Cybersecurity

RESEARCH

Open Access

LayerCFL: an efficient federated learning with layer-wised clustering

Jie Yuan[1,2], Rui Qian[1,2], Tingting Yuan[3*], Mingliang Sun[1,2], Jirui Li[4] and Xiaoyong Li[1,2]

Abstract

Federated Learning (FL) suffers from the Non-IID problem in practice, which poses a challenge for efficient and accurate model training. To address this challenge, prior research has introduced clustered FL (CFL), which involves clustering clients and training them separately. Despite its potential benefits, CFL can be computationally and communicationally expensive when the data distribution is unknown beforehand. This is because CFL involves the entire neural networks of involved clients in computing the clusters during training, which can become increasingly time-consuming with large-sized models. To tackle this issue, this paper proposes an efficient CFL approach called LayerCFL that employs a Layer-wised clustering technique. In LayerCFL, clients are clustered based on a limited number of layers of neural networks that are pre-selected using statistical and experimental methods. Our experimental results demonstrate the effectiveness of LayerCFL in mitigating the impact of Non-IID data, improving the accuracy of clustering, and enhancing computational efficiency.

Keywords Federated learning, Clustered federated learning, Non-IID, Layer-wised clustering

Introduction

The advancements in science and technology have led to the generation of a vast amount of data from daily used intelligent devices, such as smart phone, smart wearable devices, etc. It is projected that global internet users have climbed to 4.95 billion at the start of 2022, with internet penetration now standing at 62.5 percent of the world's total population (Kemp 2022). Global internet users generate huge amounts of data each day, which drives the development of many data-intensive applications, including recommendation systems (Gao et al. 2022) and natural language processing, such as ChatGPT (OpenAI 2022). Additionally, these applications demand a greater volume of data generated by various devices to train machine learning (ML) models with greater precision and accuracy (Zhou et al. 2017; Al-Jarrah et al. 2015). However, training ML models with distributed data is challenging due to the difficulty of collecting the data from various devices. With the introduction of regulations like the *General Data Protection Regulation (GDPR)* (Voigt and Von dem Bussche 2017), privacy protection in ML has become a significant area of research focus (Papernot et al. 2016; Liu et al. 2021).

Federated Learning (FL) (McMahan et al. 2017, 2016; Kairouz et al. 2021; Li et al. 2020a; Yang et al. 2019a) is a novel distributed ML framework that allows intelligent devices to collaboratively train a shared global model without revealing their local data (Mothukuri et al. 2021), making it a popular choice in finance (Yang et al. 2019b), medicine (Silva et al. 2019), and image vision (Liu et al. 2020). However, it faces practical challenges, including the Non-IID (Not identically and independently distributed) data problem (McMahan et al. 2017; Li et al. 2019; Zhao et al. 2018). For example, varying reading habits

*Correspondence:
Tingting Yuan
tingting.yuan@cs.uni-goettingen.de
[1] School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing, China
[2] Key Laboratory of Trustworthy Distributed Computing and Service (BUPT), Ministry of Education, Beijing, China
[3] Institute of Computer Science, Faculty of Mathematics and Computer Science, University of Goettingen, Göettingen, Germany
[4] School of Information Technology, Henan University of Chinese Medicine, Zhengzhou, Henan, China

and preference among users can lead to Non-IID data in movie recommendation application (Wu et al. 2021), as seen in the case of user A's preference for romantic film and user B's liking for science fiction film. Existing studies (Ma et al. 2022b; Li et al. 2020b; Wang et al. 2019) show that non-IID data can lead to slow convergence, low accuracy, and increased communication costs, highlighting the challenges associated with such data.

To reduce the detrimental effect of non-IID data on joint learning, a framework called *Clustered Federated Learning (CFL)* (Sattler et al. 2020) has been proposed. Some CFL-based methods (Sattler et al. 2020; Ghosh et al. 2020; Gong et al. 2022) group clients based on the similarity of data distribution, enabling them to jointly train the sharing model in group. Most approaches approximate clients' local data distribution using their local models' weight-updates or gradients. But these methods bring excessive computation and communication overhead, resulting in poor clustering accuracy.

However, some other CFL-based methods (Gong et al. 2022), while accounting for differences between model layers, ignore the impact of training on the model under different Non-IID settings. Thus, it is necessary to identify the appropriate method for approximating each client's data distribution. This paper presents LayerCFL, an efficient CFL with Layer-wised clustering, which can reduce the computing cost and improve the clustering accuracy by selecting the appropriate part of the model's layer data to participate in the computing client clustering under different data distribution environments.(R1.-9) Major contributions are briefly summarized as follows:

- As far as we know, this paper is the first to comprehensively analyze the layer-wised impacts of neural networks in CFL clustering in different Non-IID environments.(R1.-9) Based on that, we propose a approach to calculate the similarities in layer-level, which can reflect more accurate similarity among clients than traditional model based CFL methods.
- Moreover, we shed light on how to select the layers given different Non-IID data settings (i.e., the feature distribution skew and label distribution skew).
- Our experiments provide evidence that LayerCFL is effective at clustering clients, requiring fewer communication rounds. Moreover, our findings demonstrate that LayerCFL is well-suited to scenarios where each client has relatively small and large amounts of data.

The remainder of this paper is structured as follows: In Section 2, we provide an overview of related work in the field of CFL and Non-IID. Section 3 details the problem formulation and presents LayerCFL. In Section 4, we describe the experimental settings used to evaluate our approach, and in Section 5, we present the experimental results and analysis. Finally, in Section 6, we conclude our paper and discuss our future plans for research in this field.

Related work

This section provides an introduction to CFL and the different types of Non-IID data. It also introduces the latest research on neural networks' Layer-wised impacts for other domains.

Clustered federated learning and non-IID data

Due to diverse user usage habits, clients participating in FL may have significantly distinct local data distributions, a challenge known as Non-IID data. This issue has garnered considerable attention, hindering the progress of FL (Ma et al. 2022b; Li et al. 2020b; Wang et al. 2019). The classification of Non-IID is a complex and evolving subject with different academic perspectives (Criado et al. 2022; Zhu et al. 2021; Ma et al. 2022b), and most of the literature we surveyed classified Non-IID into five types: (1) feature distribution skew, (2) label distribution skew, (3) same label with different features, (4) same features with different labels, and (5) quantity skew. Among the five Non-IID types, "feature distribution skew" and "label distribution skew" are the two experimental settings that are widely used by researchers while the other three Non-IID types are hardly mentioned (Li et al. 2022). Hence, we choose the above two Non-IID types as the experimental settings in this paper. In the following, we will analyze the related studies on the mitigation of the Non-IID problem (R1.-1)(R1.-6).

CFL-based methods improve the accuracy of the model by aggregating clients with similar data distribution into corresponding clusters to train the model separately. *FMTL* (Sattler et al. 2020) iteratively divides clients into different clusters by calculating cosine similarity between local models of clients until no new clusters are generated. This iterative clustering method requires a critical point to limit the number of clusters, which is related to the final effect.

Some other studies specify the number of clusters in advance through hyperparameters to achieve better clustering results (Ghosh et al. 2020; Mansour et al. 2020). *IFCA* (Ghosh et al. 2020) assigns clients to clusters based on the global model's minimal losses on their local data, while *HypCluster* (Mansour et al. 2020) enables clients to dynamically relocate to the most suitable cluster by running an updated model. But the performance of this type of approach depends on experience, because the number of cluster types must be predetermined.

Existing research also helps clustering clients through the K-means method (Dennis et al. 2021; Yang et al. 2022). *k-FED* (Dennis et al. 2021) utilized the K-means clustering method based on Lloyd's method to calculate the cluster center in a single round of communication. *G-FML* (Yang et al. 2022) method is based on personalized FL and clustering learning, and the accuracy of the model is improved by personalized FL after clustering by the K-means method.

Layer-wised FL for non-IID data

Some existing work computes the model similarity to reflect the data distribution, typically using model-based approaches, such as model weights (Gong et al. 2022) or model weight-updates (Sattler et al. 2020). Nevertheless, as the number of clients and model sizes increase, these approaches tend to place greater workload on the server. Some research has demonstrated differences between different layers of a same model, with higher-layer weights exhibiting greater task-relatedness than lower-layer weights (Yosinski et al. 2014; Zeiler and Fergus 2014).

FedDnC (Chandran et al. 2021) uses an approach based on Federated averaging (FedAvg) that alternates between freezing and learning parts of the model layer to improve the model's effect, similar to transfer learning. *pFedLA* (Ma et al. 2022a) is a layer-wised personalized federated learning, allowing each client to discern the relative importance of each layer with respect to other clients. Although the above works do not directly address CFL, they provide evidence that treating all layers of a neural network equally may not be necessary. This insight can inform the development of CFL approaches that consider the varying importance of different layers in clustering, leading to more efficient and effective CFL.

For CFL, although *FLC* (Kim et al. 2021) aims to improve model performance by independently processing the weights of various layers in the model, it is limited to only performs a single-time classification on the client, dividing it into two clusters. Beside, *AdaCFL* (Gong et al. 2022) can select partial model weights (i.e., fully connected or classifier layers) to cluster clients for label distribution skew. However, *FLC* and *AdaCFL* ignore various Non-IID settings that can arise in FL, which can limit their ability to adapt different real world scenarios.

Problem formulation

This section aims to provide a detailed explanation of the problem formulation for LayerCFL.

Clustered federated learning

FedAvg (McMahan et al. 2017) is widely used to improve communication efficiency among clients in FL with assumption that all client data is IID. Specifically, FedAvg trains the shared model of the server by weighted average aggregation of the model parameters trained locally by clients. We assume that there are M clients, D_m is the dataset on the client m and $n_m = \|D_m\|$ is the number of data, $f_i(\Theta)$ is the loss function of model Θ for some data point i. Then the local empirical risk of client m on data set D_m is defined as:

$$f_m(\Theta) = \frac{1}{n_m} \sum_{i \in D_m} f_i(\Theta). \tag{1}$$

The goal of FedAvg is to minimize the following formula:

$$f(\Theta) = \sum_{m=1}^{M} \frac{n_m}{N} f_m(\Theta), \tag{2}$$

where $N = \sum_{m=1}^{M} n_m$ is the total number of data across all clients. Data generated by clients in a real-world environment is often heterogeneous, i.e., Non-IID, which greatly affects the accuracy of the shared model on the server. CFL (Sattler et al. 2020), a practical framework, has recently gained attention as an effective approach to solving this problem. CFL divides M clients into K clusters $\mathcal{G} = \{g_1, \ldots, g_K\}$, and satisfies the condition that $M \geq K \geq 2$ and $g_i \bigcap g_j = \emptyset (i \neq j)$, where $\Sigma_{i=1}^{K} \|g_i\| = M$. Each cluster g_i optimization target is Eq. (2).

Figure 1 depicts the architecture of a basic CFL system. Clients participating in learning are iteratively clustered into a cluster with clients having similar data distributions. Each cluster collaborates to learn a shared global model, while ensuring that the privacy of their respective local data is maintained throughout the process. CFL computes a model distance matrix to represent the distribution of data stored locally, using model weights or weight updates. Step (a) in the Fig. 1 illustrates that the client transmits the parameters of the model that is trained on the local data to the server. This enables the server to compute the model distance matrix, which is subsequently leveraged to facilitate the clustering process.

Next, we introduce the meaning of the model weight-updates that needs to be uploaded by the client in step (a) at first. We assume that θ_i is the model weights of the client c_i, D_i is the data stored locally in client c_i and θ^t is the t round model weight which is synchronous between client c_i and server. The client c_i performs round t iterations of stochastic gradient descent with minibatches sampled from its local data D_i and the result is:

$$SGD(\theta_i^{t+1}) = SGD(\theta^t, D_i), \tag{3}$$

where θ_i^{t+1} is the model weight obtained by client c_i in round $t+1$ training. According to Eq. (3), we can easily

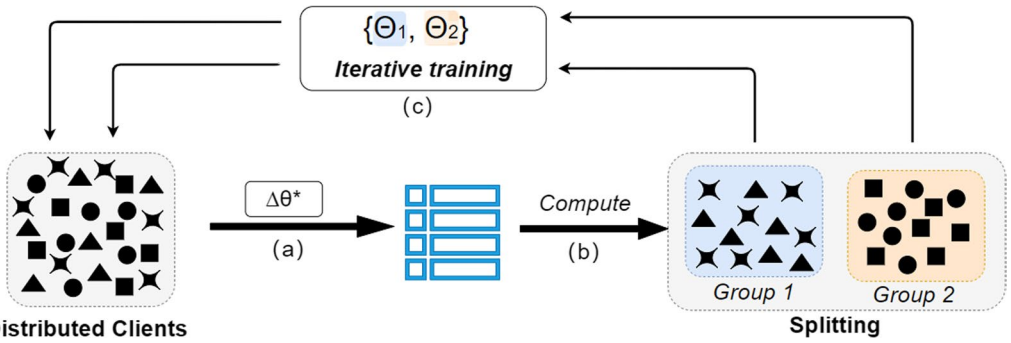


Fig. 1 Overview of CFL framework

conclude that the weight-update of the client c_i in round t is:

$$\Delta\theta_i^{t+1} = SGD(\theta_i^{t+1}) - \theta^t. \tag{4}$$

Following the computation of the server in step (b) of Fig. 1, all participating clients are partitioned into distinct clusters using the clustering algorithm. Each cluster proceeds to share a common global model, which is subsequently distributed to clients within the same cluster to update their local model parameters. The clustering process can be repeated iteratively until a certain convergence criterion is met, beyond which further clustering is deemed unnecessary. In theory, clients with similar data distributions are expected to have similar model weights trained on their data, as opposed to clients with dissimilar data distributions (Sattler et al. 2020; Wang et al. 2020; Ouyang et al. 2021). However, *FMTL* (Sattler et al. 2020) demonstrates a connection between the weight-updates of each client's training round and its data distribution. To quantify the difference between clients with different data distributions, mathematical methods such as l_2 distance and cosine distance are commonly used to measure similarity.

To show the model distance between any two clients c_i and c_j, *AdaCFL* (Gong et al. 2022) choose to use l_2 distance to calculate the similarity of the model weight to obtain the model distance matrix, and the model distance is:

$$l_2 : \ dist(c_i, c_j) = \| \ \theta_i - \theta_j \ \|_{l_2} = \sqrt{\sum_{\vartheta \in \theta} (\vartheta_i - \vartheta_j)^2}, \tag{5}$$

where ϑ is value of the model weight θ.

The definition of the cosine similarity between the weight-updates of clients c_i and c_j is as follows:

$$\Delta\cos : \ dist(c_i, c_j) = \frac{\langle \Delta\theta_i, \Delta\theta_j \rangle}{\| \ \Delta\theta_i \ \| \| \ \Delta\theta_j \ \|}. \tag{6}$$

As mentioned earlier, we can also calculate the cosine distance of the model weights among each clients or the l_2 distance of the model weight-updates. The formula is shown below:

$$\Delta l_2 : \ dist(c_i, c_j) = \| \ \Delta\theta_i - \Delta\theta_j \ \|_{l_2}, \tag{7}$$

$$\cos : \ dist(c_i, c_j) = \frac{\langle \theta_i, \theta_j \rangle}{\| \ \theta_i \ \| \| \ \theta_j \ \|}. \tag{8}$$

Following step (b) of Fig. 1 we get a model distance matrix α, which is obtained by calculating the cosine distance of each clients' model weights or weight-updates. Any two clients c_i and c_j belonging to the same cluster g_k (theoretically their data distribution is the same), and the minimum value between clients in a cluster in the model distance matrix is:

$$\alpha_{\text{intra}}^{\min} := \min_{c_i, c_j \in g_k} dist(c_i, c_j), \tag{9}$$

In the case of splitting the clients into two clusters ($K = 2$), the result of maximum value between clients belonging to different clusters in the model distance matrix simplifies to:

$$\alpha_{\text{cross}}^{\max} := \max_{c_i \in g_x, c_j \in g_y} dist(c_i, c_j), \tag{10}$$

where $g_x \bigcap g_y = \emptyset$.

According to the relevant proof given by Sattler et al. (2020), client i and j have different data distributions, then:

$$\alpha_{\text{cross}}^{\max} < \alpha_{\text{intra}}^{\min} \tag{11}$$

However, in practice, compared to cross-cluster similarity $\alpha_{\text{cross}}^{\max}$, which can be easily computed, the computation of intra-cluster similarity $\alpha_{\text{intra}}^{\min}$ is difficult to obtain because it requires knowledge of the cluster structure. Therefore, we give a more general correct separation theorem(R1.-2):

Theorem 1 *(Correct clustering theorem): Let M clients and their local data sets $D_1, \ldots, D_M$ belong to K different data distributions $\varphi_1, \ldots, \varphi_K$. The empirical risk function on the local model θ for each client i is able to approximate the true risk when the amount of local data is large enough*(*R*1.-2):

$$r_i(\theta) = \sum_{(x,y)\in D_i} los(f(x), y) \tag{12}$$

$$\approx R_{\varphi_i}(\theta). \tag{13}$$

We assume that the data volume of client i is not large enough and θ^* is the target solution of FL, then we have:

$$\left\|R_{\varphi_i}(\theta^*)\right\| > \left\|R_{\varphi_i}(\theta^*) - \nabla r_i(\theta^*)\right\|, \tag{14}$$

and define:

$$\gamma_i := \frac{\left\|R_{\varphi_i}(\theta^*) - \nabla r_i(\theta^*)\right\|}{\left\|R_{\varphi_i}(\theta^*)\right\|} \in [0, 1). \tag{15}$$

Then when M clients are dichotomized into two different clusters g_1 and g_2, because the minimum similarity within a cluster requires intra-cluster knowledge, so $\alpha_{\text{intra}}^{\min}$ can only be estimated by the following equation:

$$\alpha_{\text{intra}}^{\min} := \min_{i,j\in\varphi_k} \alpha(\nabla r_i(\theta^*), \nabla r_j(\theta^*)) \tag{16}$$

$$\geq \min_{i,j\in\varphi_k} H_{i,j}, \tag{17}$$

with

$$H_{i,j} = -\gamma_i\gamma_j + \sqrt{1-\gamma_i^2}\sqrt{1-\gamma_j^2} \in (-1, 1]. \tag{18}$$

Then we can further obtain:

$$\alpha_{\text{intra}}^{\min} \geq -\gamma_i\gamma_j + \sqrt{1-\gamma_i^2}\sqrt{1-\gamma_j^2} \geq 1 - 2\gamma_{max}^2. \tag{19}$$

Therefore, combining Eq. (11) we are able to know that the dichotomy is correct if

$$\sqrt{\frac{1-\alpha_{\text{cross}}^{\max}}{2}} > \gamma_{max}. \tag{20}$$

The proof of the theorem can be given by *CFL*(Sattler et al. 2020) and will not be repeated in this paper.(R1.-2)

Next, we define the *Gap*, which is an important indicator of the correctness of clustering in our experiment, as it can clearly indicate whether the clustering is correct. And *Gap* is defined and used by CFL(Sattler et al. 2020). *Gap* is defined as a means measures the distance between two clusters to verify the clustering is accurate. Gap_1 is the separation gap between two clusters when measured using cosine distance, defined as:

$$Gap_1 := \alpha_{\text{intra}}^{\min} - \alpha_{\text{cross}}^{\max}. \tag{21}$$

When $Gap_1 > 0$ the separation is accurate, that is, the data distribution of clients in the same cluster is similar. Besides, Gap_2 is the separation gap between two clusters when measured using l_2 distance: For the model distance matrix obtained from l_2 distance calculation, gap can be defined as:

$$Gap_2 := \alpha_{\text{cross}}^{\min} - \alpha_{\text{intra}}^{\max}. \tag{22}$$

Because contrary to the cosine distance method, the more similar the data distribution of each clients, the smaller the l_2 distance between them.

Layer-wised clustering

AdaCFL (Gong et al. 2022) proposes to use partial weights instead of whole model weights to compute a model distance matrix that reflects the similarity of any two client models. Experimental results show that the method can significantly reduce the computational cost and outperform other baseline methods.(R1.-3) Specifically, the fully connected (FC) layer weights of the model can be used to better represent the data distribution of different clients. The experimental results of (Chandran et al. 2021) also show that the weight divergence of the model near the FC layer is also significantly higher than that of the convolutional (Conv) layer, which seems to support the conclusion of *AdaCFL*. However, the experimental environment does not consider the influence of different Non-IID environments, which motivates us to explore the generality and validity of its conclusions through more experiments. In the "Experiments" section, we demonstrate the contribution of our LayerCFL algorithm by examining how different Non-IID environments affect different layers of

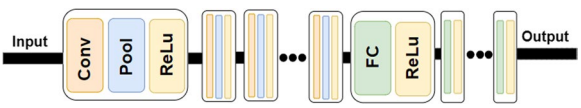


Fig. 2 A simple CNN example

the model in the clustering process. (R1.-3) Fig. 2 illustrates a framework of a simple CNN model.

Algorithm 1:

Input: Initial parameters θ_t, $\varepsilon_1 > 0$, set of clients g, local minibatch size B, local epoches E, learning rate η
Server do:
repeat
  for *each client* $i \in g$ *in parallel* **do**
    $\triangle\theta^i_{t+1} \longleftarrow$ ClientUpdate(θ_g)
  end
  $\theta_{t+1} \longleftarrow \theta_t + \sum_{i=1}^{g} \frac{n_i}{n_g} \triangle \theta^i_{t+1}$
until $\left\| \sum_{i=1}^{g} \frac{n_i}{n_g} \triangle \theta^i_{t+1} \right\| < \varepsilon_1$;
return θ_{t+1}
ClientUpdate(θ):
$\mathcal{B} \longleftarrow$ (split Date into batches of size B)
for *each local epoch* i *From 1 to* E **do**
  for *batch* $b \in \mathcal{B}$ **do**
    $\theta_i \leftarrow \theta - \eta \nabla \zeta(\theta, b)$
  end
end
$\triangle\theta \leftarrow \theta_i - \theta$
return $\triangle\theta$ to server

Algorithm 2:

Input: Initial global parameters θ, set of clients $\mathbf{g}$, $\varepsilon_2 > 0$
$\theta_g \longleftarrow$ FederatedLearning(θ, $\mathbf{g}$)
Get model weight-updates $\triangle\theta \leftarrow \theta_g - \theta$
if *condition is "Feature distribution skew"* **then**
  Select the Conv layer n according to previous statistics
  $\theta^*_g \longleftarrow$ Select n-layer weight-updates of the model $\triangle\theta$
else if *condition is "Label distribution skew"* **then**
  Select the FC layer n according to previous statistics
  $\theta^*_g \longleftarrow$ Select n-layer weight-updates of the model $\triangle\theta$
else
  $\theta^*_g \longleftarrow \triangle\theta$
end
$g_1, g_2 \leftarrow$ Hierarchical clustering based on α calculated by θ^*_g
if $max_{i \in g} \left\| \triangle\theta^*_i \right\| \geq \varepsilon_2$ *and satisfy the Theorem 1 (Correct clustering Theorem)* **then**
  $\theta_{g_i}, i \in g_1 \leftarrow$ LayerCFL(θ_g, g_1)
  $\theta_{g_i}, i \in g_2 \leftarrow$ LayerCFL(θ_g, g_2)
else
  $\theta_{g_i} \leftarrow \theta_g, i \in \mathbf{g}$
end
return $\theta_{g_i}, i \in \mathbf{g}$

The existing literature shows that weights at higher levels of the model are task-related than weights at lower levels of the same model (Yosinski et al. 2014; M. et al. 2018). The Conv layer of this CNN model mainly extracts input data features, while the FC layer plays the role of classifier, that is, it is related to tasks. The above research suggests that we can compute the model distance matrix by selecting the data from a certain layer of the model (such as the FC layer or Conv layer) to better express the model similarity. Some new formulas can be used to express the distance between any two clients (R1.-7):

$$l_2^p : \ dist^*(c_i, c_j) = \| \ \theta_i^* - \theta_j^* \ \|_{l_2} = \sqrt{\sum_{\vartheta \in \theta^*} (\vartheta_i - \vartheta_j)^2}, \tag{23}$$

$$\Delta l_2^p : \ dist^*(c_i, c_j) = \| \ \Delta\theta_i^* - \Delta\theta_j^* \ \|_{l_2}, \tag{24}$$

$$\cos^p : \ dist^*(c_i, c_j) = \frac{\left\langle \theta_i^*, \theta_j^* \right\rangle}{\| \ \theta_i^* \ \| \| \ \theta_j^* \ \|}, \tag{25}$$

$$\Delta \cos^p : \ dist^*(c_i, c_j) = \frac{\left\langle \Delta\theta_i^*, \Delta\theta_j^* \right\rangle}{\| \ \Delta\theta_i^* \ \| \| \ \Delta\theta_j^* \ \|}, \tag{26}$$

where θ^* is the weight corresponds to the selected partial layer of the model. The data for the clustering algorithm of our LayerCFL method in various Non-IID environments are derived from different layer models. We evaluate the performance of LayerCFL in two Non-IID environments.(R1.-3)

In the following section, we present the two Non-IID environments to distinguish the difference between them.(R1.-6)

- **Feature distribution skew** Client i and j have different distribution probabilities $P(x)$ for a given feature x, but different clients have the same probability to predict the probability $P(y|x)$ of label y by using feature x. For example, on EMINIST dataset, client i and j have different image data of the letter T: i has mostly bold letter T images while j has mostly slanted T images, despite storing the data with the same label T. This Non-IID data distribution, known as feature distribution skew, can negatively impact the shared model if they participate in FL at the same time.
- **Label distribution skew** Given any data point (x, y), the label distribution of different clients i and j is different, that is, $P_i(y) \neq P_j(y)$. But the feature is the same when given label y, which means that $P(x|y)$ is the same. For example, 90% of client i's data is digits 7 and 10% is some other number. For client i, the data distribution is similar, but the 90% digits are 5. But given label y, the probability $P(x|y)$ of feature x for any client is the same.

In the "label distribution skew" setting, the FC layer reflects the different data distributions better because it is more task-related. However, this is not hold for other Non-IID settings (e.g., feature distribution skew). In this

setting, using the FC layer to compute the model distance matrix results in a weaker clustering effect. This is because the "feature distribution skew" setting makes the Conv layer capture the data distribution differences among the clients due to the learned features, while the classification layer is relatively insensitive.(R1.-4)

LayerCFL algorithm

Exploring the impact of various environments on the weights of different layers within a model, as well as developing more efficient and accurate methods for calculating the differences between individual client models to cluster clients into appropriate clusters, is a crucial research problem. Addressing this problem is exactly what our proposed LayerCFL aims to achieve. Combined with our current work, the LayerCFL algorithm is proposed in this paper as shown in Algorithm 2.

LayerCFL begins with initializing global model parameter θ and cluster g, and performs FL to make the global shared model of the initial cluster converge to a clustering condition according to Algorithm 1. ε_1 is an empirical hyperparameter used to determine whether FL is close to the stationary solution. The stop condition is defined as:

$$\left\| \sum_{i=1}^{g} \frac{n_i}{n_g} \bigtriangleup \theta_{t+1}^{i} \right\| < \varepsilon_1, \tag{27}$$

and according to the CFL(Sattler et al. 2020) recommendation, ε_1 can be set to around tenth of the maximum average update norm $\varepsilon_1 \approx \max \|\triangle\theta\|/10$. After obtaining the stable solution for FL to reach the clustering start condition, we select the weight-updates of the fixed layer θ^* (Obtain in advance through statistics) of the model uploaded by the client according to the Non-IID environment we are in. And we select the model layers according to the environment: (1) In "Feature distribution skew", we use the data of the Conv layer (2) In "Label distribution skew", we use the data of the FC layer (3) For other cases we use all model parameters by default. It means that the worst performance of our algorithm is not lower than that of the CFL algorithm.

Next we need to check the second condition for clustering, i.e., to verify if the current global model is a stable solution for the clients in cluster. We make the judgment with the help of the hyperparameter ε_1:

$$max_{i \in g} \left\| \theta_i^* \right\| \geq \varepsilon_2, \tag{28}$$

and ε_2 is set in the range $[\varepsilon_1, 10\varepsilon_1]$.

If the clustering conditions are met, we cluster the clients using cosine distance and obtain two clusters. We use *Theorem 1* to validate the clustering and run the LayerCFL algorithm for each cluster. Otherwise, we discard the clustering and revert to the original cluster. The whole algorithm procedure is given in Algorithm 2, and a schematic illustration is presented in Fig. 3 (R1.-2) (R1.-5).

Experiments

In our experience, we evaluated multiple methods on experimental clients with different data distributions and used two classical public datasets. We carried out all of our experiments with PyTorch and collected training data on a server equipped with Intel Xeon Platinum 8255C CPU and NVIDIA TESLA T4 GPU.

Datasets

We chose EMNIST and CIFAR10 as our training datasets.

- EMINIST (Cohen et al. 2017) contains is an extension of MNIST, which divides data into six methods for easy use. We use the Letters dataset, which contains 26 letter types of data and the size of each image is 28×28.
- CIFAR-10 (Krizhevsky and Hinton 2009) contains 10 classes of three-channel images, where the size of each image is 32×32.

In all subsequent experiments, the model we trained was VGG16 (Simonyan and Zisserman 2014), and we used the adam optimizer where we set the batch size to 50 on each client. In order to fit the model to the EMNIST dataset, we transform the image so that the original size of 28×28 becomes 32×32. For the VGG16 model, we changed the initial size of 3 channels to 1 channel so that we could train on the EMNIST dataset. Next, we will introduce the settings related to the Non-IID environment in our experiments.

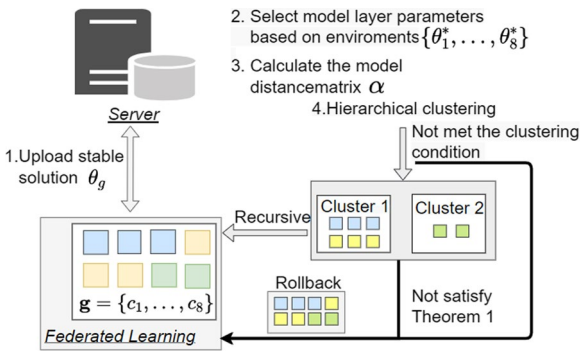


Fig. 3 Schematic overview over the LayerCFL

Non-IID settings

(1) *Feature distribution skew*

Our experimental setup is similar to (Ghosh et al. 2020), where each client has samples from all classes, but some clients rotate their stored image data. We use α to denote the number of clusters we partition the clients into, with a value of 4 indicating that all experiment clients are divided into four groups. Each group of clients performs rotations of 90, 180, 270 degrees, and non-rotation on their stored image data, respectively. To simulate this environment we divided the clients $m = 20$ into $k = 4$ clusters for the two datasets in our next experiment, that is, $\alpha = 4$.

(2) *Label distribution skew*

We use β to represent different levels of Non-IID data like (Wang et al. 2020). When $\beta = 0.8$, 80% of each client's local data belongs to one class, while the remaining 20% belongs to other classes. However, we found that larger β values make it easier to separate clients from different clusters using the model distance matrix, making it difficult to compare different methods when the data volume or number of rounds is low. Therefore, we typically use $\beta = 0.5$ in our experiments, even though it causes the curve to collapse quickly as the number of rounds increases. In our experiment, we divided the clients $m = 20$ into $k = 4$ clusters and 50% of the client's local data in each cluster contained one type of data and the remaining 50% contained other data.

Performance

This section aims to assess the effectiveness of LayerCFL in two Non-IID data settings, by analyzing its clustering speed (in rounds) and data volume requirements. We compare LayerCFL with two baselines, namely *FMTL* and *AdaCFL*.

Performance in feature distribution skew

(1) Performance under different distance calculation formulas for clustering

To assess the sensitivity of different model layers to training data distribution, we use heat maps to visualize the model distance matrix across clients trained on the CIFAR10 dataset. As previously described, we process the image data using the $\alpha = 2$ setting, and divided the $m = 10$ clients into $k = 2$ clusters. Results for various model layers are shown in Fig. 4.

Figure 4 shows that the distance matrix based on model weight of different layers reflects client clusters. Based on Fig. 4, the left half reflects the model distance matrix calculated from part of the Conv layer weights, and can distinguish different clusters more easily than the right half. Our research suggests that in feature distribution skew environments, this feature is more reflected in the Conv

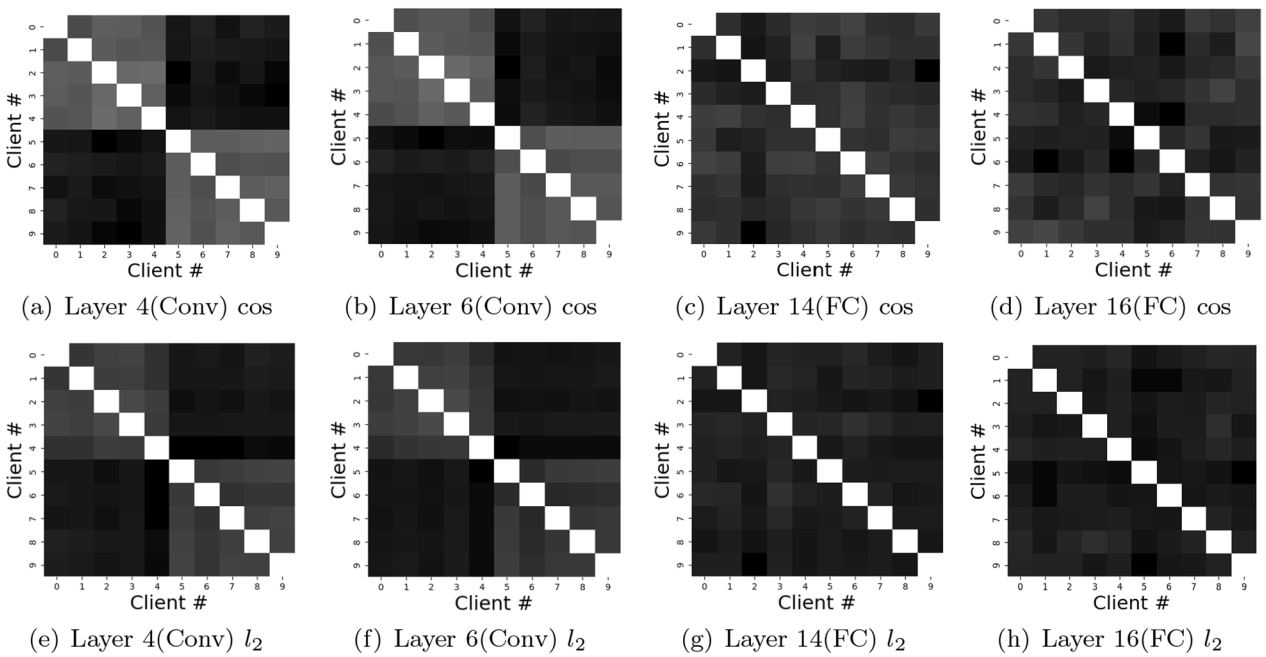


Fig. 4 Visualization of model distance matrix using weight calculations from VGG16 model's different layers. Conv denotes the convolution layer, while FC denotes the fully connected layer. cos and l_2 represent the cosine distance and Euclidean distance, respectively, used for calculating the model distance. Lighter colors indicate greater similarity in data distribution between the two clients

layer than the FC layer, which becomes more pronounced with increasing model complexity. In simple models, the FC layer can also reflect this feature, but it is less pronounced than in the Conv layer. In models like VGG16, it is difficult for the FC layer to distinguish between different clusters.

Figure 4 shows two methods for calculating the model distance matrix based on the weight of the same layer: Euler distance and cosine distance. The upper and lower halves of the figure correspond to these two methods, respectively.

This image shows that the cosine distance method is more effective in distinguishing between different clusters, while the Euler distance method is less effective. We can easily find that the phenomenon in Fig. 4 is obviously different from the conclusion of (Gong et al. 2022). This indicates the need for extending the incomplete conclusion of (Gong et al. 2022) with more experiments in various Non-IID environments to enhance its applicability.

To better demonstrate the sensitivity of each layer of the model to the distribution of client data, we train for 30 rounds under the above experimental conditions until the clustering condition is stable, and then record the *Gap* values of the model distance matrix computed from each layer of the model, as shown in Fig. 5. The x-axis is the layers of the model and the y-axis is the *Gap* value, where the *Gap* value of 0 means that the clustering is incorrect. Based on the definition of *Gap* and Fig. 4, we can infer that the larger the *Gap* value, the easier it is to distinguish between two different clusters of clients in the heatmap. We can also see from Fig. 5 that the *Gap* value for the FC layer is relatively small compared to the Conv layer. Moreover, there are classification errors in the case of the FC layer. In our subsequent experiments, we select different layers to participate in the clustering computation based on the statistical results to obtain the maximum benefit (R1.-9).

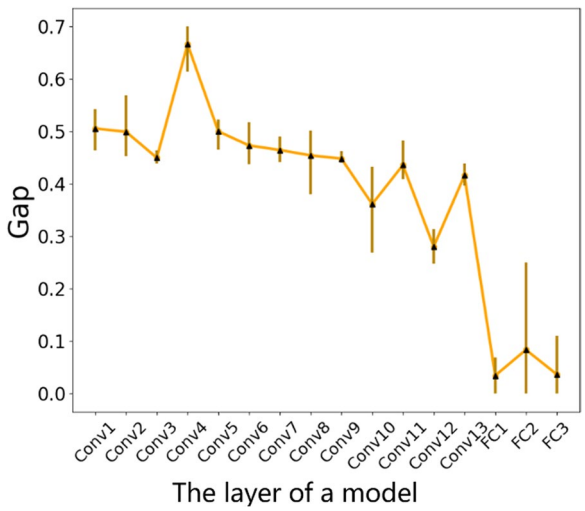


Fig. 5 Visualize the "Gap" values obtained after 30 rounds of training in the "Feature distribution skew" environment and clustering operations based on the parameters of each layer of the model. To facilitate the plotting display, we record the Gap in the category error case as 0 uniformly

(2) Clustering speed comparison under different data volume

We use $\alpha = 4$, and based on both experiential and experimental analysis, we select the fourth layer of the model for calculation. We present a comprehensive analysis of five distinct methodologies for partitioning clusters and conduct experiments on two data sets. Our objective is to compare and contrast the advantages and drawbacks of each method.

To this end, we record the minimum number of communication rounds required for each approach to achieve separation condition ($Gap > 0$) as the size of the data stored on each client increases.

We can estimate the convergence speed of the model under different algorithms by recording the minimum number of training iterations needed for successful clustering of different algorithms. The fewer iterations required, the faster the model converges. The result as shown in Fig. 6. The x-axis represents the data volume, while the y-axis represents the number of rounds of communication. The results were obtained by testing 100 to 1500 data volumes in 200 data intervals.

For EMNIST, the red bar achieves the separation condition with the lowest number of training rounds under low data quantity. As data quantity increases, the red bar maintain the best performance, while the blue and orange bar consistently performs poorly. The yellow and grey bar fail to achieve the separation condition in most cases. For CIFAR10, the red bar performs consistently best under almost all data quantity conditions. Figure 6 shows that the red and blue bar perform well in both datasets, with the red bar being more suitable for most environments. The experiments demonstrate the superiority of cosine distance in distinguishing clients with different data distributions compared to Euler distance. Additionally, incorporating weight-updates is advantageous over solely using model weights. Using partial model layer data has reasonable advantages compared to using all model layer data, guiding the use of partial weight-updates to improve accuracy while reducing computation.

In summary, LayerCFL (Red bar) can achieve the separation condition in different data environments with fewer communication rounds than other separation methods, including baseline methods *FMTL* (Blue bar) and *AdaCFL* (Yellow bar).

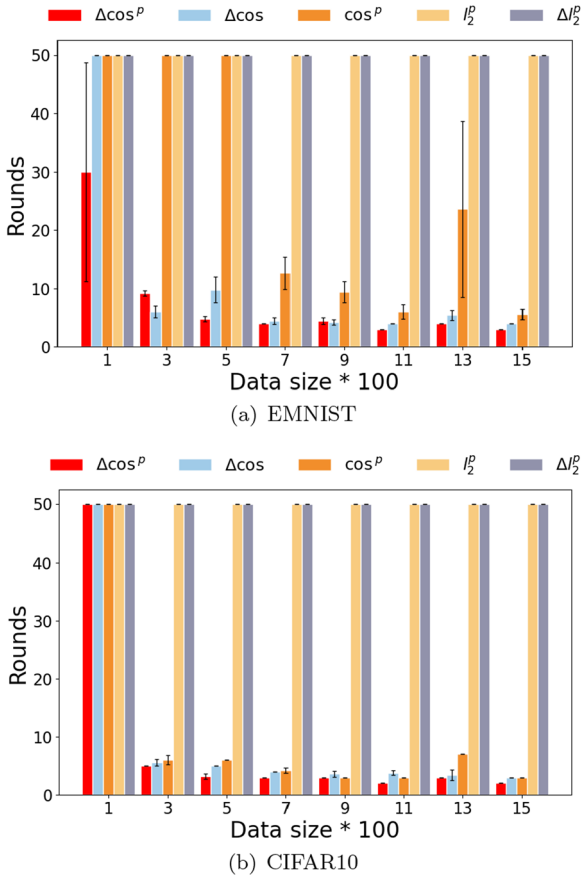


Fig. 6 Visualize the rounds of $Gap > 0$ achieved by each method under different data volume environments. Δ cos, l_2^p, Δl_2^p, cos^p and Δ cos^p stand for the distance defined by Eqs. (6), (23), (24), (25), and (26). Δ cos is FMTL and l_2^p is AdaCFL

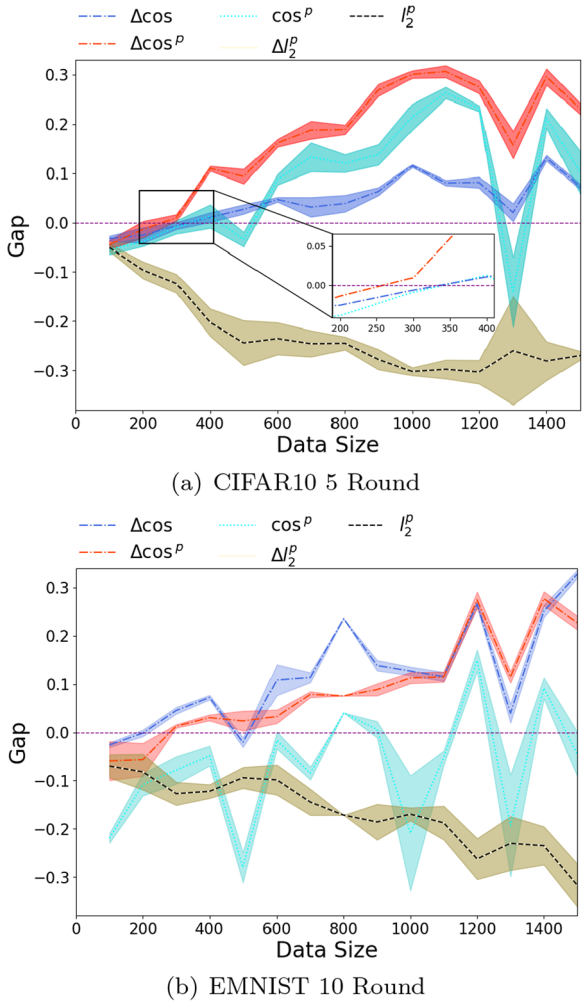


Fig. 7 Visualize the change of *Gap* with the increase of data volume under fixed round. Δ cos, l_2^p, Δl_2^p, cos^p and Δ cos^p stand for the distance defined by Eqs. (6), (23), (24), (25), and (26)

(3) Data volume requirement comparison

To better demonstrate the pros and cons of different methods for calculating the model distance matrix, we collect data and plotted Fig. 7, which shows how the performance changes with increasing data volume under fixed training rounds for each method. Figure 7a and b depict the results for CIFAR10 and EMNIST datasets, respectively, with 100 data intervals and data sizes ranging from 100 to 1500 per client. Figure 7a shows the results for training round 5 with each fixed amount of data in the CIFAR10 dataset, while Fig. 7b shows the results for 10 rounds of training with each fixed amount of data in the EMNIST dataset. The x-axis represents the data volume, while the y-axis represents the *Gap*. When the value of *Gap* exceeds 0, it serves as an indicator that the employed method is capable of effectively segregating dissimilar clusters. Furthermore, methods that can satisfy the segregation criterion while employing minimal data requirements within a predetermined number of communication rounds possess a distinct advantage.

For CIFAR10 dataset, i.e. (a) in Fig. 7, the red line achieved the best performance, achieving separation with less data requirement and higher stability. Although clients in different clusters can be separated when $Gap > 0$, the higher the value of *Gap* is, the higher the clustering accuracy of this method is. For EMNIST dataset, the blue line calculation performed better with low data volumes, but as data increased, the performance of the red line became comparable. This suggests that in higher data volume environments, partial model weight-updates using cosine distance can reduce computational costs while maintaining accuracy.

Based on the above data, we can make some hypotheses. When there is feature distribution skew, the Conv layer may display more information about the data distribution compared to the FC layer. Therefore, to calculate the model distance matrix, choosing to calculate only part of the model layer may improve clustering accuracy and reduce computation. Furthermore, the experimental evidence substantiates that LayerCFL employs the cosine distance of partial model weight-updates, for clustering purposes. This approach confers several benefits over alternative methods in the majority of cases.

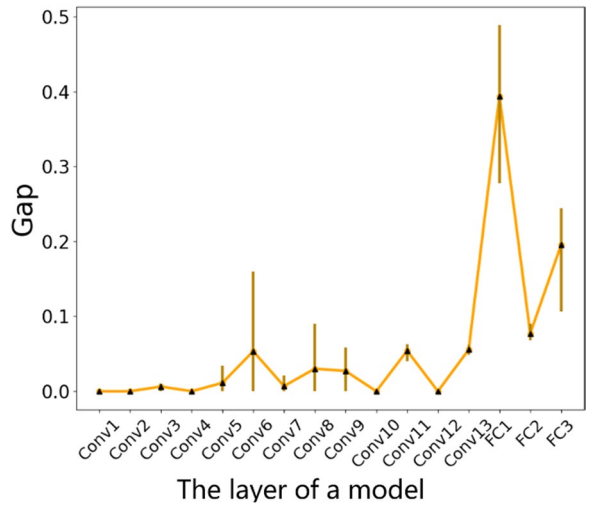


Fig. 8 Visualize the "Gap" values obtained after 30 rounds of training in the "Label distribution skew" environment and clustering operations based on the parameters of each layer of the model. To facilitate the plotting display, we record the Gap in the category error case as 0 uniformly

Performance in label distribution skew

(1) Performance under different distance calculation formulas for clustering

We trained on the EMNIST dataset with $\beta = 0.8$, dividing clients $m = 10$ into $k = 2$ clusters. However, the model distance matrix calculated using different layers showed opposite results to the feature distribution skew. The FC layer better displayed different data distribution characteristics between clients than the Conv layer, consistent with the conclusion of (Gong et al. 2022). See Fig. 9.

Figure 9 clearly shows the differences between clients in different clusters on the right side. The model distance matrix on left side may not completely separate clients of different clusters, some distinguishing characteristics are present, albeit less obvious than those observed in the full-connection layer as shown on the rightside of Fig. 9. Moreover, the cosine distance calculation method yields a clearer model distance matrix compared to the Euler distance calculation method, as evidenced by the comparison of the upper and lower sides of Fig. 9.

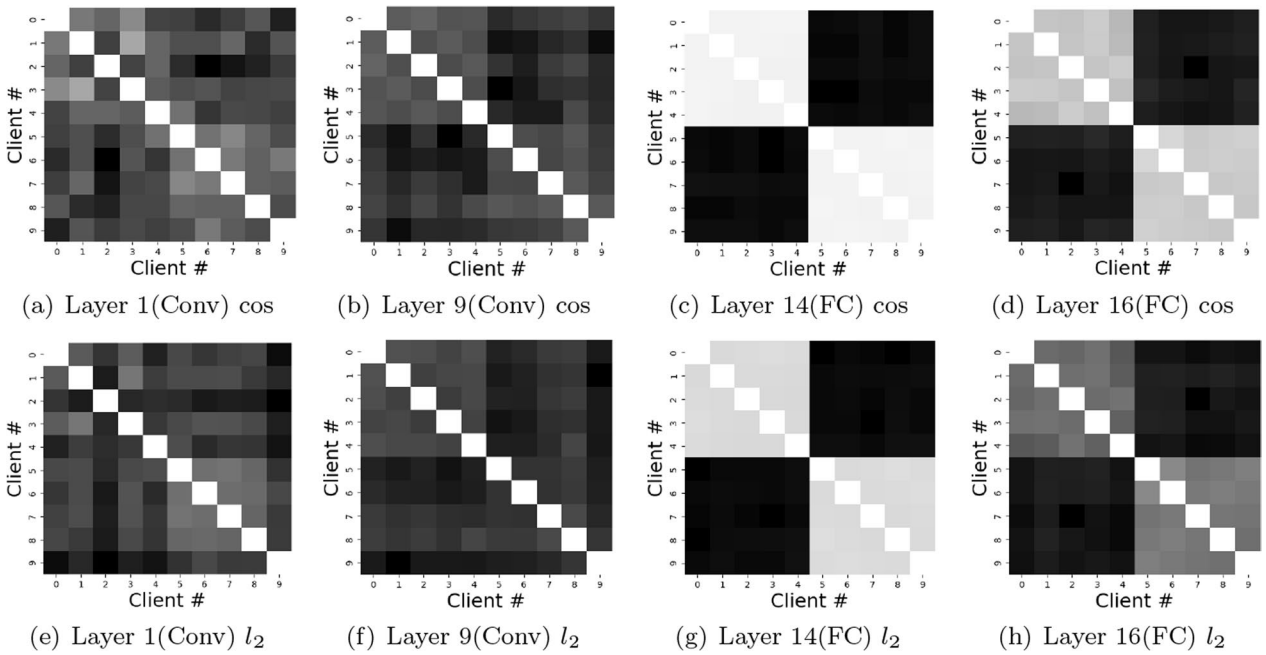


Fig. 9 Visualization of model distance matrix based on weight calculation of different layers of VGG16 model. Conv refers to the convolution layer in the model and FC refers to the fully connected layer. cos means the model distance calculated by the cosine distance and l_2 means the model distance calculated by the Euler distance. The lighter the color in the image, the more similar the data distribution between the two clients

To better demonstrate the sensitivity of each layer of the model to the distribution of client data, We plot the *Gap* values of the distance matrix of the model computed by each layer of the model after 30 rounds of training in the above environment. The results are shown in Fig. 8. We can see from the figure that the data of the Conv layer can also distinguish the clients of different clusters compared to the FC layer but less effectively. This is good evidence that for the "Label distribution skew" environment we should choose the data of the FC layer for clustering computation and have great benefit. Based on the results, we finally select the appropriate model layer to continue the experiment (R1.-9).

(2) Performance under different amounts of data and different rounds

Fig. 10 displays experimental results with $\beta = 0.5$ for each clients storing 1000 images for 50 rounds of training. According to Fig. 9, we select the data of the first FC layer of the model to participate in the calculation. The black, light blue, gold, red, and blue curves correspond to different methods for calculating the model distance matrix. The x-axis represents the data volume, while the y-axis represents the *Gap*. This figure is intended to look at the communication rounds required for each approach to successfully separate the different clusters ($Gap > 0$) with a fixed amount of data stored on each client.

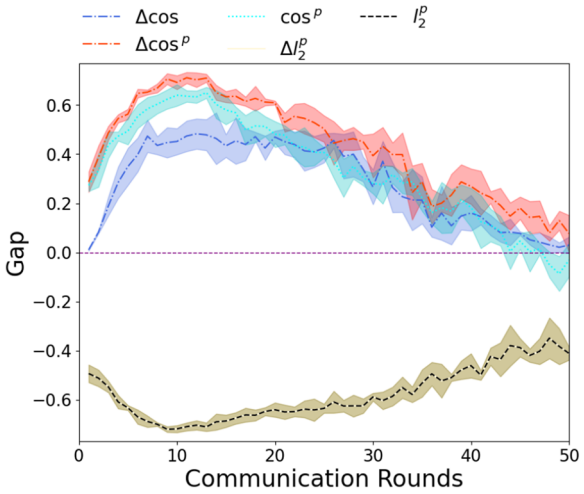
From Fig. 10, the blue, red and light blue curves (Eqs. (6), (26) and (25)) can separate clusters well under $\beta = 0.5$, but their effectiveness weakens with increasing rounds. The red curve is generally better at showing the difference between clusters and the coincidence of the gold and black curves suggests that mathematically the two methods give exactly the same result.

In fact, the blue, red and light blue curves cannot be compared by observing the conditions required to achieve the separation condition ($Gap > 0$). In order to reflect the stability of different algorithms, we count the proportion of successful clusters ($Gap > 0$) in 50 rounds of training for different algorithms under different conditions.

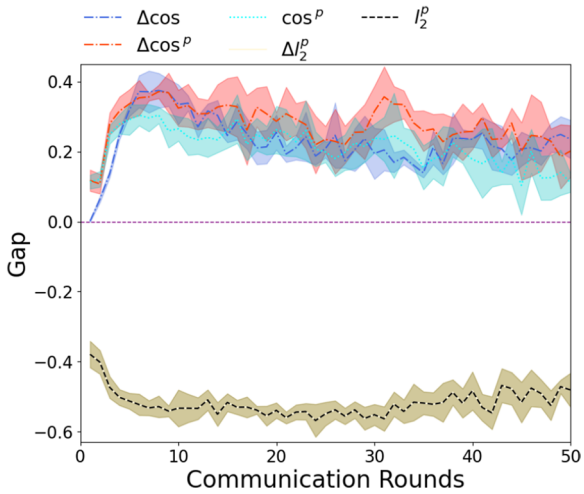
Figure 11 shows experimental results for 100–1500 data sizes stored at 200 data intervals per client and 50 rounds of training at each fixed data volume. Figure 11a shows the results on the EMNIST dataset, and Fig. 11b shows the results on the CIFAR10 dataset. The Eqs. (23) and (24) do not meet the separation condition and are excluded.

From Fig. 11, the blue, orange, and grey bars maintain $Gap > 0$ in most cases within 50 rounds of training under each fixed data volume condition. For EMNIST, the blue bars outperform others in maintaining stability during training, followed by the orange bars.

For CIFAR10, the blue and orange bars show similar performance and maintain $Gap > 0$ in 50 rounds. Therefore, the blue bar is a better performer in various situations. In other words, we can believe that the cosine distance of partial weight-updates (LayerCFL) can be used to approximate the similarity of data distribution between clients in label distribution skew environment.

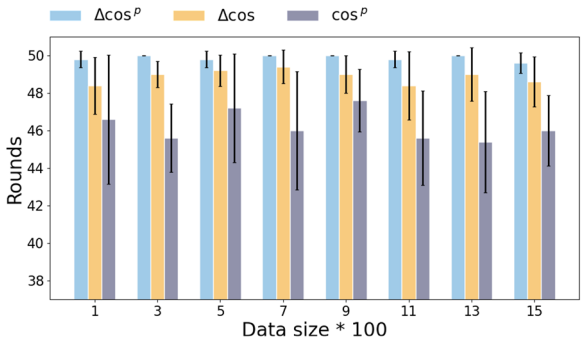


(a) EMNIST 1000 data

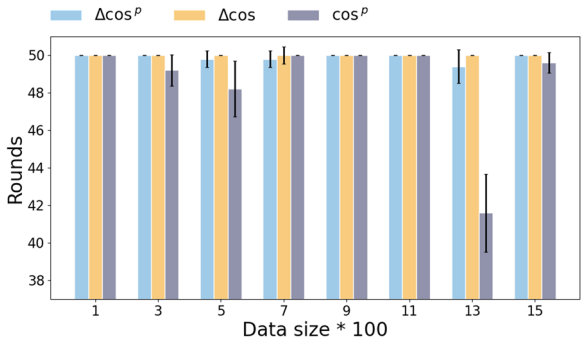


(b) CIFAR10 1000 data

Fig. 10 Visually observe the Gap changes with the increase of training rounds in two datasets with different data amounts. Δ cos, l_2^p, Δl_2^p, cos^p and Δ cos^p stand for the distance defined by Eqs. (6), (23), (24), (25), and (26)



(a) EMNIST



(b) CIFAR10

Fig. 11 Visualize the various methods to maintain the amount of $Gap > 0$ in different data volume environments. Δ cos, cos^p and Δ cos^p stand for the distance defined by Eqs. (6), (25), and (26)

Conclusion

This paper proposes LayerCFL, a layer-wise clustering method to address the problem of inefficient clustering in CFL. LayerCFL is an efficient method that filters partial layers from large-sized models and introduces cosine distance on model weight-updates for clustering. Furthermore, LayerCFL offers various schemes for selecting layers for different non-IID datasets. Cov layers are selected for non-IID datasets with feature distribution skew, while FC layers are selected for datasets with label distribution skew. We evaluate LayerCFL using the EMNIST and CIFAR10 datasets under different data distributions in non-IID settings. Our experiments show that LayerCFL reduces computation resources by an average of 6.47× and communication rounds by an average of 12.89%, making it an effective solution for inefficient clustering compared with the baseline approach *FMTL*.

In our future work, we will expand our investigation to include other types of Non-IID settings, such as different label features, varied labels for the same features, and quantity skew. This broader exploration will provide a more comprehensive understanding of the challenges and implications of Non-IID in machine learning. (R1.-1) Next, we plan to enhance LayerCFL by incorporating meta-learning techniques. This will allow it to adaptively learn filtering and weights for each layer based on the Non-IID nature of client data. Additionally, we aim to deploy LayerCFL to solve practical problems, such as recommendation systems.

Acknowledgements
Supported by the National Natural Science Foundation of China (No. 62002028, No. 62102040 and No. 62202066).

Authors' contributions
All the authors read and approved the final manuscript.

Declaration

Competing interests
All authors declare no conflict of interest.

Received: 29 March 2023 Accepted: 26 June 2023
Published online: 04 December 2023

References

Al-Jarrah OY, Yoo PD, Muhaidat S, Karagiannidis GK, Taha K (2015) Efficient machine learning for big data: a review. Big Data Res 2(3):87–93
Chandran P, Bhat R, Chakravarthi A, Chandar S (2021) Weight divergence driven divide-and-conquer approach for optimal federated learning from non-iid data. arXiv:2106.14503
Cohen G, Afshar S, Tapson J, Van Schaik A (2017) Emnist: Extending mnist to handwritten letters. In: 2017 international joint conference on neural networks (IJCNN), IEEE, pp 2921–2926
Criado MF, Casado FE, Iglesias R, Regueiro CV, Barro S (2022) Non-iid data and continual learning processes in federated learning: a long road ahead. Inform Fusion 88:263–280. https://doi.org/10.1016/j.inffus.2022.07.024
Dennis DK, Li T, Smith V (2021) Heterogeneity for the win: One-shot federated clustering. In: International conference on machine learning, PMLR, pp 2611–2620
Gao C, Wang X, He X, Li Y (2022) Graph neural networks for recommender system. In: WSDM '22: The fifteenth ACM international conference on Web search and data mining
Ghosh A, Chung J, Yin D, Ramchandran K (2020) An efficient framework for clustered federated learning. Adv Neural Inf Process Syst 33:19,586-19,597
Gong B, Xing T, Liu Z, Wang J, Liu X (2022) Adaptive clustered federated learning for heterogeneous data in edge computing. Mobile Networks Appl 27(4):1520–1530. https://doi.org/10.1007/s11036-022-01978-8
Kairouz P, McMahan HB, Avent B, Bellet A, Bennis M, Bhagoji AN, Bonawitz K, Charles Z, Cormode G, Cummings R et al (2021) Advances and open problems in federated learning. Foundations and Trends® in Machine Learning 14(1–2):1–210
Kemp S (2022) Digital 2022: Global overview report, datareportal, 2022. https://datareportal.com/reports/digital-2022-global-overview-report
Kim H, Kim Y, Park H (2021) Reducing model cost based on the weights of each layer for federated learning clustering. In: 2021 Twelfth International Conference on Ubiquitous and Future Networks (ICUFN), IEEE, pp 405–408
Krizhevsky A, Hinton G (2009) Learning multiple layers of features from tiny images. University of Toronto, Tech Rep, Computer Science Department, p 1
Li Q, Diao Y, Chen Q, He B (2022) Federated learning on non-iid data silos: An experimental study. In: 2022 IEEE 38th International Conference on Data Engineering (ICDE), pp 965–978, https://doi.org/10.1109/ICDE53745.2022.00077
Li T, Sahu AK, Talwalkar A, Smith V (2020) Federated learning: challenges, methods, and future directions. IEEE Signal Process Mag 37(3):50–60

Li T, Sahu AK, Zaheer M, Sanjabi M, Talwalkar A, Smith V (2020) Federated optimization in heterogeneous networks. Proc Mach Learn Syst 2:429–450
Li X, Huang K, Yang W, Wang S, Zhang Z (2019) On the convergence of fedavg on non-iid data. arXiv:1907.02189
Liu B, Ding M, Shaham S, Rahayu W, Farokhi F, Lin Z (2021) When machine learning meets privacy: a survey and outlook. ACM Comput Surv (CSUR) 54(2):1–36
Liu F, Wu X, Ge S, Fan W, Zou Y (2020) Federated learning for vision-and-language grounding problems. Proc AAAI Conf Artif Intell 34:11572–11579
M L, Y C, Z C (2018) Transferable representation learning with deep adaptation networks. IEEE Trans Pattern Anal Mach Intell
Ma X, Zhang J, Guo S, Xu W (2022a) Layer-wised model aggregation for personalized federated learning
Ma X, Zhu J, Lin Z, Chen S, Qin Y (2022) A state-of-the-art survey on solving non-iid data in federated learning. FGCS, Future generations computer systems
Mansour Y, Mohri M, Ro J, Suresh AT (2020) Three approaches for personalization with applications to federated learning. Computer Science
McMahan B, Moore E, Ramage D, Hampson S, y Arcas BA (2017) Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics, PMLR, pp 1273–1282
McMahan HB, Moore E, Ramage D, y Arcas BA (2016) Federated learning of deep networks using model averaging. arXiv preprint arXiv:1602.05629 2
Mothukuri V, Parizi RM, Pouriyeh S, Huang Y, Dehghantanha A, Srivastava G (2021) A survey on security and privacy of federated learning. Futur Gener Comput Syst 115:619–640
OpenAI (2022) Chatgpt: optimizing language models for dialogue. https://openai.com/blog/chatgpt/
Ouyang X, Xie Z, Zhou J, Huang J, Xing G (2021) Clusterfl: a similarity-aware federated learning system for human activity recognition. In: MobiSys '21: The 19th Annual International Conference on Mobile Systems, Applications, and Services
Papernot N, McDaniel P, Sinha A, Wellman M (2016) Towards the science of security and privacy in machine learning. arXiv preprint arXiv:1611.03814
Sattler F, Muller KR, Samek W (2020) Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. IEEE Transactions on Neural Networks and Learning Systems PP(99):1–13
Silva S, Gutman BA, Romero E, Thompson PM, Altmann A, Lorenzi M (2019) Federated learning in distributed medical databases: Meta-analysis of large-scale subcortical brain data. In: 2019 IEEE 16th international symposium on biomedical imaging (ISBI 2019), IEEE, pp 270–274
Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. Computer Science
Voigt P, Von dem Bussche A (2017) The eu general data protection regulation (gdpr). A Practical Guide, 1st Ed, Cham: Springer International Publishing 10(3152676):10–5555
Wang H, Kaplan Z, Niu D, Li B (2020) Optimizing federated learning on non-iid data with reinforcement learning. In: IEEE INFOCOM 2020 - IEEE Conference on Computer Communications
Wang X, Han Y, Wang C, Zhao Q, Chen M (2019) In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning. IEEE Network PP(99):1–10
Wu C, Wu F, Cao Y, Huang Y, Xie X (2021) Fedgnn: Federated graph neural network for privacy-preserving recommendation. arXiv preprint arXiv:2102.04925
Yang L, Huang J, Lin W, Cao J (2022) Personalized federated learning on non-iid data via group-based meta-learning. ACM Trans Knowl Discov Data just Accepted. https://doi.org/10.1145/3558005
Yang Q, Liu Y, Chen T, Tong Y (2019) Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology 10(2):1–19
Yang W, Zhang Y, Ye K, Li L, Xu CZ (2019b) Ffd: A federated learning based method for credit card fraud detection. In: Big Data–BigData 2019: 8th International Congress, Held as Part of the Services Conference Federation, SCF 2019, San Diego, CA, USA, June 25–30, 2019, Proceedings 8, Springer, pp 18–32
Yosinski J, Clune J, Bengio Y, Lipson H (2014) How transferable are features in deep neural networks? MIT Press, Cambridge
Zeiler M, Fergus R (2014) Visualizing and understanding convolutional networks. In: ECCV 2014
Zhao Y, Li M, Lai L, Suda N, Civin D, Chandra V (2018) Federated learning with non-iid data. arXiv preprint arXiv:1806.00582
Zhou L, Pan S, Wang J, Vasilakos AV (2017) Machine learning on big data: Opportunities and challenges. Neurocomputing 237:350–361
Zhu H, Xu J, Liu S, Jin Y (2021) Federated learning on non-iid data: A survey. Neurocomputing 465:371–390 https://doi.org/10.1016/j.neucom.2021.07.098, www.sciencedirect.com/science/article/pii/S0925231221013254

Publisher's Note