M2F: Multi-centered Fairness-Aware Federated Learning Framework

Jing Deng, Handi Chen, Yunhin Chan, and Edith Ngai(✉)

The University of Hong Kong, Hong Kong SAR, China
{gracedeng,hdchen}@connect.hku.hk, {yhchan,chngai}@eee.hku.hk

Abstract. Federated learning (FL) is a promising technique to train machine learning models across distributed and privacy-conscious devices. The clients co-train a global model, while their contributions to global model training differ due to inherent heterogeneity in data and capabilities. This induces inequitable incentives for their contributions. Existing incentive mechanisms relying on a single model for incentive allocation often underestimate client contributions when there are significant data discrepancies among them. Therefore, this paper proposes a Multi-centered Fairness-aware FL framework (M2F). It implements a clustering method based on model similarity to construct personalized contribution evaluation adaptively. We also design a multi-dimensional metric to evaluate client quality by considering participation rate, computation ability, and training dataset size. In this design, clients receive a customized variant of the aggregated gradient as an incentive at the end of each training iteration. Experimental results validate that the M2F framework can accurately differentiate clients with heterogeneous datasets and diverse quality by increasing the convergence speed and accuracy gaps among them, hence promoting fairness.

Keywords: Federated learning · fairness · incentive mechanism

1 Introduction

Clients participating in standard federated learning receive the same global model in each iteration [2–4,8,22]. However, there exist free-riders [7], who are inclined to depend on information shared by others, rather than offering valuable local model information as a strategy to minimize personal training expenses. In this case, rewarding all clients with the same global model is unfair to active clients (i.e., the clients contribute more but receive the same incentives as the free-riders) in FL, who may be reluctant to contribute to the training process. Thus, clients' incentive fairness in FL has increasingly become a focus for researchers.

To achieve collaborative fairness, existing works mainly aim at a well-performed global model [10,12,15,18] by selecting clients based on the evaluation result to mitigate the influence of clients with less training data. However, clients'

V. C. M. Leung et al. (Eds.): Qshine 2023, LNICST 574, pp. 95–108, 2024.
https://doi.org/10.1007/978-3-031-65123-6_7

quality evaluated by a single global model in FL training is unfair for clients with large-discrepancy data distribution. Rational clients are self-interested and only concerned about the performance of their local models. Therefore, a personalized fairness-aware framework to adapt clients to different objective models is required in the heterogeneous FL training mentioned above.

To solve the above problem, the challenges to be tackled can be listed as follows:

- How to evaluate the model contribution of self-interested clients in heterogeneous FL training? Clients' local models are usually referenced against the global model. In many cases [5,11,14,15,21], distances between the global model and clients' local model are always taken as evaluation criteria to reflect clients' model contribution. However, client contributions can be underestimated in this way, especially when there are few intersections between their label space. In other words, quality measurement relying on a single global model is deceptive. Therefore, an adaptive reference frame for contribution evaluation is essential, especially among clients with diverse data distributions.
- How to evaluate the client quality in a comprehensive way? In recent years, most existing works only consider clients' model similarity to the global model and local training data size. However, client quality is also affected by other factors, such as participation rate and computation ability. A multi-dimension matrix is needed to obtain a comprehensive description of client quality.
- How to design a feedback mechanism to incentive clients according to diverse evaluated contributions? For clients with high contributions, their local model performance will be affected by low-contribution clients in a negative way in the long-term perspective. Therefore, it is necessary to distinguish clients by allocating rewards proportional to their contribution.

In this paper, we adapted hierarchical clustering in a federated learning framework to separate clients of various data distributions into different evaluation reference frames based on their relative contribution scores toward each other. Clients in the same cluster conduct gradient-weighted aggregation. Then a multi-dimensional matrix composed of computation ability, dataset size, and participation rate is applied to describe the quality score of clients in the same cluster. After quality evaluation, a portion of aggregated gradients will be allocated to a client according to its quality score. The contributions of our work can be summarized as follows:

- We propose a novel multi-centered federated learning framework to achieve personalized fairness in heterogeneous FL training by clustering clients according to their model similarity.
- We design a multi-dimension metric based on client ability, including participation rate, computation ability, and dataset size, to evaluate clients' quality comprehensively.
- We propose an incentive mechanism to leverage partial model information as rewards to give feedback to clients according to their quality evaluation.

– We demonstrate the effectiveness and fairness of the proposed M2F framework through comparisons to baseline and among clients in terms of data and quality heterogeneity.

2 Related Work

In recent years, researchers have proposed various incentive mechanisms to provide proper feedback [6,11,12,15,18,21] to participants in FL and promote fair collaboration [1,5,10,13,14,17,19,20]. These mechanisms aim to encourage active device participation, prevent free-riding behavior, and mitigate the impact of quality disparities among participants.

To ensure incentive fairness in FL, it is essential to prioritize fairness in evaluating client contributions. One common approach for tracking client contributions throughout the training process is by maintaining a reputation score list on the server. In each iteration, the reputation list will be updated by clients' contributions in the current iteration. Lyu *et al.* [6] utilize validation accuracy achieved by each client on a global model validation dataset to represent client contribution. However, a balanced and fair validation dataset in heterogeneous FL settings is probably unavailable, which may lead to bias in contribution evaluation. In [15], client reputation scores are updated by the cosine similarity among clients' model updates and global model updates. When the global model is trained collaboratively among clients with high-discrepancy datasets, the dissimilarity between the global and local models may not necessarily represent a low contribution level. Contribution from clients with complementary data may be underestimated or even neglected. Zhang *et al.* [20] calculate reputation through clients' local model updates, requiring a huge storage overhead to store the global and clients' local models in all iterations.

In addition to reputation, Sharply Value (SV) is also widely used in describing client contribution [1,5,11,13,14], which considers all possible permutations of the order in which users participate the FL training and calculates the average marginal contribution of each player across all such permutations. Song *et al.* [11] use gradient updates collected from clients to reconstruct the aggregated model in the server, which avoids retraining among clients in different subsets. Inspired by [1], Wang *et al.* [13] proposed two approaches to approximate SV effectively, i.e., permutation sampling-based approximation and group testing-based approximation. To further reduce the computation overhead, Liu *et al.* [5] proposed the Guided Truncation Gradient Shapley (GTGShapley) approach, which eliminates sub-model evaluation of the entire set in between-round truncation and a subset of permutations in within-round truncation when the remaining marginal gain is small. However, none of these methods is applicable when clients' model information is regarded as a valuable reward to others, since aggregation among clients in the calculation of SV has already leaked feedback to clients before the incentive scheme truly started.

Incentive mechanisms for achieving fairness can be classified into monetary and non-monetary incentives. Monetary incentive mechanisms, such as [17,18,

21], reward clients with bonuses, while non-monetary incentive mechanisms, such as [6,10,15,19] use a variation of the global model or a partially aggregated model as feedback to clients. Clients are typically drawn to monetary rewards, except when the value of the global model is incalculable or when rewarding clients with the future revenue generated by the global model is unfeasible.

In this paper, we design a framework to create an adaptive contribution evaluation standard for each client with diverse data distribution and incentive them with a customized variant of the aggregated gradient to achieve personalized fairness in FL.

3 Preliminaries and Framework

3.1 Federated Learning Problem Statement and Notations

In M2F framework, we consider a set of clients, denoted by $N := 1, \ldots, n$, along with their corresponding local datasets $D_1, \ldots, D_n$. The main notations mentioned in this paper are listed in Table 1. In each iteration, only a subset of these clients, denoted by $P := i, \ldots, j$, participate in the federated learning process, noted by $P := i, ..., j$. Client i $(i \in P)$ conducts local training on local datasets D_i for e_i epochs, modifying the model it has been allocated, which was broadcast by the server. Following this, the client sends the updated gradient g_i^t back to the server. The loss function of client i and the updated gradient after e epochs of local training can be denoted as follows:

$$F_i = \sum_{(x_j, y_j) \in D_i} f(w_i^t, x_j, y_j), \tag{1}$$

$$g_i^t = \Delta_{w_i^t} F_i. \tag{2}$$

With the collected gradients $g_{i\,i \in P}^t$, the server conducts Agglomerative Clustering [9] to group clients into C clusters. This clustering algorithm merges two clients with the minimum cosine distance at a time. The objective function of Agglomerative Clustering can be expressed as follows:

$$min \sum_{\forall i,j \in k, \forall k \in C} dist(i,j), \tag{3}$$

where $dist(\cdot)$ measures the distance between the two clusters being merged. Clients in the same cluster are aggregated with a weight determined by their quality score, q_i^t. For each cluster $k \in C$, the aggregated gradient can be represented as follows:

$$\hat{g}_k^t = \frac{q_i^t}{\sum\limits_{\forall i, c_i^t = k} q_i^t} g_i^t. \tag{4}$$

After multi-centered gradient aggregation, the server allocates partial aggregated gradient parameters to client i in proportion to its quality score. The allocated gradient can be denoted as follows:

$$\hat{g}_i^{t-} = sort(\hat{g}_{c_i}^t)_{\{:num_i^t\}}, \tag{5}$$

Table 1. Description of notations.

Notation	Description
N	The set of clients
P^t	The set of clients participating in iteration t
D_i	The local dataset of client i
e_i	The number of local training epochs of client i
C^t	The number of clusters in iteration t
c_i^t	The index of cluster client i belonging to in iteration t
g_i^t	The gradient of client i in iteration t
$\hat{g}_k^t$	The aggregated gradient of cluster $k \in C^t$ in iteration t
w_i^t	The weight of client i in iteration t
α_i^t	The aggregation weight of client i in iteration t
$r_{i,j}^t$	The relative contribution score of client i to client j in iteration t
ξ_i^t	The computation ability of client i in iteration t
τ_i^t	The local training time of client i in iteration t
ϕ_i^t	The participation rate of client i in iteration t
δ_i^t	In iteration t, the ratio of client i's training data volume to the total training data volume of all clients in the same group
q_i^t	The quality score of client i in iteration t
num_i^t	The number of gradients client i can download in iteration t
β^t	The clustering threshold of iteration t

$$num_i^t = \frac{q_i^t}{\sum\limits_{\forall i, c_i^t = k} q_i^t} |\hat{g}_{c_i}^t|. \tag{6}$$

Herein, $sort(\cdot)$ is a function that sorts parameters in descending order. c_i denotes the index of the cluster that client i belongs to, and num_i^t denotes the number of parameters that client i is allocated at the end of iteration t.

With the gradients allocated by the server, the process of local parameter updates can be represented as follows:

$$w_i^{t+1} \leftarrow w_i^t - \eta \cdot \hat{g}_i^{t-}. \tag{7}$$

3.2 Framework Overview

As depicted in Fig. 1, M2F framework primarily consists of two entities: the clients and the server. The clients incorporate only a Local Training module. On the server's side, there are four modules: Hierarchical Clustering, Quality Score Calculation, Multi-centered Gradient Aggregation, and Aggregated Gradient Allocation. The workflow of the framework proceeds as follows:

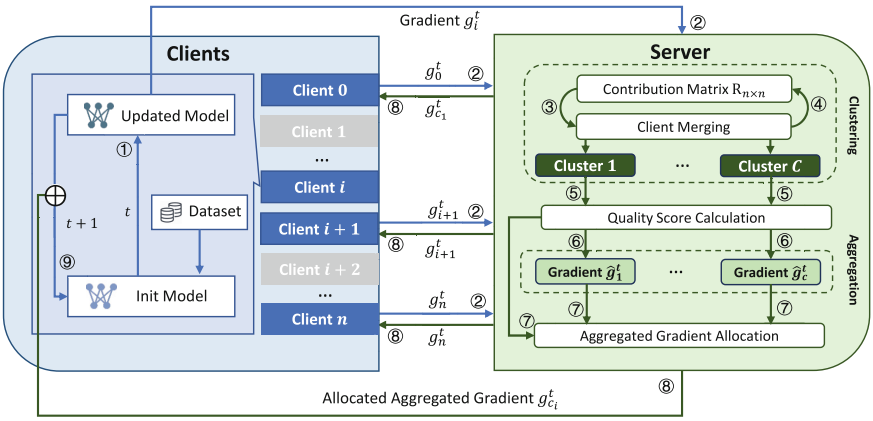

Fig. 1. The workflow of M2F framework

- **Step** ①: During iteration t, a subset of clients, denoted as P^t, participates in M2F framework. For each client $i \in P^t$, the local initial model is updated by conducting model training for e_i epochs on the local dataset D_i, yielding the local gradient g_i^t.
- **Step** ②: Client i sends the gradient g_i^t to the server.
- **Step** ③: The server maintains an $n * n$ distance matrix recording the pairwise cosine distance among all clients. Upon receiving the gradients from all clients in P^t, the server updates the i-th row for each $i \in P^t$. Initially, each client forms a cluster. Subsequently, the two clusters with the smallest cosine distance are merged into a new cluster.
- **Step** ④: The distance matrix is updated according to the result of Step 3. Steps 3 and 4 are then repeated until a stop condition is met. By the end of the Hierarchical Clustering module, clients are divided into C^t clusters.
- **Step** ⑤: The server conducts an intra-cluster quality score calculation. A client's quality score is determined by three factors: computational ability, data size, and participation rate.
- **Step** ⑥: For each cluster $k \in P^t$, the gradients from the clients are weighted by their quality scores and aggregated to produce $\hat{g}_k^t$.
- **Step** ⑦: The Aggregated Gradient Allocation module, taking $\hat{g}_k^t$ and clients' intra-cluster quality scores as input, rewards client i with a num_i number of gradient parameters.
- **Step** ⑧: The server sends the allocated gradient $\hat{g}_i^{t-}$ back to client i.
- **Step** ⑨: Upon receiving the allocated gradient $\hat{g}_i^{t-}$ from the server, client i updates its local model. The updated local model then serves as the initial model in iteration $t+1$.

We elaborate on the methodology in the framework in the following section.

4 Methodology

In this section, we further depict the methodology details of client side and server side respectively in Algorithm 1 and 2, and explain all the equations used in the algorithms.

4.1 Hierarchical Clustering in Multi-model Aggregation

In the proposed framework, clients are rewarded with a variant of the aggregated model from the cluster to which they belong. Hence, before information sharing between two clients, an evaluation to simulate the relative contribution between these clients is necessary. To implement the above-mentioned bottom-up hierarchical clustering, we employ the Agglomerative Clustering algorithm in the Hierarchical Clustering module. This algorithm maintains a distance matrix among the nodes to be clustered and iteratively merges the two nodes with the smallest distance until a termination condition is met. Importantly, this algorithm does not require assumptions about the number of clusters. Instead, we adjust the distance threshold among clusters to determine the final number of clusters.

Cosine distance between client gradients is leveraged to measure the relative contribution between two clients. The premise is when the angle between the update directions of the local models of client i and client j is small, the aggregation of the two clients' models tends to benefit their local models. In M2F framework, we use $R_{n\times n}$, a cosine distance matrix, to describe the dissimilarity among clients. This is the complement of the cosine similarity matrix. The i-th row and j-th column of matrix R can be represented as follows:

$$R_{i,j} = 1 - \frac{g_i \cdot g_j}{||g_i|| \cdot ||g_j||}, \tag{8}$$

where $||.||$ represents the Euclidean norm. A small $R_{i,j}$ represents a high similarity between the gradients from clients i and j.

4.2 Multi-dimensional Incentive Scheme

Different from frameworks that only consider model similarity [5,7,13,20], dataset size [16] or both [21], our method clusters clients based on their model similarity and then incorporates the following three dimensions for reward allocation at the end of each iteration, i.e., participation rate, computation ability, and dataset size.

– **Participation rate ϕ:** Client i's participation rate at iteration t is denoted as:

$$\phi_i^t = num(parti)/t, \tag{9}$$

representing the level of engagement and involvement of client i across the past t iterations.

Algorithm 1: M2F Framework: *ClientSide*

Input: local dataset D_i, the number of training epoch e_i, the obtained aggregated gradient $\hat{g}_i^{t-}$
Output: model gradient g_i^{t+1}

Initialize local model with initial weight downloaded from server;
Download allocated gradient $\hat{g}_i^{t-}$ from server;
Validate local model before and after updated by $\hat{g}_i^{t-}$ to obtain accuracy acc_i^{t-} and acc_i^{t+}, respectively;
if $acc_i^{t+} > acc_i^{t-}$ ***or*** $\frac{acc_i^{t-} - acc_i^{t+}}{acc_i^{t-}} < \sigma$ **then**
    Update local model with allocated gradient $\hat{g}_i^{t-}$;
end
foreach *epoch* $e \in e_i$ **do**
    Train local model on D_i and update local model based on loss function Eq. (1);
    Calculate gradient g_i^{t+1} with Eq. (2);
end
return updated gradient g_i^{t+1}.

– **Computation ability ξ:** We use local training time to gauge clients' hardware capacity for model training, calculated as follows:

$$\xi_i^t = \frac{1}{\tau_i^t / max_{j \in c_i}(\tau_j^t)}. \tag{10}$$

– **Dataset size ratio δ:** The size of clients' local data reflects their ability in data collection. We assume all clients use their entire local dataset for training and are willing to share all the gradients trained on this dataset. The dataset size ratio for client i at iteration t is calculated as follows:

$$\delta_i^t = \frac{|D_i|}{\sum\limits_{\forall j \in c_i} |D_j|}. \tag{11}$$

Then we can represent the quality score q_i^t for each client as:

$$q_i^t = \phi_i^t \cdot \xi_i^t \cdot \delta_i^t. \tag{12}$$

With the quality score in hand, the number of aggregated gradients a client can download and the gradients it receives at the end of the current iteration are represented by Eq. 6 and Eq. 5, respectively.

After receiving the rewarded gradients, clients with low-quality scores might face penalties, potentially experiencing a reduction in model performance. To prevent overly harsh penalties from the server, clients will validate the rewarded gradients and accept them if the model accuracy improves or if the accuracy reduction rate falls within a tolerable threshold (σ). By setting different values for σ, we can control the severity of the penalty imposed on a client in each iteration.

5 Experimental Evaluation

This section presents an experimental setup and result analysis on both the M2F and FedAvg frameworks under heterogeneous data distributions.

5.1 Experimental Setup

Dataset and Non-iid Data Distributions. We utilize the MNIST dataset for implementing handwritten digit classification tasks, consisting of 60,000 training images and 10,000 testing images. To simulate Non-iid data distributions among clients, we partition the label space of MNIST into 2 groups: $\{1, 2, 3, 4, 5\}$ and $\{0, 6, 7, 8, 9\}$. Each client holds a local dataset associated with one of the label spaces mentioned above, including the training dataset and test dataset. The number of data samples for each label is set randomly.

Models. We adopt a 2-layer CNN as the local model architecture for all clients. The CNN model takes a 2D image with a shape of $(28, 28, 1)$ as its input. The first convolutional layer generates 32 feature maps using a kernel size of $3x3$. The second convolutional layer produces 64 feature maps with the same kernel size. These two layers are followed by a maxPooling layer, which reduces the spatial dimensions of the feature maps by taking the maximum value within a $2x2$ window with a stride of 2. Subsequently, two dropout layers are applied ($p = 0.25$ and $p = 0.5$, respectively), each followed by a fully connected layer.

Participation Rate. There are 40 clients, and only a portion will participate in the training process during each iteration. Each client is assigned a predefined participation rate, randomly determined, which governs its participation routine. We categorize clients' participation rates into three levels: lazy ($\phi = 0.2$), normal ($\phi = 0.5$), and diligent ($\phi = 1.0$).

Computation Ability. The server is assumed to wait for a specific period to collect gradients from clients who complete at least one epoch of local training. Over a set timeframe, clients possessing superior computation ability can complete more local training epochs. For simplification, we define a random number of local training epochs for each client, ranging from 1 to 3, to differentiate their computation abilities.

5.2 Experimental Results

Effect of Participation Rate to Clients' Model Performance. The quality score metric contains three dimensions, with clients' training dataset size and computation ability being commonly recognized factors. However, the participation rate is seldom discussed and analyzed as a significant factor influencing clients' quality. Therefore, we compared clients with different participation rates

Algorithm 2: M2F Framework: *ServerSide*

Input: clients' gradients $\{g_i^t\}_{\forall i \in P^t}$, training iteration set T
Output: aggregated gradients $\{\hat{g}_i^{t-}\}_{\forall i \in P^t}$

Phase 1: Initialization
Randomly select a subset of clients with ratio λ ($default = 10\%$);
Aggregate weights received from clients to obtain an initial model;
Broadcast the initial model to all clients in N;
foreach *iteration* $t \in T$ **do**
  Receive gradients g_i^t from client $i, \forall i \in P^t$;
  Phase 2: Hierarchical Clustering
  Take each client as a cluster;
  while *distance threshold* β^t *not reached* **do**
    Calculate pairwise gradient cosine distance among clusters using Eq. (8);
    Merge two clusters with the minimum distance $\min_{\forall i,j \in P^t, i \neq j} R_i, j$;
  end
  Phase 3: Quality Score Calculation
  foreach *cluster* $k \in C^t$ **do**
    Calculate clients participation rate $\{\phi_i^t\}_{\forall i, c_i = k}$ using Eq. (9);
    Calculate clients computation ability $\{\xi_i^t\}_{\forall i, c_i = k}$ using Eq. (10);
    Calculate clients dataset size ratio $\{\delta_i^t\}_{\forall i, c_i = k}$ using Eq. (11);
    Calculate clients quality score $\{q_i^t\}_{\forall i, c_i = k}$ using Eq. (12);
    Aggregate clients' gradients $\{g_i^t\}_{\forall i, c_i = k}$ using quality score as weights and obtain $\hat{g}_k^t$;
  end
  Phase 4: Aggregated Gradient Allocation
  foreach *client* $i \in P^t$ **do**
    Calculate the number of parameters client i can download using Eq. (6);
    Sort and allocate the aggregated gradient $\hat{g}_{c_i}^t$ using Eq. (5);
    Send back gradient $\hat{g}_i^{t-}$ to client i;
  end
end
return aggregated gradient $\{\hat{g}_i^{t-}\}_{\forall i \in P^t}$.

in this experiment by evaluating their model accuracy on the local validation dataset. We use the FedAvg algorithm as the baseline.

Figure 2 displays three clients (index 0, 6, and 13) sharing the same label space ($\{1, 2, 3, 4, 5\}$) and training dataset size. Additionally, they undergo the same number of local training epochs ($e_i = 3$) in each iteration of participation. The predefined participation rates for these clients are 0.2, 0.5, and 1.0, respectively. Accordingly, we generate a participation routine for each client, ensuring the total number of participation iterations equals the total iteration count ($T = 100$) times their respective predefined participation rate. This setup also applies to clients in Fig. 3 (index 20, 26, 33), whose label space is ($\{0, 6, 7, 8, 9\}$). In Fig. 2, our M2F framework demonstrates that the local model of client 13 outperforms those of client 6 and client 0 in terms of higher convergence speed and validation accuracy after 100 iterations. Remarkably, client 0 fails to con-

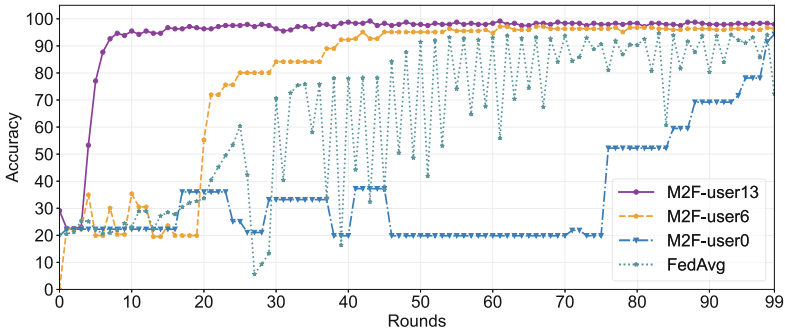

Fig. 2. Validation accuracy of models on label space $\{1, 2, 3, 4, 5\}$.

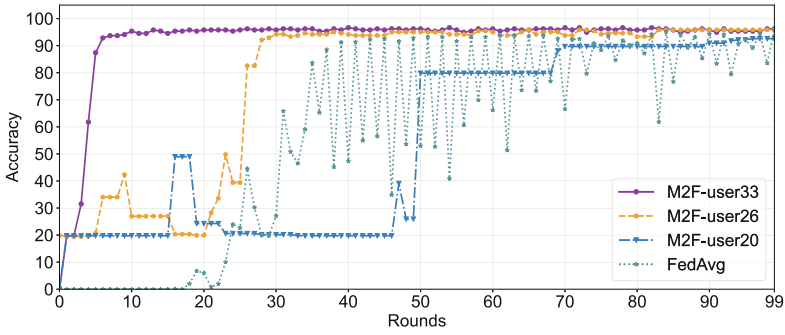

Fig. 3. Validation accuracy of models on label space $\{0, 6, 7, 8, 9\}$.

verge even after 100 training iterations. In Fig. 3, local model client 33 achieves the best performance among the three clients, while client 20 has a lower convergence speed and validation accuracy. By comparing the performance of clients in both Fig. 2 and Fig. 3, we observe that M2F framework effectively distinguishes clients with different participation rates and mitigates the negative effects of less active clients on the more diligent ones.

In FedAvg, clients' gradients are aggregated regardless of their heterogeneous label spaces. Furthermore, clients of varying participation rates are rewarded with the same global model. Figure 2 illustrates the global model accuracy, evaluated using a test dataset with the label space ($\{1, 2, 3, 4, 5\}$). Figure 3 demonstrates the global model accuracy evaluated on the test dataset with label space ($\{0, 6, 7, 8, 9\}$). As shown in both figures, strong oscillation appears in the global model performance for two label spaces. Additionally, the global model accuracy consistently remains lower than that of clients with high and medium participation rates trained using M2F framework.

Table 2. Experiment results of partial clients with label {1,2,3,4,5}

User index	Local epoch	Dataset size	Participation rate	Average accuracy	Convergent iteration	Iteration reach 90%
1	1	1071	0.2	94.78	45	43
5	2	1369	0.2	94.94	36	31
9	1	995	0.5	92.71	64	60
12	2	**1498**	1	**96.65**	13	13
14	1	1396	1	96.54	23	17
16	**3**	1036	**1**	95.09	**11**	**11**

Effect of Multiple Quality Factors to Clients' Model Performance. In the following experiment, the local training epochs e_i are random numbers ranging from 1 to 3, and the clients' dataset size is a portion (randomly selected from 0.5 to 1.0) of the one set in the previous experiment. Table 2 and Table 3 present the average accuracy achieved after convergence within 100 iterations, the number of iterations before convergence, and the number of iterations before clients' model accuracy reaches 90% of clients with differences in the three quality evaluation factors. In our framework, the client quality score is calculated by multiplying three metrics: computation ability, data ratio, and participation rate, which are all normalized ratios among clients in the same cluster. Since clients may belong to different clusters in each iteration, the quality score of clients in a certain iteration is not able to describe clients' objective settings. Therefore, we list the three factors that determine the quality score instead. In Table 2, compared to other clients, client 9 holds the least training dataset and the worst computation ability, while its participation rate is at a medium level. As a result, it takes client 9 the most number of iterations to achieve the lowest average model accuracy. As for client 16, who participates in each iteration and trains the maximum number of epochs locally on a medium-level dataset volume, its local model converges to high accuracy in the early stage of the learning process. If we fix the participation rate and decrease the local epoch number but increase

Table 3. Experiment results of partial clients with label {0,6,7,8,9}

User index	Local epoch	Dataset size	Participation rate	Average accuracy	Convergent iteration	Iteration reach 90%
21	**3**	1064	0.2	95.32	16	16
24	1	**1351**	0.2	95.57	43	36
29	2	988	0.5	**95.63**	28	12
35	**3**	1195	**1**	94.81	**9**	**9**
37	**3**	955	**1**	93.70	11	**9**
39	**3**	1084	**1**	93.67	**9**	**9**

the dataset size, clients 12 and 14 can still achieve a relatively high average accuracy after more iterations.

In Table 3, clients 35, 37, and 39 differ only in dataset size. Client 35, holding the maximum values in all the factors, converges to the highest accuracy within the least time. As for clients 37 and 39, it is shown that by decreasing clients' dataset size, clients can still converge to a similar average accuracy with more iterations. If the local epoch and participation rate are both low, as shown by client 24, the client's convergent iteration number increases significantly. In conclusion, clients' average accuracy and convergent iteration number are affected by the three factors in a comprehensive way. The negative effect caused by decreasing one factor may be complemented by increasing the other two factors. However, the dropping of two or more factors certainly leads to a lower accuracy and longer time to converge. In this way, clients of different quality can be distinguished by M2F framework in model performance.

6 Conclusion

This paper introduces a Multi-centered Fairness-aware Federated Learning framework (M2F) designed to address fairness issues in client contribution evaluation and incentive allocation in federated learning. M2F framework first applies hierarchical clustering to separate clients based on their model similarities. A multi-dimensional metric is proposed to evaluate the quality and customize rewards for clients with a variant of aggregated gradients. Experimental results demonstrated that M2F can provide appropriate feedback to clients according to their data distribution and comprehensive quality, which can consequently influence the speed of convergence and model accuracy. By employing M2F, the detrimental effects of discrepancies in data distribution and quality among clients are mitigated, enabling higher-quality clients to achieve superior convergence speed and accuracy.

References

1. Jia, R., et al.: Towards efficient data valuation based on the shapley value. In: The 22nd International Conference on Artificial Intelligence and Statistics, pp. 1167–1176. PMLR (2019)
2. Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S., Stich, S., Suresh, A.T.: Scaffold: stochastic controlled averaging for federated learning. In: International Conference on Machine Learning, pp. 5132–5143. PMLR (2020)
3. Li, Q., He, B., Song, D.: Model-contrastive federated learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10713–10722 (2021)
4. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. Proc. Mach. Learn. Syst. **2**, 429–450 (2020)
5. Liu, Z., Chen, Y., Yu, H., Liu, Y., Cui, L.: Gtg-shapley: efficient and accurate participant contribution evaluation in federated learning. ACM Trans. Intell. Syst. Technol. (TIST) **13**(4), 1–21 (2022)

6. Lyu, L., Xu, X., Wang, Q., Yu, H.: Collaborative fairness in federated learning. Federated Learn. Priv. Incent. 189–204 (2020)
7. Lyu, L., et al.: Towards fair and privacy-preserving federated deep models. IEEE Trans. Parallel Distrib. Syst. **31**(11), 2524–2541 (2020)
8. McMahan, B., Moore, E., Ramage, D., Hampson, S., Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics, pp. 1273–1282. PMLR (2017)
9. Murtagh, F., Legendre, P.: Ward's hierarchical agglomerative clustering method: which algorithms implement ward's criterion? J. Classif. **31**, 274–295 (2014)
10. Shi, Z., et al.: FedFAIM: a model performance-based fair incentive mechanism for federated learning. IEEE Trans. Big Data (2022)
11. Song, T., Tong, Y., Wei, S.: Profit allocation for federated learning. In: 2019 IEEE International Conference on Big Data (Big Data), pp. 2577–2586. IEEE (2019)
12. Song, Z., Sun, H., Yang, H.H., Wang, X., Zhang, Y., Quek, T.Q.: Reputation-based federated learning for secure wireless networks. IEEE Internet Things J. **9**(2), 1212–1226 (2021)
13. Wang, T., Rausch, J., Zhang, C., Jia, R., Song, D.: A principled approach to data valuation for federated learning. Federated Learn. Priv. Incent. 153–167 (2020)
14. Wei, S., Tong, Y., Zhou, Z., Song, T.: Efficient and fair data valuation for horizontal federated learning. Federated Learn. Priv. Incent. 139–152 (2020)
15. Xu, X., Lyu, L.: A reputation mechanism is all you need: Collaborative fairness and adversarial robustness in federated learning. arXiv preprint arXiv:2011.10464 (2020)
16. Xu, X., Wu, Z., Foo, C.S., Low, B.K.H.: Validation free and replication robust volume-based data valuation. Adv. Neural. Inf. Process. Syst. **34**, 10837–10848 (2021)
17. Yu, H., et al.: A fairness-aware incentive scheme for federated learning. In: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, pp. 393–399 (2020)
18. Zeng, R., Zhang, S., Wang, J., Chu, X.: FMore: an incentive scheme of multi-dimensional auction for federated learning in MEC. In: 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS), pp. 278–288. IEEE (2020)
19. Zhang, J., Li, C., Robles-Kelly, A., Kankanhalli, M.: Hierarchically fair federated learning. arXiv preprint arXiv:2004.10386 (2020)
20. Zhang, J., Wu, Y., Pan, R.: Incentive mechanism for horizontal federated learning based on reputation and reverse auction. In: Proceedings of the Web Conference 2021, pp. 947–956 (2021)
21. Zhao, B., Liu, X., Chen, W.N.: When crowdsensing meets federated learning: privacy-preserving mobile crowdsensing system. arXiv preprint arXiv:2102.10109 (2021)
22. Zhu, Z., Hong, J., Zhou, J.: Data-free knowledge distillation for heterogeneous federated learning. In: International Conference on Machine Learning, pp. 12878–12889. PMLR (2021)