to the global model training process by introducing a metric known as the contribution score. The contribution score allows us to quantify the level of contribution provided by each client. Utilizing this metric, the clients are categorized into three clusters: high-contributing clients, ordinary clients, and free-riders. High-contributing clients typically represent companies or organizations with abundant data and computing resources, which play a crucial role in contributing to the global model. Ordinary clients possess a moderate amount of data and may also make valuable contributions. In contrast, free-riders refer to clients who consume network bandwidth and computing resources without actively contributing their own data. The global model w_t is updated as follows:

$$w_t = \sum_{k=1}^{K} \frac{\mathrm{cs}_1^k}{\mathrm{CS}} w_t^k + \frac{\mathrm{cs}_2^k}{\mathrm{CS}} w_t^k + \frac{\mathrm{cs}_3^k}{\mathrm{CS}} w_t^k, \tag{5}$$

where cs_i^k represents the contribution score of the k-th client in the i-th cluster and i can take the values of 1, 2, or 3. CS is the total contribution score from all K clients and is defined as $\mathrm{CS} = \sum_{k=1}^{K} \mathrm{cs}_1^k + \mathrm{cs}_2^k + \mathrm{cs}_3^k$.

From (5), we can see that the global model of federated learning is an aggregation of local models from three types of clients. The formula assigns different weights to each local model based on contribution scores. This is where the problem arises. If we give higher weights to free-riders, then their local models will have a greater impact on the global model, but their data volume, data quality, and data diversity are all very low, which will reduce the performance and accuracy of the global model. At the same time, this will also make high-contributing clients feel unfair, because they have invested more resources and computing power, but have received less rewards. If we give too high weights to high-contributing clients, then their local models will dominate the global model, but their data features may differ greatly from other clients, which will cause the global model to overfit the data of high-contributing clients and ignore the data of other clients. This will reduce the generalization ability of the global model, which is its performance on unknown or new data.

To address the issue of free-riders, effective incentive mechanisms should be integrated into the FL framework to promote active client participation and data contribution. These measures may include reward systems or reputation evaluation mechanisms. A comprehensive FL system should consider the participation of all types of clients, incentivizing high-contributing clients while also penalizing free-riders who do not contribute.

3.3. Clustering Based on Contribution Scores. In order to address the fairness issue in FL, a differentiated global model approach is proposed in this paper, which allows for different clients to obtain different models. Specifically, we use a contribution score metric to measure the contribution made by each client to the global model training process. Based on the contribution scores, clients are divided into three clusters: high-contributing clients, ordinary clients, and free-riders. FL algorithm is run separately for each cluster, and clients are assigned different models based on their cluster membership. This differential model approach is designed to balance the contributions of clients and incentivize more participation, thereby improving the fairness and effectiveness of FL.

Contributed scores are stored on the server instead of being maintained by the clients. This prevents malicious clients from tampering with their scores. FL applies a homomorphic encryption algorithm to protect the security of the models [35]. Homomorphic encryption is a special kind of encryption technique that can perform various arithmetic operations on ciphertexts without decrypting them. Moreover, the clients in FL only transmit model parameters or gradients instead of raw data. These measures effectively protect the privacy and security of data in FL.

To effectively classify clients into three clusters, we adopt the k-means++ algorithm [36]. The algorithm starts by selecting the first cluster center cc1 uniformly at random from the data set. To select subsequent cluster centers cc_i, the algorithm computes the squared distance $D(x)^2$ from each data point x to the closest cluster center that has already been chosen and chooses a new center with probability proportional to this squared distance. This ensures that new centers are chosen far from existing centers, which improves the quality of clustering. The algorithm then assigns each data point to the nearest cluster center and computes the new cluster centers. These steps are repeated until convergence. The calculation method for the squared distance $D(x)^2$ is as follows [36]:

$$D(x)^2 = \min_{\mathrm{cc}\in \mathrm{CC}} |x-\mathrm{cc}|^2, \tag{6}$$

where CC represents the set of currently selected cluster centers and $|x-\mathrm{cc}|$ represents the Euclidean distance between sample point x and the closest cluster center cc. The probability function of x_i selected as the cluster center is computed as follows:

$$P(x_i) = \frac{D(x_i)^2}{\sum_{x_j\in X} D(x_j)^2}, \tag{7}$$

where x_i is a point in the dataset X. The numerator in the formula is to increase the probability of selecting a point that is far away from the existing cluster centers as the next cluster center. This ensures that the new centers are well-spaced and helps to improve the clustering quality. Meanwhile, the denominator serves to normalize the probability so that the sum of the probability of all points being selected is equal to 1. By doing so, the algorithm can ensure that the probability of selecting any given point is proportional to its squared distance from the nearest cluster center that has already been selected.

In this work, we divided all clients into three clusters based on their contribution scores by setting k to 3 in k-means++. The entire process of the k-means++ algorithm is summarized in Algorithm 1. This division is not a one-time process. After each update of the contribution scores, we need to use the k-means++ algorithm to recluster. Therefore, to ensure the fairness of FL and make the clustering of the k-means++ algorithm reasonable, an optimal contribution scores updating strategy needs to be adopted. In the proposed scheme, we use DQN to dynamically update the contribution scores of each client, ensuring the fairness of FL.

4. DQN-Based Contribution Scores Update Strategy

In this article, we adopt contribution scores to evaluate and acknowledge the individual contributions of clients in the context of FL. However, devising a fair and effective scheme for computing these scores poses a significant challenge. It necessitates a delicate balance between appropriately rewarding each client's efforts and deterring free-riders seeking personal benefits. Assigning a client with a score higher than their actual contribution might foster discontent among other clients and exacerbate the free-rider problem. Conversely, if a client's contribution is inaccurately reflected in their scores, it may dampen their enthusiasm and diminish their engagement in the FL process. Hence, it is imperative to derive contribution scores in a judicious manner that ensures fairness. By doing so, we can establish an equitable FL system that motivates active participation and fosters collaborative progress.

Determining the optimal contribution scores presents a formidable challenge for the server in the FL. The dynamic and uncertain nature of client behavior within the FL environment introduces complexities that directly impact the assignment of contribution scores. The stochasticity of client behavior over time and the potential for clients to withdraw from the FL task due to various environmental factors further exacerbate the challenge. These factors may encompass unreliable network connections, device mobility, or device energy limitations. Therefore, to address this issue and achieve the optimal contribution scores selection, we propose a contribution scores update mechanism leveraging DQN in this work.

4.1. Reinforcement Learning for Contribution Scores Update.

In our proposed scheme, the server in FL acts as an agent for DQN to determine the optimal contribution scores update policy by interacting with the FL environment. The server can receive model parameters sent to it by each client. Based on these parameters, the server can estimate the quality of the model submitted by the client. Moreover, the server also knows the previous contribution score of each client and which cluster the client belongs to. Therefore, the server can estimate the current state based on the previous contribution scores and the quality of the received model. Based on the optimal policy, the server can choose the action, i.e., the contribution scores update the policy to maximize the reward. We define the components of our proposed model as follows:

(1) *Agent* represents the parameter server of the FL system.

(2) *Environment* represents the FL system that is divided into three clusters based on the contribution scores.

```
Input: contribution scores set Con, number of clusters $k$
Output: clustering results $C = \{C_1, C_2, \ldots, C_k\}$
(1)  Randomly select a sample $cc_1$ from Con as the first cluster center.
(2)  for $j = 2, 3, \ldots, k$ do
(3)      Calculate the shortest distance $d_i$ between each sample $x_i$ and the existing cluster centers $\mathrm{cc}_1, \mathrm{cc}_2, \ldots, \mathrm{cc}_{j-1}$.
(4)      Select the next cluster center $cc_j$ with probability $p_i = d_i^2 / \sum_{i=1}^{n} d_i^2$.
(5)  end for
(6)  Use the selected $k$ cluster centers as initial cluster centers, i.e., $\mathrm{CC} = \{\mathrm{cc}_1, \mathrm{cc}_2, \ldots, \mathrm{cc}_k\}$.
(7)  while The cluster centers no longer change do
(8)      For each sample $x_i \in$ Con, calculate its distance to each cluster center $\mathrm{dist}(x_i, \mathrm{cc}_j)$, where $\mathrm{cc}_j \in CC$.
(9)      Assign each sample $x_i$ to the cluster $C_j$ of nearest cluster center $\mathrm{cc}_j$.
(10)     for $j = 1, 2, \ldots, k$ do
(11)         Calculate the mean $\mathrm{mean}_j$ of all samples in cluster $C_j$.
(12)         update the cluster center $\mathrm{cc}_j = \mathrm{mean}_j$
(13)     end for
(14) end while
(15) Output clustering results $C = \{C_1, C_2, \ldots, C_k\}$
```

Algorithm 1: Clustering algorithm based on contribution.

(3) *State Space* s_t of the server is defined using contribution scores, model accuracy, and clustering results as follows:

$$s_t^k = \left[\mathrm{cs}^k, \mathrm{acc}^k, C\right]^T; \quad k \in 1, 2, \ldots, K, \tag{8}$$

where cs^k is the contribution score of the k-th client calculated by the server, and $cs^k \in [1, 10]$. acc^k is the accuracy of the model **submitted** by the k-th client at each global iteration t. C is the clustering result by using k-means++ based on the contribution scores. These state metrics provide the basis for determining the contribution scores update strategy in our scheme. Hence, the state space of the server can be defined as follows:

$$s_t = \left[s_t^1, \ldots, s_t^k, \ldots, s_t^K\right]. \tag{9}$$

(4) *Action Space* is denoted by a_t. At each global iteration t, the server selects an action a_t^k for client k. The contribution scores update policy is denoted by

$$a_t = \left[a_t^1, \ldots, a_t^k, \ldots, a_t^K\right], \tag{10}$$

where a^k is defined as a discrete variable taking values in the set $\{-1, 0, 1\}$. Specifically, it represents the actions of decreasing, maintaining, and increasing the contribution score cs^k for client k.

(5) *Reward* r_t acts as a reinforcement signal that indicates the effectiveness of its chosen actions, guiding the learning and optimization process. In order to measure the contributions of different parties in FL, we employ a method based on deletion diagnostics [37]. We assume a test dataset is available at the server to evaluate the learned model after global model aggregation. The deletion approach entails reaggregating the model each time a local model from a cluster is omitted and measuring the change in model accuracy. Specifically, when evaluating the impact of the k-th client in the cluster C_i in FL, the influence measure can be formulated as follows:

$$\mathrm{Influence}^{-k} = \mathrm{acc}_i - \mathrm{acc}_i^{-k}, \tag{11}$$

where acc_i is the global model accuracy of cluster C_i and acc_i^{-k} is the accuracy of the global model when the local model of the k-th client is omitted. Then, it is necessary to perform min-max normalization on $\mathrm{Influence}^{-k}$ within each cluster separately. Reward r_t^k of agent k at current iteration t is denoted by

$$r_t^k = \mathrm{Influence}^{-k} \times \frac{cs^k}{\mathrm{CS}}. \tag{12}$$

The reward r_t can be defined as follows:

$$r_t = \left[r_t^1, \ldots, r_t^k, \ldots, r_t^K\right]. \tag{13}$$

The server, i.e., the agent perceives a state s_t based on feedback from the FL system and selects an action a_t, represents contribution scores update policy. The action a_t is determined by the optimal contribution scores update policy π given the current state. The server executes the chosen action a_t and receives the reward r_t from the FL environment. The environment follows a Markov decision process (MDP) and transitions to a new state s_{t+1} following MDP. The objective of the agent is to maximize the expected long-term discounted rewards by determining the optimal policy. The optimal policy can be obtained using the Q-value, which can be updated by using the following expression:

$$\begin{aligned} Q_{t+1}(s_t, a_t) &= Q_t(s_t, a_t) \\ &\quad + \alpha\left[r_t + \gamma \times \max Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t)\right], \end{aligned} \tag{14}$$

where α is the learning rate, r_t denotes the immediate reward received by the agent for selecting action a_t in state s_t at time t, γ is the discount factor that determines the significance of future rewards, s_{t+1} represents the new state reached after executing action a_t, and $\max Q_t(s_{t+1}, a_{t+1})$ denotes the maximum Q-value among all possible actions a_{t+1} in state s_{t+1} at time $t + 1$. This Q-value update equation is known as the Q-learning update rule with temporal difference (TD) learning and is used to update the Q-value function following each action taken by the agent at each time step. The Q-learning algorithm with TD learning is a model-free reinforcement learning algorithm that employs this update rule to learn the optimal policy for the agent over time. The optimal policy can be expressed as follows:

$$\pi^* = \underset{a \in A}{\operatorname{argmax}} Q^*(s, a). \tag{15}$$

In the MDP, the effectiveness of the Q-learning algorithm relies on a thorough exploration of states and actions, facilitating the convergence of Q-values towards the optimal Q-value, denoted as Q^*. The optimal Q-value Q^* is then utilized to derive the optimal policy. However, in large-scale MDPs, the size of the Q-table becomes prohibitively large, making exploration challenging. Due to the enormous number of state-action combinations, fully populating and updating the Q-table requires a significant amount of storage space and computational resources. To avoid the enormous overhead of maintaining a large Q-table, alternative methods are needed to approximate the Q-values. One popular approach is to utilize function approximators, such as neural networks, to estimate the Q-values.

4.2. Deep Reinforcement Learning. In the DQN algorithm, the Q-value function is approximated using a deep neural network. This network takes the state as input and outputs a Q-value for each possible action. During the training process, instead of directly updating a Q-table, the weights of the network are updated to minimize the discrepancy between the predicted Q-values and the target Q-values. DQN leverages state-action pairs to maximize the cumulative discounted rewards. The neural network employs weights, denoted as θ, to establish relationships between inputs and outputs. The objective of DQN is to minimize the loss by finding the optimal weights along the gradients. Initially, the weight of the DQN coefficients is randomly initialized. Over time, the DQN iteratively updates its weights based on the discrepancy between the expected reward and the ground truth reward. At each iteration, the loss function is minimized using the following equation:

$$l(\theta) = E\left[\left(Q_{\text{target}} - Q(s_t, a_t; \theta)\right)^2\right], \tag{16}$$

where θ is the set of weights in the deep neural network, $Q(s_t, a_t; \theta)$ is the predicted value, and Q_{target} is the target values which can be computed as follows:

$$Q_{\text{target}} = r + \gamma \max_{a \in A} Q(s_{t+1}, a_{t+1}; \theta), \tag{17}$$

where r is the reward received for taking action a_t in state s_t and γ is the discount factor for future rewards.

The objective of the DQN algorithm is to determine the optimal set of weights denoted as θ. By utilizing the gradients of the loss function with respect to the weights, the DQN algorithm updates the weights iteratively in a manner that minimizes the discrepancy between the estimated Q-values and the true Q-values. This process involves adjusting the weights in the direction that reduces the overall loss, allowing the DQN model to better approximate the optimal Q-values for different state-action pairs. Through this iterative weight update process, the DQN algorithm gradually learns to estimate more accurate Q-values, enhancing its ability to make optimal decisions in complex environments such as FL.

In our DQN environment, the policy is randomly initialized and progressively refined through training. The DQN is deployed on the server, which interacts with the environment to gather information for decision-making. The agent selects actions based on the current state and the learned policy from the DQN. The exploration-exploitation trade-off is a critical aspect of the DQN algorithm. Exploration involves trying out new actions to gain a better understanding of the environment, while exploitation involves leveraging the acquired knowledge to make optimal decisions based on the current state. Striking the right balance between exploration and exploitation is crucial to achieving good performance. To address this challenge, a commonly used technique in DQN is a ϵ-greedy algorithm. This strategy entails selecting a random action with a probability of ϵ and choosing the action with the highest Q-value with a probability of $1 - \epsilon$. The value of ϵ typically decreases over time to shift the emphasis towards exploitation as the agent gains more knowledge about the environment. By carefully managing the exploration-exploitation trade-off through a ϵ-greedy algorithm, the DQN can effectively explore the environment while gradually focusing on exploiting the learned knowledge to converge towards an optimal policy.

The agent of DQN actively interacts with the environment by selecting actions and receiving corresponding rewards. These interaction data are recorded and stored in a memory buffer, creating a collection of past experiences. During the training process, the deep neural network is updated using mini-batches of experiences sampled from the memory buffer. The objective is to minimize the discrepancy between the predicted Q-values and the actual rewards obtained. Experience replay is a technique used in deep reinforcement learning [38]. By doing so, the agent can learn from a diverse set of experiences, preventing the learning process from being biased towards recent experiences. By leveraging experience replay, DQN leverages past experiences to improve the stability and efficiency of training. The agent can draw upon a diverse set of experiences and learn from a broader range of scenarios, resulting in enhanced performance and adaptability in complex environments such as FL. The whole procedure of contribution scores update with DQN is summarized in Algorithm 2.

```
     Input: current state $s_t = \{cs, acc, C\}$
     Output: contribution scores set Con
(1)  Initialize the global model parameters for each cluster, experience replay memory $D$
(2)  Initialize action-value function $Q$ with random weights $\theta$
(3)  Initialize target action-value function $\widehat{Q}$ with weights $\theta^- = \theta$
(4)  for $t = 1$ to $T$ do
(5)      With probability $\epsilon$ select random actions $a_t$
(6)      Otherwise select actions $a_t = \text{argmax}_a Q(s_t, a; \theta)$
(7)      Execute actions $a_t$, update contribution score $cs$ for each client.
(8)      Obtain clustering results $C$ by using Algorithm 1 based on contribution score $cs$
(9)      Run the FL algorithms independently on each cluster
(10)     Observe reward $r_t$ and next state $s_{t+1}$
(11)     Store transition $(s_t, a_t, r_t, s_{t+1})$ in $D$
(12)     Sample random mini-batch of transitions from $D$
(13)     Update weights $\theta$ by minimizing the loss
(14)     At every certain step, update the target network weights: $\theta^- = \theta$
(15) end for
```

Algorithm 2: DQN-based contribution scores update for FL.

5. Performance Evaluation

5.1. Simulation Settings. This section evaluates the performance of our proposed DQN-based contribution scores update and evaluation strategy through simulations. The simulations were performed on Nvidia GeForce GPUs version RTX 3060 running on Windows 11. The proposed model is developed using Python 3 and Pytorch. The experiment is conducted on the well-known MNIST dataset for handwritten digit recognition, which consists of 60,000 training examples and 10,000 test examples [39]. Each example is a 28×28 gray-level image, with digits located at the center of the image. This dataset has been extensively used in several FL evaluations.

We simulated the FL environment by iteratively training models on the MNIST dataset. For the local model architecture of the clients, we chose to employ a convolutional neural network (CNN). When training the local models, we set the learning rate to 0.01 and the batch size to 20. The training samples were distributed in a highly imbalanced manner, where the labels available to the clients were unevenly and randomly distributed. In this work, we considered a more realistic scenario where 2500 randomly selected examples were assigned to high-contributing clients, 600 examples were allocated to ordinary clients, and free-riders had access to only 150 examples of data.

The DQN agent consists of two hidden layers, with the number of units equal to the number of states and actions, respectively. The neural network utilizes the rectified linear unit (ReLU) as the activation function for all hidden layers, enhancing the fitting capability of this network. For learning the neural network parameters, we employ the Adam optimizer with a learning rate of 0.01. The discount factor γ is set to 0.99 to adjust the importance of future rewards in the current decision-making process. To break the temporal correlation between consecutive training samples, we use a replay memory of size 64. The simulation parameters of the DQN are summarized in Table 1.

Table 1: Simulation parameters for DQN-based contribution scores update for FL.

Parameter	Value
Replay memory size	10000
Batch size	64
Optimizer	Adam
Activation function	ReLu
Learning rate of DQN	0.01
Discount factor (γ)	0.99
Initial contribution score (cs)	5
Number of workers (K)	8 to 32
Ratio of high-contributing clients	0.25
Ratio of ordinary clients	0.5
Ratio of free-riders	0.25
Number of FL iterations	100

To visualize the fairness of FL from a holistic perspective, we quantify fairness by calculating the Pearson correlation coefficient between the contributions of clients (i.e., the amount of data contributed by each client) and their rewards (i.e., the final model accuracy achieved by each client). Specifically, we construct a coordinate system with the contributions of clients as the x-axis, and the corresponding model accuracies achieved by each client as the y-axis [24]. By computing the Pearson correlation coefficient, we can quantify the fairness of FL. The fairness metric ranges from −1 to 1, where higher values indicate better fairness. Conversely, negative coefficients indicate poorer fairness.

5.2. Performance Results. We conducted a series of experiments to evaluate the performance of our proposed method in a FL environment. The experiments involved different numbers of clients, namely 4, 8, 16, and 32, denoted as N4, N8, N16, and N32, respectively. Table 2 presents a comparison of fairness between the FedAvg algorithm and our proposed method, considering different numbers of clients

Table 2: Fairness of FedAvg and our method over MNIST dataset, with different client numbers (N-k).

Method	N4	N8	N16	N32
FedAvg	−0.23	−0.46	0.13	0.04
Our method	0.90	0.77	0.68	0.73

in the MNIST dataset. Fairness evaluation is conducted based on the Pearson correlation coefficients between client contributions and the corresponding model accuracy received by each client. Higher coefficients indicate a stronger positive correlation, indicating a higher level of fairness in the distribution of the model.

For the FedAvg algorithm, poor correlation coefficients are observed across all client numbers (N4, N8, N16, and N32), with some coefficients being negative, indicating a lack of fairness. Negative coefficients imply an inverse relationship between client contributions and model accuracy, where higher contributions do not necessarily result in higher accuracy. This signifies an unfair distribution of the model, with clients contributing more data and receiving lower model accuracy.

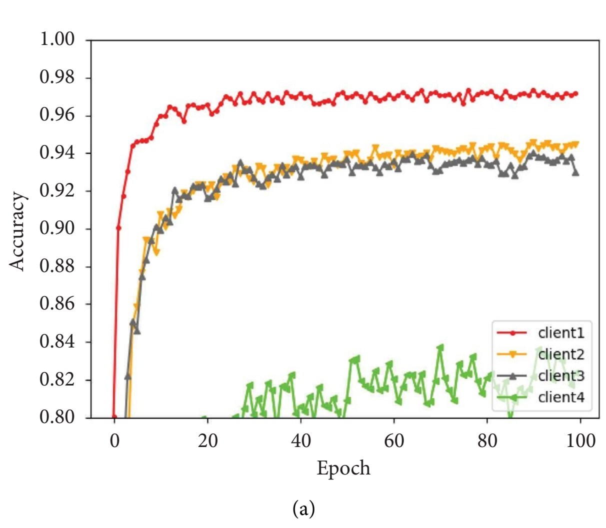
In contrast, our proposed method demonstrates significant improvements in fairness. Positive correlation coefficients are obtained, all above 0.5, indicating a positive relationship between client contributions and model accuracy. Higher data contributions are positively correlated with higher accuracy, suggesting a fair distribution of the model among clients. These positive coefficients reflect the effectiveness of our method in addressing fairness concerns in federated learning.

The findings emphasize the benefits of incorporating the DQN-based contribution scores update method into the federated learning framework. By dynamically updating the contribution scores of clients based on DQN, our method achieves a more equitable distribution of the model, ensuring that clients contributing more data are rewarded with higher accuracy models. The positive correlation coefficients affirm the improved fairness achieved by our proposed method, validating its potential for ensuring a more equitable distribution of the model among participating clients. This improvement in fairness is particularly noteworthy when compared to the FedAvg algorithm, which lacks mechanisms for addressing fairness concerns.

In Figures 2–5, we compare the performance of our proposed method with several baseline methods, including FedAvg, centralized framework, and standalone framework. Our proposed method consists of three global models represented by curves C1, C2, and C3 in the figures. These curves correspond to the model accuracy that clients with different contribution scores will obtain. FedAvg is a common federated learning algorithm. The centralized framework represents the centralized approach, assuming that the server can access user data and centralize all client data for training. Centralized training infringes upon client privacy. The standalone framework assumes that clients do not collaborate with each other and train their models independently using their own data. This method maximizes client privacy but may lead to suboptimal results.

In Figure 2, we simulate the case with four participating clients. Figure 2(a) shows the varying model performance among different clients due to the differences in their respective sample data. The high-contributing clients quickly improve their accuracy, converging to approximately 0.97. The ordinary clients achieve slightly lower accuracy, converging around 0.93. On the other hand, the free-riders achieve the poorest model performance, struggling to surpass an accuracy of 0.9. To provide a comprehensive comparison, we calculate the average accuracy of each client in the standalone framework and include it as the standalone curve in Figure 2(b). Figure 2(b) illustrates the accuracy variations of different methods as the models iterate over time. With each iteration, the performance of these models improves and eventually converges. Among them, the centralized framework exhibits the fastest convergence rate and achieves the highest model performance. However, in the initial stages, the performance of the FedAvg model is inferior to the models corresponding to cluster C1, due to the influence of free-riders. From the graph, it can be seen that our proposed method effectively enhances fairness in FL. The model performance corresponding to cluster C1 outperforms the model performance of C2, and the model performance of C2 is superior to C3. This correlation between the model performance and the contribution scores of clients highlights the impact of our approach. Furthermore, the model performance of cluster C3, composed of clients with low contribution scores, is even inferior to the average performance of the standalone framework.

To further validate the effectiveness of our proposed method, we expand the number of participating clients and conduct multiple experiments. Figures 3–5 display the accuracy curves of different methods when the number of clients is 8, 16, and 32, respectively. In Figures 3(a), 4(a), and 5(a), it is evident that the accuracy of models, trained by clients with varying sample quantities, can be distinctly categorized into three groups, effectively simulating the differences between clients with different contributions. Even clients with the same sample quantity exhibit slight variations in accuracy, as each sample of clients is randomly selected from the MNIST dataset. In the initial stages of model iteration, certain clients do not receive matching contribution scores since all contribution scores of clients are initialized as 5. As shown in Figure 3(b), the accuracy curve corresponding to cluster C3 rises rapidly at the beginning. However, as the model iterates, each client receives a reasonable contribution score and is assigned to the corresponding cluster. Clients who make significant contributions in cluster C3 receive higher contribution scores, leading them to leave C3 and join the cluster where they belong. This gradually restores the accuracy curve of cluster C3 to its appropriate level. As the number of participating clients increases, the training dataset also expands, resulting in improved model accuracy for different methods. When



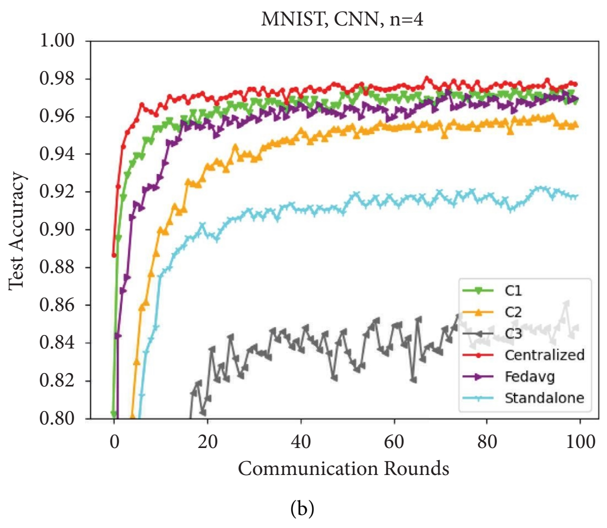
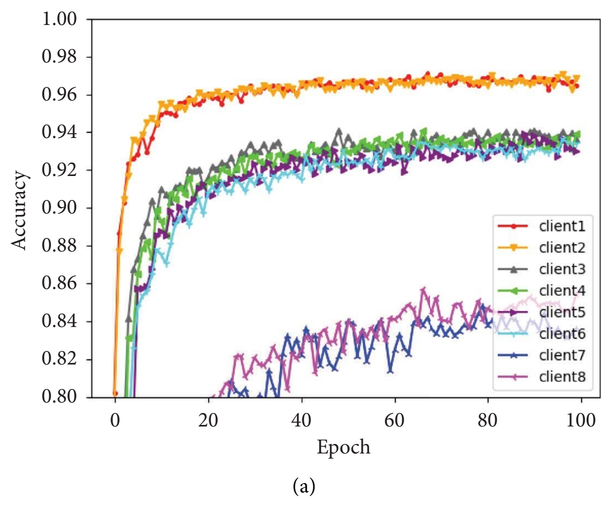


FIGURE 2: Performance comparison with 4 participating clients. (a) Comparison between clients. (b) Comparison of different methods.



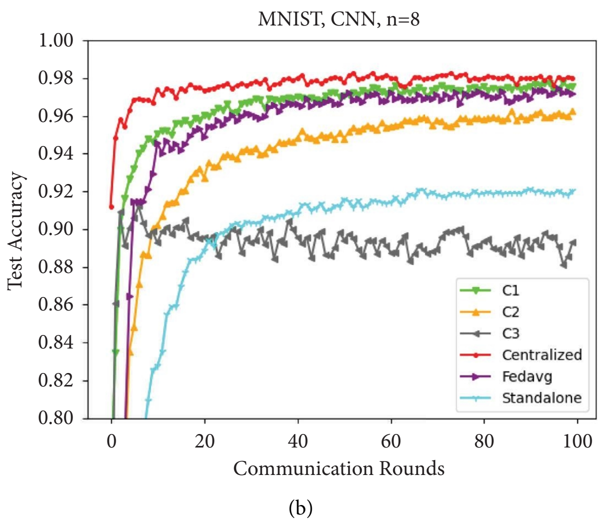
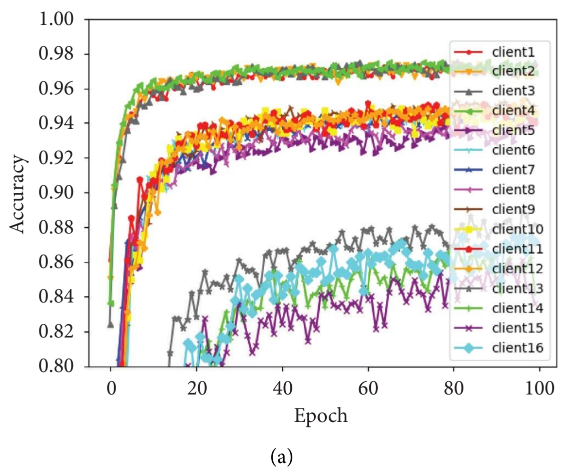


FIGURE 3: Performance comparison with 8 participating clients. (a) Comparison between clients. (b) Comparison of different methods.



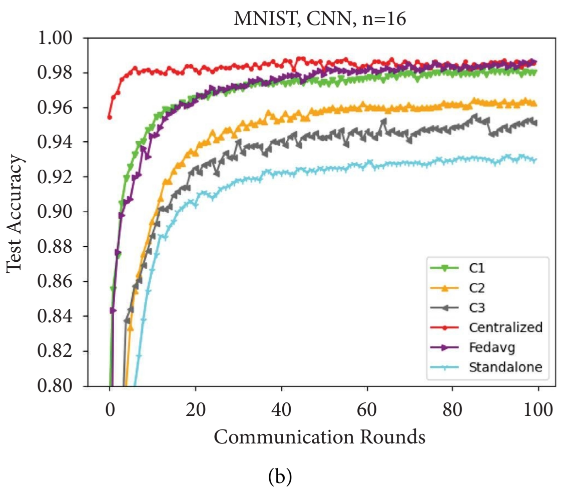


FIGURE 4: Performance comparison with 16 participating clients. (a) Comparison between clients. (b) Comparison of different methods.

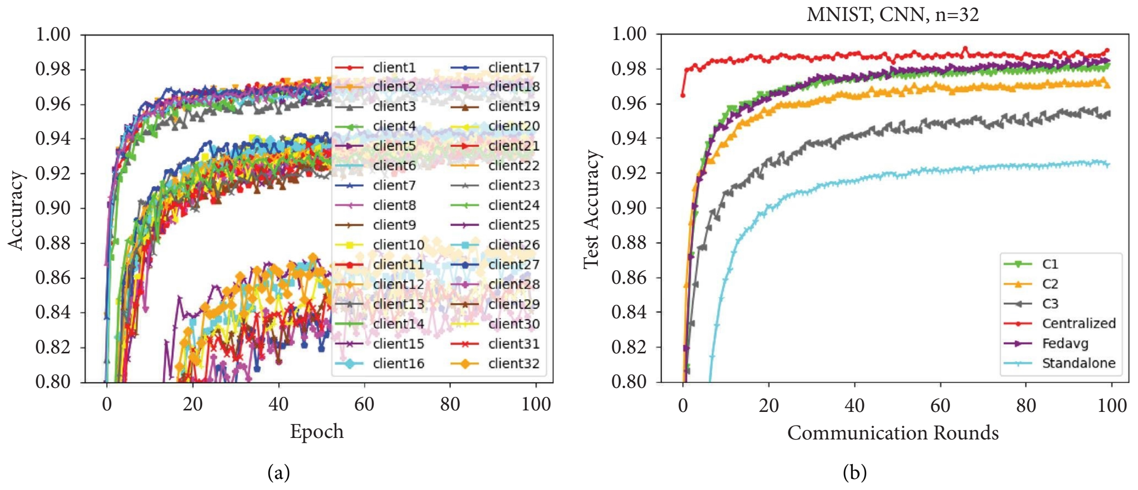


Figure 5: Performance comparison with 32 participating clients. (a) Comparison between clients. (b) Comparison of different methods.

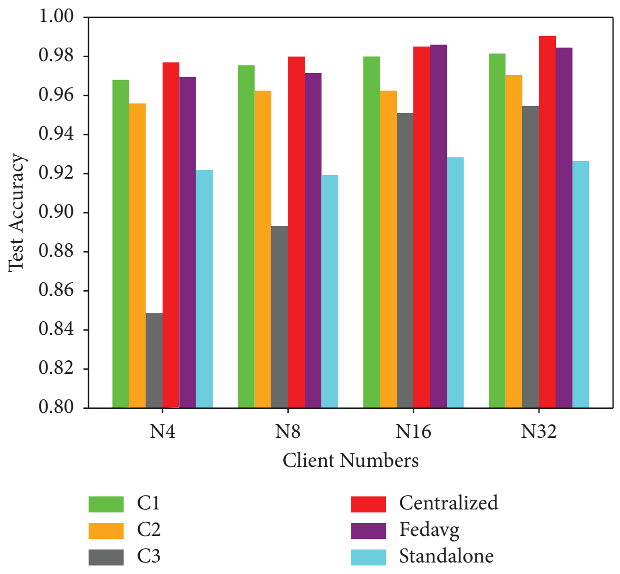


Figure 6: Final model accuracy across varying client numbers.

the number of clients is 4 and 8, the model performance of cluster C3 is worse than the average of the standalone framework. However, in Figures 4(b) and 5(b), the model performance of cluster C3 surpasses the average of the standalone framework. Benefiting from the increased training samples, the centralized curve in Figure 5(b) exhibits the most significant improvement, achieving an accuracy of above 0.96 in the early stages of training and reaching 0.99 after 100 rounds. Since the standalone curve in Figure (a) represents the average of the standalone framework, it is minimally affected by changes in the number of clients, remaining around 0.92.

To ensure fairness in FL, we aim to provide greater benefits to clients who contribute more, i.e., better models. For visual comparison, we present bar graphs in Figure 6, depicting the model accuracies obtained by clients under different methods and client quantities. It is evident that the centralized framework achieves the highest model performance. However, it requires the server to have access to the privacy data of all clients, which is challenging to realize in practice. On the other hand, FedAvg considers data privacy and achieves slightly lower model performance compared to the centralized framework. Nevertheless, it does not consider fairness. In FedAvg, regardless of the contributions, clients can only obtain the same model, which discourages active participation from high-contributing clients. To address this, we utilize the k-means++ algorithm to cluster different clients based on their contribution scores, resulting in three clusters: C1, C2, and C3. Figure 6 demonstrates a clear hierarchy, where clusters composed of clients with higher contribution scores exhibit superior model performance compared to clusters with lower contribution scores. This effectively ensures fairness in FL and motivates clients to contribute more data to the FL process.

6. Conclusion

This study presents the issue of fairness in FL and proposes a method of differentiated global models to enhance fairness. Specifically, we use DQN to dynamically adjust the contribution scores and group clients based on them, enabling each client to obtain global models with diverse performance. Experimental results demonstrate that our method significantly improves the fairness of FedAvg. Our method maintains fairness above 0.5, which FedAvg suffers from low or negative fairness.

For future work, we plan to consider more metrics to evaluate the contributions of clients, such as computational power and data quality. We also aim to apply our method in real-world scenarios to test its practical effectiveness and impact. By doing so, we hope to further advance FL and promote fair and equitable collaboration among clients in various domains.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This work was supported by the National Nature Science Foundation of China (No. 61962022 and No. 62062034).

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, PMLR, London, UK, 2017.

[2] W. Liang, Y. Yang, C. Yang et al., "Pdpchain: a consortium blockchain-based privacy protection scheme for personal data," *IEEE Transactions on Reliability*, vol. 72, no. 2, pp. 586–598, 2023.

[3] W. Du, D. Xu, X. Wu, and H. Tong, "Fairness-aware agnostic federated learning," in *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, SIAM, Thailand, Asia, 2021.

[4] X. Tu, K. Zhu, N. C. Luong, D. Niyato, Y. Zhang, and J. Li, "Incentive mechanisms for federated learning: from economic and game theoretic perspective," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 3, pp. 1566–1593, 2022.

[5] S. K. Lo, Y. Liu, Q. Lu et al., "Toward trustworthy ai: blockchain-based architecture design for accountability and fairness of federated learning systems," *IEEE Internet of Things Journal*, vol. 10, no. 4, pp. 3276–3284, 2023.

[6] W. Huang, T. Li, D. Wang, S. Du, J. Zhang, and T. Huang, "Fairness and accuracy in horizontal federated learning," *Information Sciences*, vol. 589, pp. 170–185, 2022.

[7] H. Lv, Z. Zheng, T. Luo et al., "Data-free evaluation of user contributions in federated learning," in *Proceedings of the 19th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, Turin, Italy, June 2021.

[8] H. Yu, Z. Liu, Y. Liu et al., "A fairness-aware incentive scheme for federated learning," in *Proceedings of the AAAI/ACM Conference on AI*, pp. 393–399, New York, NY, USA, August 2020.

[9] T. H. Thi Le, N. H. Tran, Y. K. Tun et al., "An incentive mechanism for federated learning in wireless cellular networks: an auction approach," *IEEE Transactions on Wireless Communications*, vol. 20, no. 8, pp. 4874–4887, 2021.

[10] S. Zhang, B. Hu, W. Liang, K.-C. Li, and B. B. Gupta, "A caching-based dual k-anonymous location privacy-preserving scheme for edge computing," *IEEE Internet of Things Journal*, vol. 10, no. 11, pp. 9768–9781, 2023.

[11] Y. Deng, F. Lyu, J. Ren et al., "Improving federated learning with quality-aware user incentive and auto-weighted model aggregation," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 4515–4529, 2022.

[12] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT press, Cambridge, MA, USA, 2018.

[13] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[14] J. Liu, H. Xu, L. Wang et al., "Adaptive asynchronous federated learning in resource-constrained edge computing," *IEEE Transactions on Mobile Computing*, vol. 22, no. 2, pp. 674–690, 2023.

[15] P. Zhang, C. Wang, C. Jiang, and Z. Han, "Deep reinforcement learning assisted federated learning algorithm for data management of iiot," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 12, pp. 8475–8484, 2021.

[16] Y. Jiao, P. Wang, D. Niyato, B. Lin, and D. I. Kim, "Toward an automated auction framework for wireless federated learning services market," *IEEE Transactions on Mobile Computing*, vol. 20, no. 10, pp. 3034–3048, 2021.

[17] Y. Zhao, J. Zhao, L. Jiang et al., "Privacy-preserving blockchain-based federated learning for iot devices," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1817–1829, 2021.

[18] J. Kang, Z. Xiong, D. Niyato, Y. Zou, Y. Zhang, and M. Guizani, "Reliable federated learning for mobile networks," *IEEE Wireless Communications*, vol. 27, no. 2, pp. 72–80, 2020.

[19] J. Zhang, Y. Wu, and R. Pan, "Incentive mechanism for horizontal federated learning based on reputation and reverse auction," *Proceedings of the Web Conference*, vol. 2021, pp. 947–956, 2021.

[20] S. Ma, Y. Cao, and L. Xiong, "Transparent contribution evaluation for secure federated learning on blockchain," in *Proceedings of the IEEE 37th International Conference on Data Engineering Workshops (ICDEW)*, Chania, Greece, April 2021.

[21] Z. Liu, Y. Chen, H. Yu, Y. Liu, and L. Cui, "Gtg-shapley: efficient and accurate participant contribution evaluation in federated learning," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 13, no. 4, pp. 1–21, 2022.

[22] T. Nishio, R. Shinkuma, and N. B. Mandayam, "Estimation of Individual Device Contributions for Incentivizing Federated Learning," *IEEE Globecom Workshops (GC Wkshps)*, vol. 12, 2020.

[23] Y. Deng, F. Lyu, J. Ren et al., "Fair: quality-aware federated learning with precise user incentive and model aggregation," in *Proceedings of the IEEE INFOCOM 2021- IEEE Conference on Computer Communications*, pp. 1–10, Vancouver, Canada, May 2021.

[24] 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.

[25] R. Zeng, S. Zhang, J. Wang, and X. Chu, "Fmore: an incentive scheme of multi-dimensional auction for federated learning in mec," in *Proceedings of the IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, Singapore, November 2020.

[26] Z. Shi, L. Zhang, Z. Yao et al., "Fedfaim: a model performance-based fair incentive mechanism for federated learning," *IEEE Transactions on Big Data*, vol. 14, 2022.

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

[28] Z. Fan, H. Fang, Z. Zhou et al., "Improving fairness for data valuation in horizontal federated learning," in *Proceedings of the IEEE 38th International Conference on Data Engineering (ICDE)*, Kuala Lumpur, Malaysia, June 2022.

[29] N. M. Al-Maslamani, B. S. Ciftler, M. Abdallah, and M. M. Mahmoud, "Toward secure federated learning for iot using drl-enabled reputation mechanism," *IEEE Internet of Things Journal*, vol. 9, no. 21, pp. 21971–21983, 2022.

[30] Y. Zhan and J. Zhang, "An incentive mechanism design for efficient edge learning by deep reinforcement learning approach," in *Proceedings of the IEEE INFOCOM 2020- IEEE Conference on Computer Communications*, pp. 2489–2498, Toronto, Canada, July 2020.

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

[32] L. Yu, R. Albelaihi, X. Sun, N. Ansari, and M. Devetsikiotis, "Jointly optimizing client selection and resource management in wireless federated learning for internet of things," *IEEE Internet of Things Journal*, vol. 9, no. 6, pp. 4385–4395, 2022.

[33] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[34] Y. Fraboni, R. Vidal, and M. Lorenzi, "Free-rider attacks on model aggregation in federated learning," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 1846–1854, PMLR, Valencia, Spain, June 2021.

[35] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu, "BatchCrypt: efficient homomorphic encryption for Cross-Silo federated learning," in *Proceedings of the USENIX Annual Technical Conference (USENIX ATC 20)*, Berkeley, CA, USA, July 2020.

[36] D. Arthur and S. Vassilvitskii, "K-means++ the advantages of careful seeding," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1027–1035, 2007.

[37] G. Wang, C. X. Dang, and Z. Zhou, "Measure contribution of participants in federated learning," in *Proceedings of the IEEE International Conference on Big Data (Big Data)*, Los Angeles, CA, USA, December 2019.

[38] W. Fedus, P. Ramachandran, R. Agarwal et al., "Revisiting fundamentals of experience replay," in *Proceedings of the International Conference on Machine Learning*, PMLR, Valencia, Spain, June 2020.

[39] Y. LeCun, L. D. Jackel, L. Bottou et al., "Learning algorithms for classification: a comparison on handwritten digit recognition," *Neural Networks: The Statistical Mechanics Perspective*, vol. 261, no. 276, p. 2, 1995.