A Joint Gradient and Loss Based Clustered Federated Learning Design

Licheng Lin*, Zhaohui Yang‡, Yusen Wu†, Yuchen Liu§, Mingzhe Chen*†,
*Department of Electrical and Computer Engineering, University of Miami, Coral Gables, FL, 33146, USA
‡College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, 310027, China,
†Frost Institute for Data Science and Computing, University of Miami, Coral Gables, FL, 33146, USA,
§Department of Computer Science, North Carolina State University, Raleigh, NC, 27695, USA
Emails: {lxl1293, yxw1259, mingzhe.chen}@miami.edu, yang_zhaohui@zju.edu.cn, yuchen.liu@ncsu.edu

GLOBECOM 2024 - 2024 IEEE Global Communications Conference | 979-8-3503-5125-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/GLOBECOM52923.2024.10901570

***Abstract*—In this paper, a novel clustered FL framework that enables distributed edge devices with non-IID data to independently form several clusters in a distributed manner and implement FL training within each cluster is proposed. In particular, our designed clustered FL algorithm must overcome two challenges associated with FL training. First, the server has limited FL training information (i.e., the parameter server can only obtain the FL model information of each device) and limited computational power for finding the differences among a large amount of devices. Second, each device does not have the data information of other devices for device clustering and can only use global FL model parameters received from the server and its data information to determine its cluster identity, which will increase the difficulty of device clustering. To overcome these two challenges, we propose a joint gradient and loss based distributed clustering method in which each device determines its cluster identity considering the gradient similarity and training loss. The proposed clustering method not only considers how a local FL model of one device contributes to each cluster but also the direction of gradient descent thus improving clustering speed. By delegating clustering decisions to edge devices, each device can fully leverage its private data information to determine its own cluster identity, thereby reducing clustering overhead and improving overall clustering performance. Simulation results demonstrate that our proposed clustered FL algorithm can reduce clustering iterations by up to 99% compared to the existing baseline.**

***Index Terms*—clustered federated learning, gradient and loss based distributed clustering,**

I. Introduction

The development of mobile devices and video streaming applications (i.e., metaverse and virtual reality) motivates the development of distributed learning frameworks where devices can train their models locally using their own data [1]–[4]. Federated learning (FL) [5] is a such decentralized learning algorithm that allows devices to collaboratively learn a shared machine learning (ML) model while keeping their data localized on their own devices [6]. However, standard FL may not be applied for devices with non independent and identically distributed (non-IID) data since a standard FL method directly aggregates the ML models of devices without considering the data distributions of devices. To address this problem, one promising solution is to cluster the devices according to their data distributions such that the devices in a cluster with similar data distributions can collaboratively train a ML model thus solving the non-IID problem and improving training performance. However, designing clustered FL algorithms still presents several challenges including: 1) The parameter server (PS) has limited information (i.e., FL model parameters) to determine cluster identities of all devices. 2) The PS has limited computational resource to identify differences among a large number of devices.

Recently, a number of existing works such as in [7]–[15] have studied the design and deployment of clustered FL over wireless networks. In particular, the authors in [7] designed a clustered FL algorithm that first trains local models on each device, and then uses clustering algorithms such as k-means to cluster devices according to their locally trained convergent models. The work in [8] developed a FL algorithm with hierarchical clustering approach. The designed algorithm first trains a global model over several FL training iterations and then clusters devices according to the similarities between updated local FL models. The authors in [9] designed a clustered FL framework in which an original cluster containing all devices is recursively divided into smaller sub-clusters. The device clustering starts when the FL models are stationary and ends when the gradient norm of any devices in the sub-cluster is below a preset threshold value. The work in [10] designed a novel clustered FL which integrates the clustering algorithm into the training procedure, and to iteratively adjust the devices' cluster identities through FL process. In [11], the authors investigated clustered FL under Byzantine attacks and shows that clustered FL can reliably detect and remove malicious clients. The authors in [12] introduced a clustering algorithm based on social awareness for clustered FL and developed a heuristic algorithm to minimize the training time per FL iteration. Meanwhile, the designed clustering method in [12] can eliminate the need of a centralized PS. The work in [13] designed a device selection approach for clustered FL to accelerate the convergence rate. In [14], a three-phased clustering algorithm based on generative adversarial network is introduced. The designed clustering method can create dynamic clusters and change the number of clusters over different iterations. However, most of these existing works [7]–[15] focused on the design of centralized clustering methods which may lead to significant communication and computational overhead. Meanwhile, these works [7]–[15] considered the use

This work was supported by the U.S. National Science Foundation under Grants CNS-2332834, SaTC-2350076, CNS-2312138, and SaTC-2350075.

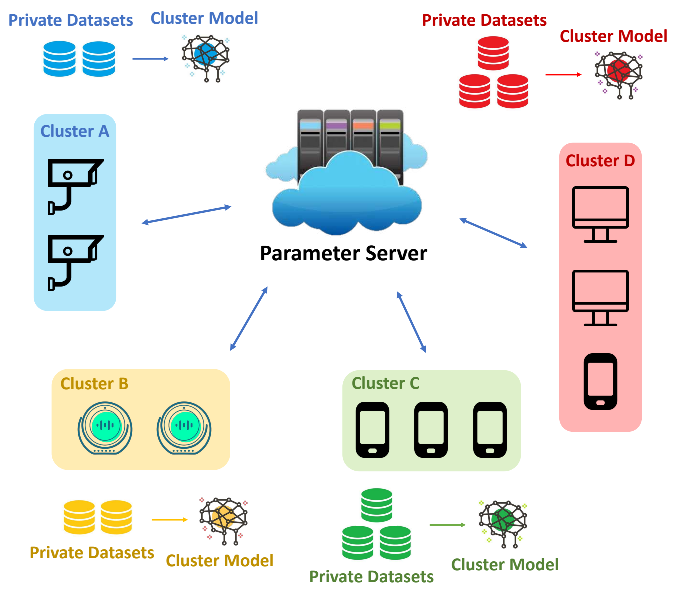


Fig. 1: A Framework of Clustered FL

of only local loss values of edge devices for device clustering without using other information (i.e., gradient vectors) of FL training.

The main contribution of this paper is a novel clustered FL framework that enables distributed edge devices with non-IID data to independently form several clusters in a distributed manner and implement FL training within each cluster. In particular, our designed clustered FL algorithm must overcome two challenges associated with FL training. First, the server has limited FL training information (i.e., the PS can only obtain the FL model information of each device) and limited computational power for finding the differences among a large amount of devices. Second, each device does not have the data information of other devices for device clustering and can only use global FL model parameters received from the server and its data information to determine its cluster identity, which will increase the difficulty of device clustering. To overcome these two challenges, we propose a joint gradient and loss based distributed clustering method in which each device determines its cluster identity considering the gradient similarity and training loss. The proposed clustering method not only considers how a local FL model of one device contributes to each cluster but also the direction of gradient descent thus improving clustering speed. By delegating clustering decisions to edge devices, each device can fully leverage its private data information to determine its own cluster identity, thereby reducing clustering overhead and improving overall clustering performance. Simulation results over multiple datasets demonstrate that our proposed clustered FL algorithm can reduce the iterations required to cluster the devices correctly by up to 99% compared to the existing baseline.

II. Proposed Clustered FL System

Consider a clustered federated learning framework in which one parameter server and a set $\mathcal{M}$ of M devices collaboratively perform federated learning algorithms. In our model, devices have different datasets and hence the data distribution of the devices is non-IID. We assume that the total number of data distributions of all devices is K. To address the data heterogeneity problem [16], devices should be divided into K clusters based on the characteristics of their datasets. The devices with similar data distributions are clustered into a group and jointly perform an FL training. In our model, we consider a general scenario where each device does not know the data distribution of other devices and the PS also does not know the data distributions of all devices. Hence, the PS cannot directly determine the cluster of each device and each device must use its limited FL parameter information to determine its cluster. To this end, it is necessary to design a novel clustered FL method where each device exploits its FL parameter information to determine its cluster individually. Next, we introduce our designed clustered FL algorithm. In particular, we first discuss the general process of clustered federated learning and then provide more details about the proposed clustering algorithm in clustered FL.

A. *General Procedure of Clustered FL*

Here, we introduce the general training process of clustered FL, which is summarized as follows:

1) The server randomly initializes K FL models at first training iteration and broadcast the parameters of these FL models to all devices. We assume that w_k^t represents the FL model parameters of cluster k at iteration t. Here, the set of devices at each group k may be changed according to the clustering results.
2) Each device $i \in \mathcal{M}$ determines its cluster identity, i.e., which cluster it belongs to, via its private dataset and the model parameters received from the PS. Since this cluster identity would change through FL process, we denote the cluster identity of device i at t-th iteration as s_i^t. Given its cluster identity s_i^t, each device will update its local FL model and transmit its FL parameters and cluster identity to the PS.
3) The PS will aggregate the FL parameters with the same cluster identity and generate a global FL model. Since the devices are divided into K clusters, the PS will generate K global FL models.
4) Repeat Steps 2-4 until converge.

From the training process of clustered FL, we see that clustered FL requires each device to use only its dataset and global FL models received from the PS to identify cluster identities and each device does not know the data distribution and cluster identify. Devices need to determine their clustering identities per iteration.

B. *Proposed Clustered FL Algorithm*

Given the general process of clustered FL, in this subsection, we introduce our proposed clustered FL, which also consists of four steps: *1)* cluster FL model broadcast, *2)* device cluster identity determination, *3)* local FL model update, and *4)* local FL model aggregation, which are specified as follows.

1) Initialization: The server randomly initializes K FL models. Each device initializes its cluster identity randomly at the first FL training round, since each device does not have any information to determine the cluster identity.

2) Cluster model broadcast: Since the devices are grouped into K clusters, the server will generate K initial global FL models for all clusters. Hence, to implement our proposed clustered FL, the server will first broadcast the parameters of K global FL models $\{\boldsymbol{w}_1^t, \boldsymbol{w}_2^t, \ldots, \boldsymbol{w}_K^t\}$ to the devices.

3) Determination of cluster identity for each device: Given the training process of clustered FL, two challenge must be solved when we design the device clustering algorithm. First, the device clustering method must be distributed since the server has limited FL training information (i.e., the PS can only obtain the FL model information of each device) and limited computational power for finding the differences among a large amount of devices. Second, each device does not have the data information of other devices for device clustering and can only use global FL model parameters received from the server and its data information to determine its cluster identity, which will increase the difficulty of device clustering. To overcome these two challenges, we propose a joint gradient and loss based distributed clustering method that consists of four steps: 1) Loss calculation, 2) Back-propagation, 3) Similarity calculation, and 4) Cluster identity determination, which are specified as follows:

Step 1: Loss calculation Given the parameters of K FL models, $\{\boldsymbol{w}_1^t, \boldsymbol{w}_2^t, \ldots, \boldsymbol{w}_K^t\}$, device i first calculates the loss with respect to each global FL model using a mini-batch of local data samples $\mathcal{Z}_i^t$, as follows:

$$\mathcal{L}_{i,k}^t(\mathcal{Z}_i^t) = \sum_{\boldsymbol{z} \in \mathcal{Z}_i^t} l(\boldsymbol{w}_k^t, \boldsymbol{z}), \forall k = 1, 2, \ldots, K. \tag{1}$$

where $\boldsymbol{z}$ is a single sample in $\mathcal{Z}_i^t$, and $l(\boldsymbol{w}_k^t, \boldsymbol{z})$ is the loss value of model $\boldsymbol{w}_k^t$ with data sample $\boldsymbol{z}$.

Step 2 Back-propagation: Next, device i can calculate the gradients of K FL models based on the loss values obtained in the first step via back-propagation algorithm. In particular, we assume that the gradient of loss function $\mathcal{L}_{i,k}^t(\mathcal{Z}_i^t)$ with respect to the global FL model $\boldsymbol{w}_k^t$ at device i is $\nabla\mathcal{L}_{i,k}^t(\mathcal{Z}_i^t), \forall k = 1, 2, \ldots, K$

Step 3 Similarity calculation: The gap between the global FL model $\boldsymbol{w}_k^t$ of cluster k at iteration t and the global FL model $\boldsymbol{w}_k^{t-1}$ of cluster k at iteration $t-1$ is

$$\Delta\boldsymbol{w}_k^{t-1} = \boldsymbol{w}_k^t - \boldsymbol{w}_k^{t-1}, \tag{2}$$

In (2), $\Delta\boldsymbol{w}_k^{t-1}$ is the average gradient of all devices in cluster k at iteration $t-1$. The similarity between the local gradient $\nabla\mathcal{L}_{i,k}^t(\mathcal{Z}_i^t)$ and $\Delta\boldsymbol{w}_k^{t-1}$ is calculated by

$$S_{i,k}^t = \frac{\left(\nabla\mathcal{L}_{i,k}^t(\mathcal{Z}_i^t)\right) \cdot \Delta\boldsymbol{w}_k^{t-1}}{|\nabla\mathcal{L}_{i,k}^t||\Delta\boldsymbol{w}_k^{t-1}|}, \forall k = 1, 2, \ldots, K. \tag{3}$$

In (3), we use cosine similarity to characterize the similarity between local gradient and the latest global FL model update, which ignores the magnitude of gradient values and focuses on the direction of gradient descent. We can also use other functions to characterize the similarity between local gradient and the latest global FL model update. For example, if we consider both the magnitude and direction, we can use euclidean metric and treat the inverse of distance as similarity (i.e. $S_{i,k}^t = -\boldsymbol{d}(\nabla\mathcal{L}_{i,k}^t, \Delta\boldsymbol{w}_k^{t-1}) = -||\nabla\mathcal{L}_{i,k}^t - \Delta\boldsymbol{w}_k^{t-1}||, \forall k = 1, 2, \ldots, K$.)

Step 4 Cluster identity determination: Given (3), the cluster identity is estimated by

$$s_i^t = \underset{k=1,2,\ldots,K}{\operatorname{argmax}} \left(\lambda S_{i,k}^t + (1-\lambda)(-\mathcal{L}_{i,k}^t)\right). \tag{4}$$

where λ is a weight parameter that controls the importance of the gradient similarity and the training loss for the cluster identification. From (3), we see that the cluster identify of each device depends on the gradient similarity and the training loss.

4) Local model update: Given cluster identity s_i^t, device i updates its local model as

$$\boldsymbol{w}_i^{(t+1)} = \boldsymbol{w}_{s_i^t}^t - \alpha\nabla\mathcal{L}_{i,s_i^t}^t, \tag{5}$$

where α is the learning rate. Then, device i transmits its updated FL model parameters $\boldsymbol{w}_i^{(t+1)}$ and the cluster identity s_i^t to the PS.

5) Local FL model aggregation: The uploaded models with the same cluster identity s_i^t are aggregated by the PS so as to generate a global model of cluster s_i^t. Denote the set of devices identified as cluster k as

$$\mathcal{M}_k = \{i | i \in \mathcal{M}, s_i^t = k\}. \tag{6}$$

The global model aggregation of cluster k can be represented as

$$\boldsymbol{w}_k^{(t+1)} = \frac{1}{|\mathcal{M}_k|} \sum_{i \in \mathcal{M}_k} \boldsymbol{w}_i^{(t+1)} \tag{7}$$

The full procedure of our proposed clustered FL algorithm is summarized in Algorithm 1.

C. Empty Cluster Problem in Cluster FL Training

To implement the proposed clustered FL, we may also need to solve a problem where one cluster may not have any devices during the training of clustered FL. This is because all the devices in this cluster may be misclassified into other clusters. Although this scenario may not happen frequently, it will significantly reduce the performance of clustered FL. In particular, at one clustered FL training iteration, if one cluster does not have any devices, the FL model of this cluster will not be updated. When the FL model is not updated, the gradient vector calculated by each device may not be correct such that the device will not select the cluster without updated FL model in the following iterations. In consequence, the number of clusters considered in our algorithm will be reduced. To address this issue, we can randomly select K devices and allocate one device to one cluster. Here, K devices can have the same data distributions and we only need to make sure that each cluster will have one device per FL training iteration. Our simulations results in the following section show that this random device allocation method (i.e., each cluster has one

Algorithm 1 Proposed Clustered Federated Learning

```
1: Input: number of clusters $K$, number of clustering iterations $T$, number of devices $M$, set of devices $\mathcal{M}$, learning rate $\lambda$, $K$ initial cluster models $\{\boldsymbol{w}_1^{(0)}, \boldsymbol{w}_2^{(0)}, \ldots, \boldsymbol{w}_K^{(0)}\}$.
2: Initialization: The server initializes $K$ FL models and each device initializes its cluster identity randomly.
3: for $t = 0, 1, \ldots, T-1$ do
4:    server: broadcast $\{\boldsymbol{w}_1^t, \boldsymbol{w}_2^t, \ldots, \boldsymbol{w}_K^t\}$ to all devices.
5:    for device $i \in \mathcal{M}$ in parallel do
6:       for $k = 1, 2, \ldots, K$ do
7:          Calculate loss $\mathcal{L}_{i,k}^t$ by (1).
8:          Obtain gradient $\nabla\mathcal{L}_{i,k}^t$ via back-propagation.
9:          Calculate gradient similarity $S_{i,k}^t$ by (3).
10:      end for
11:      Estimate cluster identity by (4).
12:      Update the local model by (5).
13:      Upload $s_i^t$ and $\boldsymbol{w}_i^t$ to the server.
14:   end for
15:   server: aggregate received models of each cluster based on (7).
16: end for
17: Output: global cluster models $\{\boldsymbol{w}_1^t, \boldsymbol{w}_2^t, \ldots, \boldsymbol{w}_K^t\}$ and the cluster identities of devices $\{s_1^t, s_2^t, \ldots, s_M^t\}$.
```

device that will not change its cluster) will only introduce at most one incorrect device classification.

III. Simulation Results and Analysis

We consider the implementation of the proposed clustered FL for learning tasks:1) 10 class hand written digits identification (i.e., MNIST [17]), 2) 10 class fashion product images classification (i.e. FashionMNIST [18]), 3) 10 class objects classification (i.e., CIFAR10 [19]) , and 4) 62 class hand written letters and digits identification (i.e., EMNIST [20]) . To evaluate the performance, we run our proposed clustered FL method in 4 experiments, each of which extracts its cluster task datasets from a classification dataset. For comparison purpose, we use the iterative clustered FL scheme from [10] as baseline.

A. Simulation Settings and Performance Metrics

Here, we first explain how to generate the dataset for the devices in each cluster of each learning task. Then, we introduce the local FL model settings for each learning task. Finally, we describe the performance metrics used in the simulations.

1) Dataset Settings: For the experiments on MNIST, FashionMNIST, and CIFAR10, we consider 80 devices jointly implement the clustered FL algorithm. These devices are equally divided into 4 clusters (i.e., clusters A, B, C, and D as shown in Fig. 2) and each cluster has 20 devices. Each dataset totally has 10 class data and a device in each cluster has 8 class data. Hence the devices in different clusters will have at least 6 overlapped class data. In Fig. 2, we show the data distribution of the four clusters for each learning task. From this figure, we see that, in MNIST, the devices in cluster A have a total of 17500 samples of number 0, 1, 2, 3, 4, 5, 6, 8, while the devices in cluster B have 14500 samples of number 0, 1, 2, 3, 4, 6, 7, 9. Hence, there are 6 overlapped class data between devices in cluster A and B. The data samples of each cluster will be further distributed to its devices equally and randomly.

For EMNIST learning task, we consider the clustered FL is implemented by 200 devices which are divided into 8 clusters. To reduce ambiguity between the uppercase and lowercase forms of some easily hard-to-distinguish letters, we merged the uppercase and lowercase classes for the letters C, I, J, K, L, M, O, P, S, U, V, W, X, Y and Z, such that 62 class data are changed into 47 classes. We further split these classes into clusters in the same manner as was done for the MNIST dataset, each of which has 40 class data.

2) Learning Models: For each learning task, we consider the use of two neural network models as local FL models. The first one is a multi-layer perceptron (MLP) with three fully-connected layers with ReLU activation. The second model is a convolutional neural network (CNN) which consists of two convolutional layers followed by fully-connected layers. The detailed MLP and CNN model architectures are shown in Fig. 3.

3) Performance Metrics: To measure the clustering accuracy of the clustered FL algorithm, we use purity which is defined as the percentage of devices that are classified correctly. The purity P^t at iteration t is mathematically expressed as

$$P^t = \frac{1}{|\mathcal{M}|}\sum_k \max_j |\mathcal{M}_j^* \cap \mathcal{M}_k^t|,$$

where $\mathcal{M}_j^*$ is the ground truth set of devices at cluster j, and $\mathcal{M}_k^t$ is the set of devices that are clustered by the clustered FL algorithm at iteration t, with

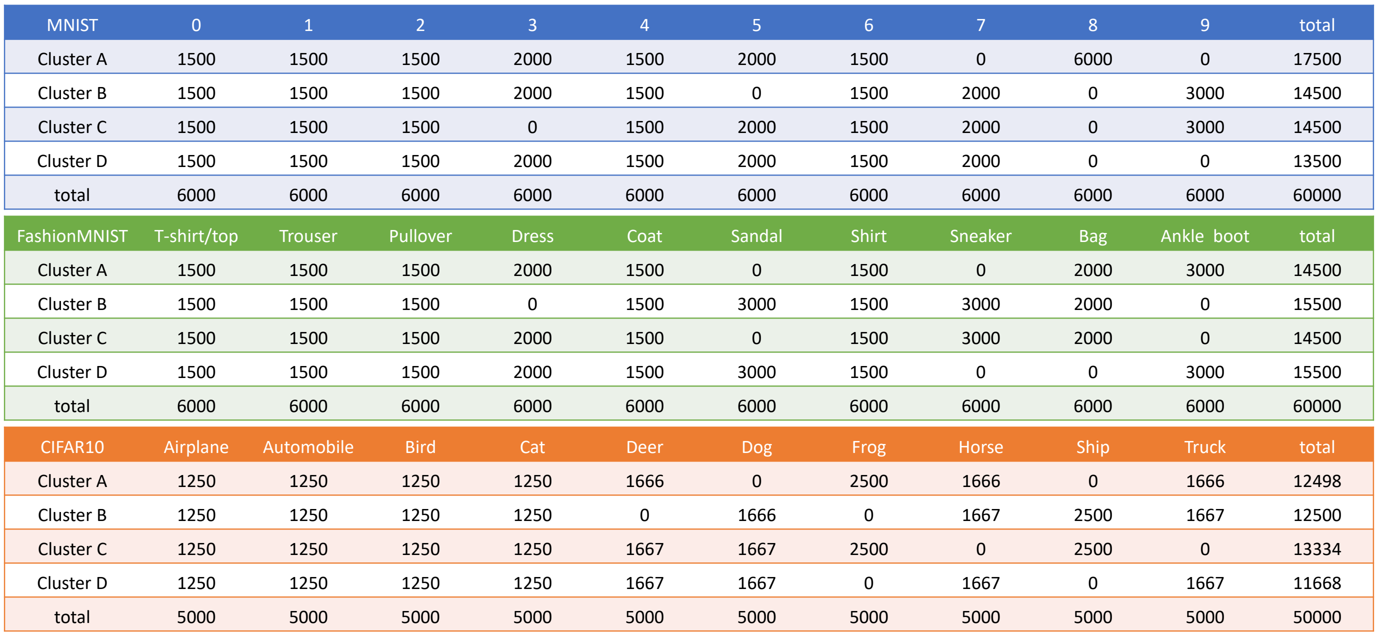
$$\mathcal{M}_k^t = \{i | i \in \mathcal{M}, s_i^t = k\}, \forall k = 1, 2, \ldots, K$$

is the set of devices with the same cluster identity k at iteration t.

In order to demonstrate that proposed algorithm brings better performance to the clustered FL training, we also use test accuracy to measure the training effect of clustered FL. While splitting the training dataset, we also split the test dataset for each user which has the same sample distribution as their training dataset. The total test accuracy of the clustered FL system is obtained by averaging test accuracy of all users.

B. Results Analysis

In Fig. 4, we show how the clustering purity, training loss, and test accuracy vary as the number of training iterations changes. This experiment is implemented over MNIST dataset. Fig. 4(a), Fig. 4(b), and Fig. 4(c) are results of MNIST experiments where MLP is used as FL models, while Fig. 4(d), Fig. 4(e), and Fig. 4(f) are results of experiments where CNN is used as FL models. Figs. 4(a) and 4(d) show how clustering purity changes as the number of iterations increases. From Fig. 4(a), we see that the proposed algorithm with $\lambda = 0.2$ can reduce 99% iterations to achieve 0.9 clustering purity compared to the baseline. This is because the proposed

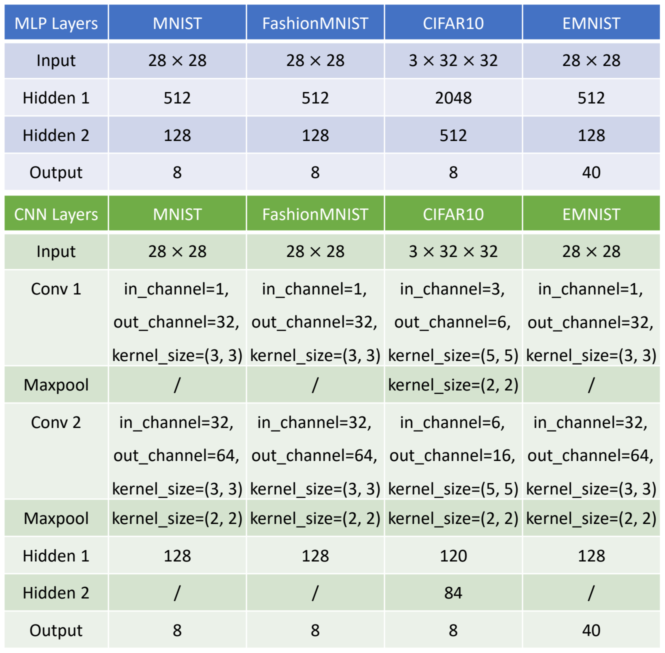


MNIST	0	1	2	3	4	5	6	7	8	9	total
Cluster A	1500	1500	1500	2000	1500	2000	1500	0	6000	0	17500
Cluster B	1500	1500	1500	2000	1500	0	1500	2000	0	3000	14500
Cluster C	1500	1500	1500	0	1500	2000	1500	2000	0	3000	14500
Cluster D	1500	1500	1500	2000	1500	2000	1500	2000	0	0	13500
total	6000	6000	6000	6000	6000	6000	6000	6000	6000	6000	60000

FashionMNIST	T-shirt/top	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot	total
Cluster A	1500	1500	1500	2000	1500	0	1500	0	2000	3000	14500
Cluster B	1500	1500	1500	0	1500	3000	1500	3000	2000	0	15500
Cluster C	1500	1500	1500	2000	1500	0	1500	3000	2000	0	14500
Cluster D	1500	1500	1500	2000	1500	3000	1500	0	0	3000	15500
total	6000	6000	6000	6000	6000	6000	6000	6000	6000	6000	60000

CIFAR10	Airplane	Automobile	Bird	Cat	Deer	Dog	Frog	Horse	Ship	Truck	total
Cluster A	1250	1250	1250	1250	1666	0	2500	1666	0	1666	12498
Cluster B	1250	1250	1250	1250	0	1666	0	1667	2500	1667	12500
Cluster C	1250	1250	1250	1250	1667	1667	2500	0	2500	0	13334
Cluster D	1250	1250	1250	1250	1667	1667	0	1667	0	1667	11668
total	5000	5000	5000	5000	5000	5000	5000	5000	5000	5000	50000

Fig. 2: Example splits of MNIST, FashionMNIST, and CIFAR10



MLP Layers	MNIST	FashionMNIST	CIFAR10	EMNIST
Input	28 × 28	28 × 28	3 × 32 × 32	28 × 28
Hidden 1	512	512	2048	512
Hidden 2	128	128	512	128
Output	8	8	8	40

CNN Layers	MNIST	FashionMNIST	CIFAR10	EMNIST
Input	28 × 28	28 × 28	3 × 32 × 32	28 × 28
Conv 1	in_channel=1, out_channel=32, kernel_size=(3, 3)	in_channel=1, out_channel=32, kernel_size=(3, 3)	in_channel=3, out_channel=6, kernel_size=(5, 5)	in_channel=1, out_channel=32, kernel_size=(3, 3)
Maxpool	/	/	kernel_size=(2, 2)	/
Conv 2	in_channel=32, out_channel=64, kernel_size=(3, 3)	in_channel=32, out_channel=64, kernel_size=(3, 3)	in_channel=6, out_channel=16, kernel_size=(5, 5)	in_channel=32, out_channel=64, kernel_size=(3, 3)
Maxpool	kernel_size=(2, 2)	kernel_size=(2, 2)	kernel_size=(2, 2)	kernel_size=(2, 2)
Hidden 1	128	128	120	128
Hidden 2	/	/	84	/
Output	8	8	8	40

Fig. 3: Model architectures of the MLP and CNN for each experiment

clustered FL algorithm jointly uses gradient direction and loss value to cluster devices. From Fig. 4(a), we also see that when λ changes from 0.1 to 0.5, the proposed algorithm can achieve higher purity at the beginning. This is because the gradient direction can cluster devices better than loss value at the beginning, therefore higher weight for gradient direction brings better performance. Fig. 4(b) and Fig. 4(c) show that the proposed clustered FL algorithm can reduce 14% iterations to achieve 0.8 test accuracy compared to the baseline. Fig. 4(d), Fig. 4(e), and Fig 4(f) show that when CNN is used as FL model, the cluster performance and training efficiency of the proposed algorithm is also better, compared to the baseline. This stems from the fact that the clustering process of the proposed algorithm is more efficient, which accelerates the training of FL.

In Fig. 5, we show how the clustering purity varies as the number of training iteration increases in experiments implemented over FashionMNIST, CIFAR10, and EMNIST. From Fig. 5(a), Fig. 5(b), and Fig. 5(c), we see that the proposed algorithm respectfully reduces iterations required to achieve 0.9 clustering purity by up to 98%, 21% and 97% compared to the baseline. This is because the proposed algorithm can jointly use gradient direction and loss value to cluster devices.

IV. Conclusion

In this paper, we have developed a novel clustered FL framework that enables distributed edge devices with non-IID data to independently form several clusters in a distributed manner and implement FL training within each cluster. In particular, our designed device method considered two unique FL features: 1) limited FL training information and computational power at the PS and 2) each device does not have the data information of other devices for device clustering and can only use global FL model parameters received from the server and its data information to determine its cluster identity. We have proposed a joint gradient and loss based distributed clustering method, in which each device determines its cluster identity considering the gradient similarity and training loss. The proposed clustering method not only considers how a local FL model of one device contributes to each cluster but also the direction of gradient descent thus improving clustering speed. Simulation results over multiple datasets demonstrate that our proposed clustered FL algorithm can yield significant gains compared to the existing method.

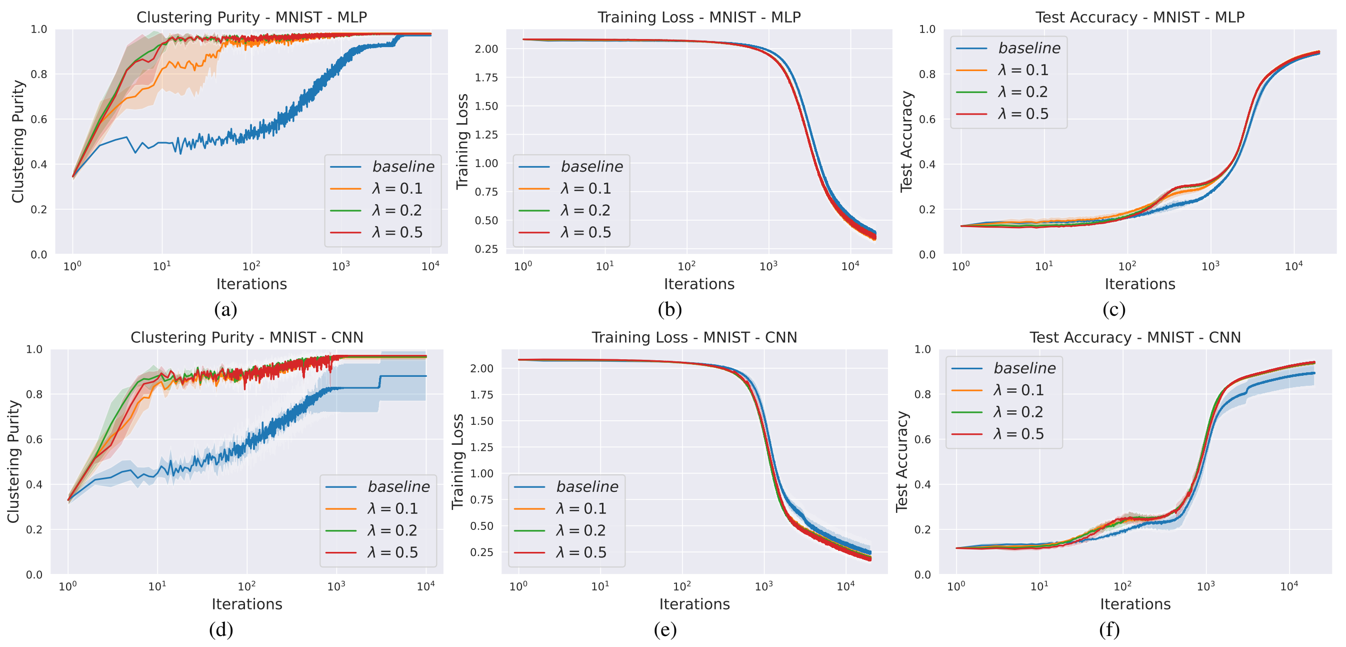


Fig. 4: Performance metrics vary as the number of clustered FL iterations changes on MNIST experiment.

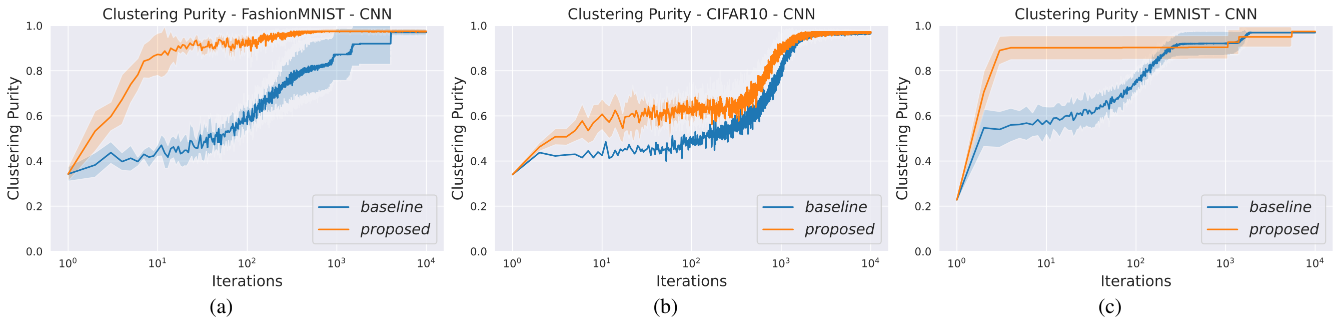


Fig. 5: Clustering Purities vary as the number of iterations increases on FashionMNIST, CIFAR10, EMNIST experiment.

References

[1] M. Chen, D. Gündüz, K. Huang, W. Saad, M. Bennis, A. V. Feljan, and H. V. Poor, "Distributed learning in wireless networks: Recent progress and future challenges," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3579–3605, Oct. 2021.

[2] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 269–283, Oct. 2020.

[3] W. Xu, Z. Yang, D. W. K. Ng, M. Levorato, Y. C. Eldar, and M. Debbah, "Edge learning for B5G networks with distributed signal processing: Semantic communication, edge computing, and wireless sensing," *IEEE J. Sel. Topics Signal Process.*, vol. 17, no. 1, pp. 9–39, Jan. 2023.

[4] Z. Yang, M. Chen, Z. Zhang, and C. Huang, "Energy efficient semantic communication over wireless networks with rate splitting," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 5, pp. 1484–1495, May 2023.

[5] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artificial Intelligence and Statistics*, Fort Lauderdale, FL, USA, April 2017.

[6] M. Chen, N. Shlezinger, H. V. Poor, Y. C. Eldar, and S. Cui, "Communication-efficient federated learning," *Proceedings of the National Academy of Sciences*, vol. 118, no. 17, p. e2024789118, April 2021.

[7] A. Ghosh, J. Hong, D. Yin, and K. Ramchandran, "Robust federated learning in a heterogeneous environment," *arXiv preprint arXiv:1906.06629*, 2019.

[8] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-IID data," in *Proc. International Joint Conference on Neural Networks (IJCNN)*, Glasgow, UK, July 2020.

[9] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 8, pp. 3710–3722, August 2020.

[10] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," *IEEE Transactions on Information Theory*, vol. 68, no. 12, pp. 8076–8091, July 2022.

[11] F. Sattler, K.-R. Müller, T. Wiegand, and W. Samek, "On the byzantine robustness of clustered federated learning," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, May 2020.

[12] L. U. Khan, M. Alsenwi, Z. Han, and C. S. Hong, "Self organizing federated learning over wireless networks: A socially aware clustering approach," in *Proc. International Conference on Information Networking (ICOIN)*, Barcelona, Spain, Jan. 2020.

[13] A. Albaseer, M. Abdallah, A. Al-Fuqaha, and A. Erbad, "Client selection approach in support of clustered federated learning over wireless edge networks," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, Madrid, Spain, Dec. 2021.

[14] Y. Kim, E. Al Hakim, J. Haraldson, H. Eriksson, J. M. B. da Silva, and C. Fischione, "Dynamic clustering in federated learning," in *Proc. IEEE International Conference on Communications*, Montreal, QC, Canada, June 2021, pp. 1–6.

[15] C. Feng, H. H. Yang, D. Hu, Z. Zhao, T. Q. Quek, and G. Min, "Mobility-aware cluster federated learning in hierarchical wireless networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 10, pp. 8441–8458, April 2022.

[16] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[18] H. Xiao, K. Rasul, and R. Vollgraf. (2017) Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms.

[19] A. Krizhevsky, "Learning multiple layers of features from tiny images," Available Online*: https://www.cs.toronto.edu/ kriz/learning-features-2009-TR.pdf*, April 2009.

[20] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, "EMNIST: Extending mnist to handwritten letters," in *Proc. International Joint Conference on Neural Networks (IJCNN)*, Anchorage, AK, USA, May 2017.