Towards Effective Clustered Federated Learning: A Peer-to-Peer Framework With Adaptive Neighbor Matching

Zexi Li, Jiaxun Lu, Member, IEEE, Shuang Luo, Didi Zhu, Yunfeng Shao, Member, IEEE, Yinchuan Li, Member, IEEE, Zhimeng Zhang, Yongheng Wang, and Chao Wu

Abstract—In federated learning (FL), clients may have diverse objectives, and merging all clients' knowledge into one global model will cause negative transfer to local performance. Thus, clustered FL is proposed to group similar clients into clusters and maintain several global models. In the literature, centralized clustered FL algorithms require the assumption of the number of clusters and hence are not effective enough to explore the latent relationships among clients. In this article, without assuming the number of clusters, we propose a peer-to-peer (P2P) FL algorithm named FARM. In FARM, clients communicate with peers to adaptively form an effective clustered topology. Specifically, we present two novel metrics for measuring client similarity and a two-stage neighbor matching algorithm based Monte Carlo method and Expectation Maximization under the Gaussian Mixture Model assumption. We have conducted theoretical analyses of FARM on the probability of neighbor estimation and the error gap to the clustered optimum. We have also implemented extensive experiments under both synthetic and real-world clustered heterogeneity. Theoretical analysis and empirical experiments show that the proposed algorithm is superior to the P2P FL counterparts, and it achieves better performance than the centralized cluster FL method. FARM is effective even under extremely low communication budgets.

Index Terms—Clustered federated learning, distributed learning, Federated learning, peer-to-peer communication

1 INTRODUCTION

The proliferation of smart devices such as mobile phones, cameras, and sensors has dramatically expanded the perception of edge intelligence, increasingly forming an Internet of Things (IoT) network [1], [2], [3]. The massive data from edge devices is key in generating powerful predictive models to provide better services to users. However,

- Zexi Li and Didi Zhu are with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China. E-mail: {zexi.li, didi_zhu}@zju.edu.cn.
- Jiaxun Lu, Yunfeng Shao, and Yinchuan Li are with Huawei Noah’s Ark Lab, Beijing 100085, China. E-mail: {ljiaxun, shaoyunfeng, yuinchuan}@huawei.com.
- Shuang Luo and Chao Wu are with the School of Public Affairs, Zhejiang University, Hangzhou 310027, China. E-mail: {luoshuang, chao.wu}@zju.edu.cn.
- Zhimeng Zhang is with the School of Software Technology, Zhejiang University, Hangzhou 310027, China. E-mail: zhimeng@zju.edu.cn.
- Yongheng Wang is with Big Data Intelligence Research Center, Zhejiang Lab, Hangzhou 311121, China. E-mail: wangyh@zjilab.com.

This work was supported in part by the National Key Research and Development Project of China under Grant 2021ZD0110400, in part by the National Natural Science Foundation of China under Grant 61901042, in part by the University Synergy Innovation Program of Anhui Province under Grant GXXT-2021-004, and in part by the Key Research Project of Zhejiang Lab under Grant 2021KE0AC02.

(Re)This work was supported in part by the National Key Research and Development Project of China under Grant 2021ZD0110400, in part by the National Natural Science Foundation of China under Grant 61901042, in part by the University Synergy Innovation Program of Anhui Province under Grant GXXT-2021-004, and in part by the Key Research Project of Zhejiang Lab under Grant 2021KE0AC02.

This article has supplementary downloadable material available at https://doi.org/10.1109/TBDATA.2022.322971, provided by the authors.

Manuscript received 31 May 2022; revised 25 October 2022; accepted 8 November 2022. Date of publication 17 November 2022; date of current version 13 November 2024.

In the real-world practice of FL, heterogeneity is an inherent problem (i.e., the Non-IID problem) since clients may have heterogeneous data distributions and thus diverse optimization objectives (learning tasks) [8]. Clustered heterogeneity is prevalent in users’ data, which means that a small group of clients has similar data distributions while there is dominant inconsistency among different groups. It is very common in applications such as recommendation systems [9], [10]. Therefore, clustered FL methods [11], [12], [13], [14] are proposed for better personalization by grouping clients into clusters and maintaining a global model in each cluster. The main challenge of clustered heterogeneity is that the latent similarity relationship among clients is unknown. Existing clustered FL researches adopt the conventional server-client communication pattern and estimate clients’ cluster identities by iterative [11] or hierarchical [13] methods. These centralized clustered FL methods highly rely on the assumed number of clusters or the assumed hierarchical level. However, in the real-world environment, the clustered relationship is latent, and it is impossible to know the number of clusters as prior knowledge. Inappropriate estimations of clusters’ numbers will cause bad convergence. Besides,
other concerns, like the reliability and communication bandwidth issues [15], brought by the central server, also hinder the performance of these centralized clustered FL methods.

To address these challenges, in this paper, we transform the client clustering problem into a binary classification problem from a peer-to-peer (P2P) perspective. Under P2P communication, each client decides whether to accept an accessible client as its neighbor based on local similarity measurement. Clients will select the most similar peers as their neighbors to realize personalization. Once the neighbor estimation is established, a clustered communication topology will be inherently built without assuming the number of clusters. Our method is named as Personalized Adaptive Neighbour Matching (PANM) and it is proved to be effective and robust in various clustered heterogeneity.

In addition to the advantages compared with centralized clustered FL, PANM has superior performance over P2P FL counterparts. In previous works, most P2P FL algorithms assume random or fixed communication topologies [16], [17] and they focus on reaching the global consensus by optimization techniques [18], [19]. But in clustered heterogeneity, the global consensus does not exist, and random or fixed communications will harm personalization and decrease the accuracy. On the contrary, our method achieves partial group consensus by realizing the adaptive topology.

Our main contributions are as follows.

- We propose two efficient, effective, and privacy-preserving metrics to evaluate the pair-wise similarity of client objectives in P2P FL. They are based on losses and gradients, respectively.
- We present PANM, a novel clustered FL algorithm based on P2P communication. PANM enables clients to match neighbors with consistent objectives (same cluster identity), improving local performance.
- We devise two stages in PANM, the first is neighbor selection based on Monte Carlo, and the second is neighbor augmentation based on Gaussian Mixture Model. We provide theoretical guarantees of PANM.
- We conduct extensive experiments on different datasets, Non-IID degrees, and network settings under both synthetic and real-world clustered heterogeneity. It is shown that PANM outperforms all P2P baselines, including Oracle that has the prior knowledge of cluster identities. Compared with centralized clustered FL algorithms, PANM is more effective in exploring latent cluster structure and has better performance.

The rest of this paper is organized as follows. Section 2 reviews related works in clustered FL and P2P FL. Section 3 provides basic formulation about P2P clustered FL. Section 4 elaborates the technical details of PANM, including the two metrics for measuring client similarity and the two-stage neighbor matching algorithm. Theoretical analysis is also included in Section 4. Section 5 presents the experimental results. Sections 6 and 7 provide further discussions and the conclusion of this paper.

2 RELATED WORKS

Clustered Federated Learning. Clustered FL holds the Non-IID assumption that different groups of clients have their own optimization objectives, and it is usually used to realize better accuracy performance [11], [13] or better compression of model updates [20], [21]. In this paper, we focus on the effectiveness of clustered FL on test accuracy, especially local personalization. In personalized clustered FL [11], [13], aggregating models in the same cluster will bring better personalization while aggregation among different clusters will cause negative transfer.1

To group clients into clusters, the main challenge of clustered FL is measuring client similarity. There are mainly three types of measurement, based on losses [11], gradients [13], [14], [23], model weights [12], [13], [22], [24], respectively. In loss-based measurement, clients receive several models and infer them on the local dataset, and the one with the smallest loss has maximal similarity [11]. For model weight and gradient measurement, cosine distance [12], [13] or euclidean distance [12], [22], [24] are used in previous works. It is verified that clients with similar data distributions will have small euclidean distances and large cosine similarities in gradients or model weights.

The methods for clustering in previous works can be divided into two streams, the K-means-based and the hierarchical. For the K-means-based approaches, FedSEM [12] first implements the K-means method based on clients’ euclidean distances of model weights on the server to cluster clients. However, server-side K-means clustering is computationally expensive. To solve this issue, Duan et al. [14], [23] use decomposed cosine similarity to speed up computation and design an efficient newcomer device cold start mechanism. Additionally, Ghosh et al. [11] propose an efficient algorithm IFCA by inherently applying K-means to the client side. IFCA keeps several global models, and clients iteratively choose which global model it is prone to contribute to based on local losses of global models. In another stream of works, hierarchical clustering methods are used to achieve better personalization. Sattler et al. [13] use a hierarchical optimal bi-partitioning algorithm based on cosine similarity of weights or gradients. By bi-partitioning, the method realizes a model tree from personalization to generalization. Further, Briggs et al. [24] design a hierarchical algorithm for a wider range of Non-IID settings, and the method separates clusters of clients by the similarity of their local models to the global model. Additionally, Dem-AI is developed for building large-scale distributed and democratized machine learning systems [22], and it realizes a bottom-up hierarchical clustering with specialized–generalized duality.

Note that all the algorithms mentioned above in clustered FL rely on the assumption of the number of clusters (the K-means-based methods) or the level of hierarchy (the hierarchical techniques). However, the number of clusters is latent and cannot be obtained as prior knowledge, and if the hyperparameters are set inappropriately, the clustering performance will degrade.

1. In some works, it is found that generalized knowledge can be transferred among cluster centroids [14], [22]. However, this only happens when the clustered heterogeneity is not dominant. In this paper, we focus on more heterogeneous clustered FL, where negative transfer exists among different clusters.
Peer-to-Peer Federated Learning. Peer-to-peer federated learning (P2P FL, also known as decentralized FL) alters the centralized topology of conventional FL, and it allows clients to communicate with limited neighbors [16], [25]. There is a study comparing decentralized algorithms like gossip learning with centralized FL in terms of communication efficiency. It is found that the best gossip variants perform comparably to the best centralized FL algorithms overall [26]. Early works related to P2P FL introduce the P2P FL problem under privacy constraints and provide theoretical guarantees. Lalitha et al. [16], [17] use a Bayesian-like approach to let clients collectively learn a model that best fits the observations over the entire network. Bellet et al. [27] make P2P FL differentially private and analyze the trade-off between utility and privacy. They mainly study P2P FL under the IID data assumption, but heterogeneity is prevalent in FL practices.

Recent works mostly discuss class imbalance heterogeneity and communication problems. First, to tackle class imbalance, Li et al. [28] use mutual knowledge distillation instead of weight averaging. Bellet et al. [29] elaborate to design a topology from holistic perspective. However, without a central server, the holistic perspective is impractical, and it is hard for clients to form such topology with limited observations. Second, communication of P2P FL can be more efficient by sparsification [30], adaptive partial gradient aggregation [31], and using max-plus linear system theory to compute throughput [32]. Most recently, swarm learning [25] has been brought up as a P2P FL customized for medical research, utilizing edge computing and blockchain as infrastructures, and it has attracted wide attention. It provides strong application practices of P2P FL.

While we are formulating this paper, we find a related same-time work (PENS) that has the same motivation as ours but uses different methods [33]. PENS adopts a two-stage strategy. In the first stage, clients select top $k$ peers as neighbors for aggregation from randomly sampled $l$ neighbor candidates in each round. After the first stage, clients select the peers that were selected from as neighbors more than "the expected amount of times" in the first stage as permanent neighbors. In the second stage, in each round, clients randomly choose $k$ neighbors for aggregation from permanent neighbors. It is possible for PENS to have noisy neighbor estimations, and we analyze the superiority of PANM to PENS in Sections 4 and 5.

3 Problem Formulation

We first set up the clustered heterogeneity following previous works [11]. There are $r$ different data distributions (clusters), $D^1, \ldots, D^r$, and that the $n$ clients are partitioned into $r$ disjoint clusters. It is assumed that every client $i$ ($i \in [n]$) which belongs to cluster $j$ ($j \in [r]$) contains IID data samples $D_i$ drawn from $D^j$. For simplicity, we assume every client has the same number of samples that $\forall i, j \in [n], |D_i| = |D^j| = d$.

We solve clustered FL by forming it into a personalized P2P FL problem, in which we learn the personalized models $w = (w_1, \ldots, w_n)$ and the neighbor graph matrix $G$. The expression is borrowed from personalized decentralized joint learning [17], and the key difference is that we set binary elements instead of continuous elements in $G$.

because in clustered heterogeneity, the task is to form a neighbor graph where the same-cluster clients should be connected (set as 1) while the different-cluster should be disconnected (set as 0). The neighbor graph matrix $G$ is an $n \times n$ square matrix, $G_{ij}$ refers to the $(i, j)$th entry of the matrix, and it indicates whether client $j$ is in the neighbor bag of client $i$, 1 for true and 0 for false. The diagonal elements $G_{ii}$ are all set to 1. To optimize $w$ and $G$, the joint optimization objective is

$$\min_{w \in [d]^n, G \in \{0, 1\}^{n \times n}} J(w, G) = \sum_{i=1}^{n} F_i(w_i) + \frac{\nu}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} G_{ij} \|w_i - w_j\|^2,$$

$$s.t. \quad g(G) = 0.$$  \hspace{1cm} (1)

There are two terms in this objective function. The first one is the sum of loss functions, each involving the personalized model and the local dataset. $F_i : \mathbb{R}^d \rightarrow \mathbb{R}$ is the loss function of client $i$ ($i \in [n]$) on its local dataset, given by: $F_i(w_i) := \mathbb{E}_{\xi \sim D_i}[f(w_i, \xi)]$, where $\xi \sim D_i$ denotes a random sample drawn from the local dataset, $\mathbb{E}()$ means expectation, and $f(w_i, \xi)$ refers to the loss function of $w_i$ on a sample $\xi$. The second term enables collaboration by encouraging two clients to have a similar model if they are in neighborhood relationship. The condition $g(G) = 0$ regularizes the graph matrix regarding the network topology. For instance, the matrix should have diagonal vector to be 1 and the Frobenius norm $\|G\|_F$ should be assigned to meet the network setup. Other functions that regularize the degree of each node, both in-degree and out-degree, can also be integrated in $g(G)$. In this paper, we solve the objective in Equation (1) by adaptively matching neighbors.

We then introduce the model update protocol in P2P FL. In P2P FL, each client $i$ first updates its local model by local training and receives models from the neighbors. Then it averages the neighbors’ models together with its local model into a new one and starts a next-round training. The process of one communication round in P2P FL can be formulated into the following:

$$w_i^{t+1} = w_i^t - \eta \nabla F_i(w_i^t) + \sum_{j \in N_i^t} (w_j^t - \eta \nabla F_j(w_j^t)).$$ \hspace{1cm} (2)

$N_i^t$ is the round $t$'s aggregation neighbors of client $i$, randomly sampled from the neighbor bag $B_i^t$. $(N_i^t \subseteq B_i^t, |N_i^t| = k, |B_i^t| = m, k \leq m)$. The neighbor bag of client $i$ is defined as the set of indexes $j$ where $G_{ij} = 1, i \neq j$, as $B_i^t = \{j : G_{ij} = 1 : j \in [n], j \neq i\}$. The process that aggregating with random sampled peers from the neighbor bag is known as random gossip communication [26], [33]. In previous P2P FL works, the neighbor bag for each client includes all the remaining peers, but in clustered heterogeneity, such communication is noisy. In PANM, we keep the neighbor bag small but pure in the first stage and augment the neighbor bag in the second stage.

To improve readability, we summarize our main notations mentioned before or soon later as in Table 1.
4 Method

In this section, we will present two metrics for client similarity measurement in Section 4.1, neighbor selection based on Monte Carlo (first stage of PANM) in Section 4.2, and neighbor augmentation based on EM-GMM in Section 4.3 (second stage of PANM). Then we will combine the similarity metrics, neighbor selection, and neighbor augmentation to devise PANM in Section 4.4. Lastly, we will provide theoretical analysis in Section 4.5.

4.1 Metrics for Measuring Client Similarity

Metrics for measuring the consistency of optimization objectives are needed to enable clients to select same-cluster peers and filter out outliers. However, due to privacy concerns, in FL, we cannot use data distance measurements like maximum mean discrepancy distance [34], since sharing data is forbidden. Loss evaluation is a simple metric, commonly used in the literature [11], [33]. In P2P FL, client ’s model and infers the model on its local dataset. If the loss is small, it means client j’s model is more similar to the central model. In a denser network, the clients’ models are more synchronized, and it is more similar to the centralized FL, therefore, will be beneficial when the P2P network is more densely connected. In a denser network, the clients’ models are more synchronized, and it is more similar to the centralized FL, therefore, will be more effective. Conversely, when the network is sparse, smaller α will help.

Notably, the combination of cosines θ1 and θ2 is more robust and effective compared with using one cosine function alone. We implement an ablation experiment as illustrated in Fig. 2. It is obvious that PANMGrad surpasses PANMGrad-theta1 and PANMGrad-theta2 by a large margin in accuracy curves. Besides, the curve of PANMGrad is stable and robust in both training stages (we will introduce the stages in Sections 4.2 and 4.3) while there are disturbances in the baseline curves. We explain the robustness and effectiveness of our metric by the complementarity of \( \cos \theta_1 \) and \( \cos \theta_2 \). \( \cos \theta_2 \) indicates the gradient direction of current round while \( \cos \theta_2 \) reflects historical neighbor-ship and optimization direction in previous rounds.

Moreover, we found the newly proposed gradient-based metric is computation-efficient compared with the loss-based metric. Without loss of generality, to simplify the In Equation (4), \( g_i = w_i^t - w_i^{t-1}, g_j = w_j^t - w_j^{t-1} \). \( g_i \) is the vectorized gradient of client i in round t. \( g_i \) and \( g_j \) have e dimensions, and e is usually large in neural networks. \( \cos \theta_{ij} \) is the cosine function of the two gradient vectors, where \( \theta_{ij} \) refers to the angle of two vectors in the high-dimensional space. In centralized FL, models are initialized as the same global model at the beginning of local training in each round that \( w_i^{t-1} = w_j^{t-1} = w^{t-1} \), so the cosine function of gradients can effectively imply the consistency. We draw a 2-dimensional toy example to intuitively show the optimization trajectories and the angles of vectorized updates in Fig. 1. The trajectory of centralized FL is shown in (a) of Fig. 1. Although in P2P FL, client model weights diverge since the first round, the measurement of gradients will be noisy. The angle of gradients in P2P FL is shown as \( \theta_{ij} \) in (b) of Fig. 1.

To solve this issue, we notice the accumulated weight updates from the initial model can signify the history optimization directions, and the cosine similarity of the weight updates can also imply the consistency of objectives as

\[
\cos \theta_{ij}^2 = \frac{(h_i', h_j')}{\|h_i'\| \cdot \|h_j'\|} 
\]

where \( h_i' = w_i^t - w_0, h_j' = w_j^t - w_0 \). \( h_i' \) is the vectorized accumulated weight updates of client i from the initial model to the model in round t. Analogical to \( \theta_{ij} \), \( \theta_{ij}^2 \) is the angle of two vectorized accumulated updates in the e-dimensional space, and we show \( \theta^2 \) in (b) of Fig. 1.

According to Equations (4) and (5), we combine the cosine functions of \( \theta_1 \) and \( \theta_2 \) to formulate our new metric as

\[
s_{ij} = \alpha \cos \theta_{ij} + (1 - \alpha) \cos \theta_{ij}^2 
\]

where \( \alpha \) is the hyperparameter controlling the weight of two cosine functions, \( \alpha \in [0, 1] \), we adopt the metric in Equation (6) in PANM, notated as PANMLoss. We note that \( \cos \theta_{ij} \) has the same range as \( \cos \theta_{ij}^2 \), which is \([-1, 1]\), so the outputs of two functions will have similar volumes. Thus, it is appropriate to set \( \alpha \) around 0.5. Additionally, we notice that larger \( \alpha \) will be beneficial when the P2P network is more densely connected. In a denser network, the clients’ models are more synchronized, and it is more similar to the centralized FL, therefore, \( \cos \theta_{ij} \) will be more effective. Conversely, when the network is sparse, smaller \( \alpha \) will help.

\[
\begin{array}{c|c|c}
\text{Notation} & \text{Meaning} \\
\hline
n & \text{Number of all clients} \\
r & \text{Number of clusters} \\
a & \text{Number of clients within a cluster} \\
d & \text{Number of data samples in a client} \\
c & \text{Assumed number of clusters in TFCA [11]} \\
k & \text{Size of aggregation neighbor list} \\
l & \text{Size of neighbor candidate list} \\
\tau & \text{Round interval of EM in the second stage} \\
\alpha & \text{Hyperparameter in the gradient-based metric} \\
N_t^i & \text{Neighbor list of client i in round t} \\
B_t^i & \text{Neighbor bag of client i in round t} \\
C_t^i & \text{Neighbor candidate list of client i in round t} \\
S_t^i & \text{Selected neighbors in EM-step} \\
M_t^i & \text{Union set of \( C_t^i \) and \( S_t^i \)} \\
H_t^i & \text{Neighbor estimation list in EM-step of client i} \\
\end{array}
\]

In Table 1, Important Notations in This Paper, we list the notation and meaning of the symbols used in this paper. 

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
analysis, we suppose the model is a $1 \times e$ vector, and a data sample is an $e \times 1$ vector, thus the computation of the inner product of gradients equals the computation of one data sample’s inference. For client $i$, the number of local samples is $d$ and the number of neighbor candidates is $l$. To calculate the candidates’ similarity, the loss-based metric has $O(dl)$ computation complexity while the gradient-based metric has $O(l)$ complexity. Usually, we have $d \gg l$, therefore the gradient-based metric is more efficient in computation. Although the loss-based metric can be more computation-efficient by reducing the dataset size, it will also reduce the effectiveness of measurement.

4.2 Neighbor Selection Based on Monte Carlo
Based on the similarity metrics mentioned in the last subsection, we can devise our P2P FL algorithm PANM. We introduce the first stage of PANM in this subsection.

In P2P FL, clients have access to receive models from randomly sampled peers ($C_i^t, |C_i^t| = 0$), and they need to select neighbors for model aggregation from these candidates. A natural way is to choose the top $k$ candidates with maximal similarities in each round, which is adopted in PENS [33]. However, this method has constant expectation on the probability of correct neighbor estimation during the training process (we will theoretically prove it in Theorem 1 and Corollary 1). If $l, k$ are not appropriately set or the environment is more heterogeneous, the neighbor estimation will be constantly noisy. We solve this challenge by resorting to Monte Carlo method. We formulate the objective of the first stage as: for each client, find the most similar peers as its neighbors. To reach this objective, the Monte Carlo method adds the neighbors in the last round together with the random sampled peers in the current round to the candidate list. As the number of rounds increases, the expected probability of the most similar neighbors increases, and if the similarity measurement is effective, these similar neighbors are the same-clustered peers. We name this method as Neighbor Selection Based on Monte Carlo (NSMC). We summarize both the first stage of PENS and NSMC in one equation, as

$$N_i^t = \arg \max_N \sum_{j \in N} s_{i,j}$$

$$s.t. \ N^t \subseteq C_i^t \cup L, |N| = k.$$  \hspace{1cm} (7)

In Equation (7), for PENS, it has $L = \emptyset$ in all rounds; and for CNI, it has $L = \emptyset$ in the first round, and $L = N_i^{t-1}$ when $t > 1$. We will present theoretical analysis on PENS and NSMC in Section 4.5. It is found that the expected probability of true neighbors (being the same-cluster) is rapidly increasing in NSMC and keeps constant in PENS. We also show the empirical results that NSMC can boost personalization even better than Oracle in Section 5. NSMC facilitates clients to select peers with globally maximal similarities and these most similar peers may be more helpful than other peers in the same cluster.

4.3 Neighbor Augmentation Based on EM-GMM
After NSMC, we enable clients to have few neighbors in the neighbor bag with high probability of being true neighbors, in other words, the precision is high. For clustered FL, the recall of clustering is also essential since each client needs to find out the whole community with the same objective. Thus, in the second stage of PANM, we facilitate clients to discover more peers with consistent objectives.

In the second stage of PENS, clients choose peers that are selected more than “the expected amount of times” in the first round, and the number of neighbor candidates is $t > [33]$. We will present theoretical analysis on PENS and NSMC in Section 4.5. It is found that the expected probability of true neighbors (being the same-cluster) is rapidly increasing in NSMC and keeps constant in PENS. We also show the empirical results that NSMC can boost personalization even better than Oracle in Section 5. NSMC facilitates clients to select peers with globally maximal similarities and these most similar peers may be more helpful than other peers in the same cluster.
stage as neighbors. But if the setting is difficult, stage-one neighbors of PENs are prone to be noisy, afterward, in the second stage, the matched neighbors are more likely to include outliers. Besides, PENs requires the hyperparameter “the expected amount of times”, and without prior knowledge of cluster information, it is hard to set the hyperparameter to an appropriate value. To better solve these problems, we propose a more effective neighbor bag augmentation method, which is based on Expectation Maximization of Gaussian Mixture Model (EM-GMM).

For a client, given a set of randomly sampled peers, it is obvious that the true neighbors (i.e., with the same cluster identity) may have high similarities with it while the false ones (i.e., with different cluster identities) have low similarities. Therefore, we make the Gaussian Mixture Model assumption that the similarities of the true neighbors obey a consistent distribution while the similarities of the false ones obey another distribution. Formally, the assumption is shown in Assumption 1.

**Assumption 1.** (Gaussian Mixture Model Assumption) For client $i$ ($\forall i \in [n]$), the similarities between the true neighbors and client $i$ obey a Gaussian distribution, parameterized by $\mathcal{N}(\mu_0, \sigma_0)$, and the similarities between the false neighbors and client $i$ obey another Gaussian distribution $\mathcal{N}(\mu_1, \sigma_1^2)$

$$s_{i,p} \sim \mathcal{N}(\mu_0, \sigma_0^2), \quad s_{i,q} \sim \mathcal{N}(\mu_1, \sigma_1^2)$$

where $N_i^t$ refers to the true neighbors of client $i$, and $\overline{N_i^t}$ refers to the false neighbors of client $i$. We have $\mu_0 > \mu_1$.

Assumption 1 is quite natural in clustered FL. In Fig. 3, intuitively, the distributions of similarities satisfy our assumption that there are two distinct Gaussians.

Under Assumption 1, we can implement neighbor augmentation by solving the GMM problem and a typical solution to GMM problem is the EM algorithm. But conventional EM algorithm is not suitable to solve this problem under the following considerations: (1) Conventional EM method requires calculating probabilities of all data points in one EM step, but for P2P FL, clients only can communicate with several neighbors in one round. (2) The focus of conventional EM solving GMM problems is to accurately estimate the parameters of Gaussians while our focus is to accurately discriminate cluster identities. (3) Additionally, EM algorithms are sensitive to initialization, and poor initialization may lead to bad convergence.

To tackle above mentioned matters, we devise our Neighbor Augmentation Based on EM-GMM (NAEM) algorithm. In each round, client $i$ randomly samples neighbor candidate list $\mathcal{C}_i^t$ from non-neighbor clients and also samples a selected neighbor list $\mathcal{S}_i^t$ from the neighbor bag $\mathcal{B}_i^t$. Then, client $i$ communicates with these clients and compute similarities $y_{ij} = s_{ij}, j \in \mathcal{M}_i^t = C_i^t \cup S_i^t$. According to Assumption 1, there are two Gaussian distributions in these similarities, the one with higher mean center refers to the same-cluster clients ($\mathcal{N}(\mu_0, \sigma_0^2)$), another one refers to the different-cluster ($\mathcal{N}(\mu_1, \sigma_1^2)$). Assuming the observed similarity $y_{ij}, j \in \mathcal{M}_i^t$ is generated by the Gaussian Mixture Model:

$$\Pr(y_j|\Theta) = \sum_{r=0}^{1} \beta_r \phi(y_j|\Theta_r).$$

Here, $\beta_r$ refers to the overall probability that $y$ is generated by distribution $r$, and $\Theta = (\beta_0, \beta_1; \Theta_0, \Theta_1)$. Our target is using EM algorithm to estimate the distribution identities of $y_j$, given by

$$y^*_{j,r} = \begin{cases} 1, & \text{if } j \text{ belongs to distribution } \mathcal{N}_r; \\ 0, & \text{otherwise.} \end{cases}$$

where $j \in \mathcal{M}_i^t$, $r \in \{0, 1\}$. Knowing that EM algorithm is sensitive to initialization, with the prior knowledge that most of the clients in $\mathcal{S}_i^t$ are true neighbors, so we can initialize a better parameter as

$$y^{(1)}_{j,r} = \begin{cases} 1, & j \in \mathcal{S}_i^t \text{ and } r = 1, \text{ or } j \in \mathcal{C}_i^t \text{ and } r = 0; \\ 0, & \text{otherwise.} \end{cases}$$

While the latent variable is the distribution parameters: $\Theta_0 = (\mu_0, \sigma_0), \Theta_1 = (\mu_1, \sigma_1)$, so the complete data is

$$(y_j, \Theta_0, \Theta_1), j \in \mathcal{M}_i^t.$$ 

Then we formulate the expectation function $Q$, based on the log likelihood function of complete data,

$$Q(\gamma, y^{(a)}) = \mathbb{E}[\log \Pr(y_j|\Theta)|y^{(a)}]$$

$$= \sum_{r=0}^{1} \left\{ n_r \log \mathbb{E}\beta_r + \sum_{j \in M^t_i} y^*_{j,r} \left[ \log \left( \frac{1}{\sqrt{2\pi}} \right) \\ - \log \mathbb{E}\sigma_r - \frac{1}{2\mathbb{E}\sigma_r^2} (y_j - \mathbb{E}\mu_r)^2 \right] \right\},$$

where $n_r = \sum_{j \in M^t_i} y^*_{j,r}$, $a$ denotes the iteration step. $E$-Step. Now we need to estimate $E(\mu_r, \sigma_r, \beta_r)$, notated as $\tilde{\mu}_r, \tilde{\sigma}_r, \tilde{\beta}_r$.

$$\tilde{\mu}_r = \frac{\sum_{j \in M^t_i} y^*_{j,r} y_j}{n_r}, \quad \tilde{\sigma}_r = \frac{\sum_{j \in M^t_i} y^*_{j,r} (y_j - \tilde{\mu}_r)^2}{n_r}, \quad \tilde{\beta}_r = \frac{n_r}{|M^t_i|}.$$ 

where $r \in \{0, 1\}$.

$M$-Step. Iterative $M$-step is to find the maximum of the function $Q(\gamma, y^{(a)})$ with respect to $y^{(a)}$, as to set $y^{(a+1)}$ in the next iterative epoch

$$y^{(a+1)} = \arg \max_{y} Q(\gamma, y^{(a)}).$$

We use the following function to maximize expectation, since $y_{ij}$ more likely belongs to $\mathcal{N}_0$ if $\beta_0\phi(y_j|\Theta_0) > \beta_1\phi(y_j|\Theta_1)$.
TABLE 2
Complexity Analysis Regarding Computation and Communication

| Methods | Similarity Computation | Communication cost | Maximal required bandwidth |
|---------|------------------------|--------------------|---------------------------|
| FedAvg  | $O(dcn(T_1 + T_2))$   | $O(n(T_1 + T_2))$  | $O(n)$                    |
| IFCA    | $O(dcn(T_1 + T_2))$   | $O(n(T_1 + T_2))$  | $O(n)$                    |
| PENS    | $O(d(nT_1 + dkT_2))$  | $O(nT_1 + nkT_2)$  | $O(l)$                    |
| PANMLoss| $O(d(k + 1)nT_1 + d(wT_2))$ | $O(n(l + k)T_1 + n(lT + kT_2))$ | $O(l + k)$|
| PANMGrad| $O(d(k + 1)nT_1 + n(lT + kT_2))$ | $O(n(l + k)T_1 + n(lT + kT_2))$ | $O(l + k)$|

Please refer to Table 1 for the meaning of notations.

$\beta_1 \phi(y_j|\Theta_1)$, vice versa.

$$
\gamma^{(a+1)}_{j,r} = \begin{cases} 
1 & \text{if } r = \arg \max_r \frac{\hat{\beta}_r \phi(y_j|\Theta_1)}{\sum_{l=0}^{n} \hat{\beta}_l \phi(y_j|\Theta_1)} \\
0 & \text{otherwise}
\end{cases},$

$$j \in M_r', r \in \{0, 1\}.$$

Repeat the E-step and M-step until $\gamma^{(a+1)} = \gamma^{(a)}$. Then we obtain the estimated true neighbors in this round noted as $H_r'$, where $\gamma'_{j,0} = 1, j \in H_r', H_r' \subseteq M_r'$, then we update the neighbor bag,

$$B_r^{t+1} = (B_r^t - S_r^t) \cup H_r'.$$

(12)

By the NAEM algorithm, clients can continually update their neighbor bags, adding new same-cluster peers and removing outliers in the neighbor bag. For model aggregation, clients can conduct gossip communication with the peers in the neighbor bag.

4.4 PANM: Personalized Adaptive Neighbor Matching

Now we present PANM by combining the algorithms mentioned above. In the first stage, client $i (i \in [n])$ communicates randomly in the network while conducting NSMC for $T_1$ rounds. The neighbor list $N_i^0$ in the last round of the first stage is set as the initial neighbor bag in the second stage, $B_i^{T_1 + 1} = N_i^{T_1}$. In the second stage, client $i$ operates gossip consultation with peers $N_i^{T_1} (N_i^{T_1} \subseteq B_i^{T_1}, |N_i^{T_1}| = k)$ sampled from the neighbor bag for aggregation and performs NAEM every $r$ rounds to update the neighbor bag $B_i^r$. The process of PANM is shown in Algorithm 1.

Complexity Analysis of PANM. We conduct complexity analysis regarding similarity computation, communication cost, and maximal required bandwidth as shown in Table 2. In Table 2, FedAvg and IFCA are centralized FL algorithms, and IFCA is the state-of-the-art clustered FL approach. PENS and our PANMLoss and PANMGrad are P2P FL methods. For the similarity computation, the complexity of the centralized IFCA relies on the assumed number of clusters ($c$), while the P2P methods rely on the number of communicating peers ($l, k$). We notice the computation is much more efficient in PANMGrad since it does not rely on local data inference. For communication cost, IFCA has more overhead than FedAvg, and in a sparse P2P network where $l \approx c$, the P2P methods have a similar overhead to IFCA. For maximal required bandwidth, P2P approaches have a dominant advantage over the centralized since $n \gg (l + k)$ is usually held in practice. We notice that PANM has more overhead compared with PENS in the first stage, but PANM realizes a more pure neighbor selection, that we can set a relatively smaller $T_1$ for PANM to reduce the overhead.

Applying PANM Under Server-Client Protocol. In applications, if only the server-client protocol is available, PANM can still be applied by simply taking the server as a relay for transmitting models. Concretely, clients send local models and requests about which peers it wants to communicate with to the server, and then the server sends the requested models to corresponding clients. The similarity measurement and aggregation process are implemented on the client side. Clients update their neighbor lists and generate the request in each round. The scheme that the server sends other clients’ models to a client for updating the client similarity matrix is adopted in personalized FL [35], showing this scheme is realistic in practice. Nevertheless, our method is essentially a P2P algorithm where clients have the autonomy to choose the neighbors, while in [35], the server updates the client similarity matrix and has the autonomy.

Algorithm 1. PANM: Personalized Adaptive Neighbor Matching

Input: $n, k, l, T_1, T_2, \eta, E, T, \alpha, w_0, W^T = \{w_i^0 = w_0, i \in [n]\}$
Output: $W^{T_1 + T_2}, B_i^r$
1: Initiate neighbor list: $N_i^0$
2: for each round $t = 1, \ldots, T_1 + T_2$ do
3:  for each client $i, i \in [n]$ in parallel do
4:  Compute $E$ epochs of local training:
5:  $w_i^{t-\frac{1}{2}} \leftarrow w_i^{t-1} - \eta E_i(w_i^{t-1})$
6:  if $l \in [T_1]$ then
7:  $N_i^t \leftarrow \text{NSMC}(N_i^t)$
8:  $w_i^t \leftarrow \text{Aggregation}(N_i^t, w_i^{t-\frac{1}{2}})$
9:  else
10:  $B_i^{t+1} \leftarrow N_i^{T_1}$
11:  if $l \leq \tau = 0$ then
12:  $B_i^t \leftarrow \text{NAEM}(B_i^{t-1})$
13:  $N_i^t \leftarrow \text{RandomSample}(B_i^t)$
14:  $w_i^t \leftarrow \text{Aggregation}(N_i^t, w_i^{t-\frac{1}{2}})$
15:  else
16:  $B_i^t \leftarrow B_i^{t-1}$
17:  $N_i^t \leftarrow \text{RandomSample}(B_i^t)$
18:  $w_i^t \leftarrow \text{Aggregation}(N_i^t, w_i^{t-\frac{1}{2}})$
19:  end if
20: end if
21: end for
22: end for

3. We note that the EM steps in NAEM require much little computation because: (1) the computation of similarity measurement includes the operation of high-dimensional vectors, while the EM steps only operate a small set of scalars (the similarity values); (2) the number of similarity values is small; (3) due to our better initialization, it is faster to converge. Thus, we omit the computation of EM steps in the analysis.
4.5 Theoretical Analysis

In this section, we provide theoretical analysis. We first give the probability model about the expected probability that all neighbors are true in Theorem 1. It shows that the Monte Carlo method will enable PANM to have an increasingly pure neighbors in the first stage, while PENS has constant expectation on the purity. Then, we provide a unified framework about the one-round error bound to the clustered optimum in Theorem 2. In conjunction with Theorems 1 and 2, we deduce the error bound of PANM in Theorem 3.

Probability Model About the Purity of Neighbors. We first deduce the probability model by providing the assumption about the effectiveness of similarity measurement.

Assumption 2. (Effectiveness of Similarity Measurement)
The metrics in Equations (3) and (6) are effective enough, so that for client $i$ ($\forall i \in [n]$), we have:

$$s_{i,p} > s_{i,q} \quad \forall p \in N^+_i, \ q \in \bar{N}^+_i. \quad (13)$$

Given Assumption 2, we can provide the expected probability that all neighbors are true in round $t$. Recall that there are $n$ clients in the system (including client $i$) and $a$ clients in the same cluster as client $i$ (including client $i$), so we can infer the following theorem.

**Theorem 1.** (Expected Probability of True Neighbors) Under Assumption 2, in the round $t$ when conducting NNSC as Equation (7) where $L = N^{-1}_i$ when $t > 1$; for client $i$, the expected probability that all neighbors are true is $P^e(k)$, we have

$$P^e(k) = G(k) * P^e(k) + R(k)$$

$$P^e(k) = G(k) * P^e(k) + R(k)$$

$$P^e(k) = G(k) * P^e(k) + R(k). \quad (14)$$

where $R(x)$ and $G(x)$ are two functions and the ‘*’ refers to the discrete convolution computation, defined as

$$R(x) = \frac{(a - 1)!(n - a)!}{((n - 1)!) \sum_{s=0}^{l-x} s!(l-s)!((n-a-s)!(a-l+s-1)!}$$

$$G(x) = \frac{(a - 1)!(n - a)!}{(n-a)!((n-a)!((n-a-x)!(a-l-x)!}$$

$$G(x) * P(x) = \sum_{m=0}^{x} G(m)P(x - m).$$

If conducting PENS as Equation (7) where $L = \emptyset$, the expected probability is

$$P^e(k) \equiv R(k). \quad (15)$$

Based on Theorem 1, we provide the following corollary.

**Corollary 1.** (The Monotonicity of Probability Functions)

Given Theorem 1, we denote $Q(t) = P^e(k), t \in [T]$ as the probability function of round $t$. The function $Q(t)$ of NNSC is monotone increasing, so we have

$$Q(t) > Q(t - 1) > \cdots > Q(2) > Q(1). \quad (16)$$

| $n, a, l, k$ | $t = 3$ | $t = 5$ | $t = 7$ |
|--------------|----------|----------|----------|
| 200,50,10,5  | 7.29     | 90.75    | 7.29     |
| 200,50,20,10 | 0.98     | 96.24    | 0.98     |
| 200,50,20,6  | 38.00    | 99.95    | 38.00    |
| 100,50,10,5  | 62.97    | 100.00   | 62.97    |

The results are shown in percentage (%).

The function $Q(t)$ of PENS is constant which satisfies

$$Q(t) = Q(t - 1) = \cdots = Q(1) = R(k). \quad (17)$$

The proofs of Theorem 1 and Corollary 1 are shown in the Appendix, available online. The Corollary 1 shows that: by NNSC, the probability increases over round, while PENS keeps it unchanged at a low value. Intuitively, we calculate the theoretical probability under different settings in Table 3. It is obvious that the probability of NNSC increases fast, it will reach 100% in round 5; whereas PENS will have constantly low probability.

However, we note that Assumption 2 is strong, and we provide this assumption just for theoretical analysis. There are gaps between theory and practice, especially in the first couple of rounds, when the similarity measurement is not effective enough. If the effectiveness of similarity is not strongly hold as Assumption 2, the probability in Theorem 1 will have a discount, but the monotone increasing property of NNSC in Corollary 1 still holds.

Error Bound to the Clustered Optimum. Inspired by [11], we propose a general theoretical framework to analyse the convergence and error bound under clustered heterogeneity in P2P FL. Then we incorporate Theorem 1 to give the error bound of PANM. To start with, we first give the following definitions.

**Definition 1.** (Optimality Within a Cluster) Knowing that there are $r$ clusters, for the data distribution of cluster $j$ ($j \in [r]$), we define the population loss of cluster $j$ and the optimal model of cluster $j$ as

$$F^j(w) := E_{\xi \sim P_j}[f(w, \xi)],$$

$$w^*_j := \arg \min_w F^j(w).$$

**Definition 2.** (Cluster Heterogeneity) We define the cluster heterogeneity as the maximal distance between the optimal models of each cluster.

$$\Delta := \max_{i \neq j, j \in [r]} \|w^*_i - w^*_j\|.$$
Thus, the number of true neighbors is \(|N_i \cap N_j^{|i}| = k(1-\epsilon)\), and the number of false neighbors is \(|N_i \cap N_j^{|j}| = k\epsilon\).

Then, we give the following assumptions for theoretical analysis.

Assumption 3. (\(\mu\)-strongly Convexity) The loss functions of each client \(F_i(w), \forall i \in [n]\) and each cluster \(F_j^{(i)}(w), \forall j \in [r]\) are all \(\mu\)-strongly convex that satisfy: \(\forall w, w'\),

\[
F(w') \geq F(w) + \langle \nabla F(w), w' - w \rangle + \frac{\mu}{2} ||w' - w||^2.
\]

Assumption 4. (L-smoothness) The loss functions of each client \(F_i(w), \forall i \in [n]\) and each cluster \(F_j^{(i)}(w), \forall j \in [r]\) are all \(L\)-smooth that satisfy: \(\forall w, w'\),

\[
F(w') \leq F(w) + \langle \nabla F(w), w' - w \rangle + \frac{L}{2} ||w' - w||^2.
\]

Assumption 5. (Bounded Gradient Variance) We bound the variance of gradients within a cluster. For every \(w\) and every \(j \in [r]\), the variance of \(\nabla f(w, \xi)\) is upper bounded by \(v^2\), when \(\xi\) is sampled from \(D^j\).

\[
E_{\xi \in D^j} ||\nabla f(w, \xi) - \nabla F_j^{(i)}(w)||^2 \leq v^2.
\]

Given the above definitions and assumptions, we present the theorem about the error bound within one communication round in arbitrary P2P algorithms.

Theorem 2. (Error Bound Within One Communication Round) For a client \(i, i \in [n]\), which belongs to cluster \(j, j \in [r]\), in a certain communication round, the error gap between its model to the clustered optimum is \(||w_i - w_j^*||\). Let \(w_i^+\) be the next-round model after communicating with neighbors. Then, the next-round error gap is

\[
||w_i^+ - w_j^*|| \leq \left(1 - \frac{\eta_j L(1 - \epsilon)}{\mu + L} + \eta_j L\epsilon\right) ||w_i - w_j^*|| + \eta_j L \epsilon + \frac{v}{\sqrt{d}} \sqrt{1 - \epsilon} + \eta_j v \sqrt{\frac{r}{kd}} \sqrt{\epsilon}.
\]

Remark 1. (Error Bound under Different \(\epsilon\)) Note that Theorem 2 is a unified bound for any algorithm and the differences between algorithms lie in the error rate of neighbor estimation \(\epsilon\).

If the algorithm is effective enough that have \(\epsilon \to 0\), we have

\[
||w_i^+ - w_j^*|| \leq \left(1 - \frac{\eta_j L(1 - \epsilon)}{\mu + L}\right) ||w_i - w_j^*|| + \frac{v}{\sqrt{d}}.
\]

In this case, it is clear that the \(||w_i - w_j^*||\) term is decreasing, and if \(d, k\) are large while \(v\) is small, the model will converge to the clustered optimum.

If the algorithm is dump that \(\epsilon \to \frac{k - 1}{k}\), we have

\[
||w_i^+ - w_j^*|| \leq (1 + \eta_j L) ||w_i - w_j^*|| + \eta_j L \epsilon + \eta_j v \sqrt{\frac{r}{kd}} + \frac{v}{\sqrt{d}}.
\]

In this case, \((1 + \eta_j L)\) shows that it is not converging to the clustered optimum. If the clustered heterogeneity is more dominant with larger \(\Delta\), the model will be further away from the optimum.

From Equation (18), we can infer that if \(1 - \frac{\eta_j L(1 - \epsilon)}{\mu + L} + \eta_j L \epsilon > 1 \iff \epsilon > \frac{\mu}{\mu + L},\) the convergence to the clustered optimum is not satisfied. More intuitively, if we assume \(\mu = L\), the condition becomes \(\epsilon > \frac{1}{2}\), which means if the proportion of false neighbors is larger than \(\frac{1}{2}\) in each round, it is impossible to converge to the clustered optimum. Therefore, the neighbor estimation and selection is quite essential for clients to converge to the clustered optimum.

Theorem 3. (Error Bound of PANM in the First Stage) We assume Assumptions 1-5 hold and \(\mu = L\), and set the learning rate \(\eta = \frac{1}{2}\). We analyse the error bound of client \(i, i \in [n]\), which belongs to cluster \(j, j \in [r]\), when applying PANM in the P2P FL system. The initial error gap is defined as \(\delta_0 = ||w_i - w_j^*||\). After \(T\) rounds, the error bound is

\[
||w_i^T - w_j^*|| \leq \frac{1}{2^{T-1}} \left[\frac{1 + 3\epsilon_0}{2} \delta_0 + \epsilon_0 \Delta + \frac{v}{\sqrt{dk(1 - \epsilon)}} + \frac{v}{L \sqrt{kd}} \right] + \sum_{t=0}^{T-1} \frac{v}{2^{t+1}} \sqrt{\epsilon}.
\]

where \(\epsilon_0 = R(k) = \frac{2^{(n-1)!}}{(n-1)!} \sum_{x=0}^{n-1} \frac{(-1)^{n-1-x}}{(n-x-1)! (n-x-1)^{n-x-1}}\).

The proofs of Theorems 2 and 3 are shown in the Appendix, available in the online supplemental material. From Theorems 2 and 3, we prove that PANM can converge to the clustered optimum and it has a linear convergence rate. It is worthy mentioning that the effect of clustered heterogeneity \(\Delta\). Intuitively, more dominant heterogeneity will result in more effective similarity measurement, in other words, Assumption 2 is more likely to be held and the error rate of neighbor estimation \(\epsilon\) is lower. In Remark 1, lower error rate will result in faster convergence to the optimum. However, on the other hand, larger \(\Delta\) will have a more dominant error term in Theorem 3.

If we can formulate \(\epsilon\) into a function of \(\Delta\), the effect of heterogeneity is more tractable. For instance, if \(\epsilon = \mathcal{O}(\frac{1}{\Delta})\), the error term \(\epsilon \Delta = \mathcal{O}(\frac{1}{\Delta})\), in this case, larger heterogeneity will benefit; and if \(\epsilon = \mathcal{O}(\frac{1}{\sqrt{\Delta}})\), the error term \(\epsilon \Delta = \mathcal{O}(\sqrt{\Delta})\), thus, larger heterogeneity will lead to larger error gap; and if \(\epsilon = 0\) in all rounds, the effect of \(\Delta\) will be removed.

5 EXPERIMENTS AND RESULTS

In this section, we evaluate our methods and compare them with baselines. P2P FL baselines include PENS [33] (state-of-the-art personalized P2P FL algorithm), Random (gossip with random neighbors), Local (without communication), FixTopology (neighbors are randomly sampled at the beginning and fixed during training). We also include Oracle (with prior knowledge of cluster identities, gossip with true neighbors) for comparison, and it is not a baseline but the ideal gossip algorithm with ground-truth cluster information, which is not realistic in practice. Oracle may indicate the upper bound of accuracy in clustered P2P FL, but the following experiments will show our methods can sometimes surpass it.
Centralized FL baselines include IFCA [11] (state-of-the-art centralized clustered FL) and centralized FedAvg [5]. Our methods include PANMLoss (PANM with metric based on loss), PANMGrad (PANM with metric based on weight updates and gradients).

5.1 Settings of Datasets

**Synthetic Clustered Heterogeneity.** We use three public benchmark datasets, MNIST [36], CIFAR10 [37], and FMNIST (Fashion-MNIST) [38]. To synthesize clustered heterogeneity, we use rotation transformation and label-swapping, respectively; we note these two settings are commonly used in clustered FL (rotation [11], [33], label-swapping [13]). All results are evaluated on each client’s local testset, and we keep the size of the local testset to 100 for all scenarios and present the averaged results among all clients. We then describe the details of clustered rotation transformation and label-swapping.

**Rotation Transformation:** There are two settings for rotation transformation. First is rotation with two clusters (0°, 180°): clients in cluster 0 keep images without any transformation (0° rotation), while clients in cluster 1 rotate every image in the trainset and testset for 180°. The second is rotation with four clusters (0°, 90°, 180°, 270°): clients in cluster 0 keep images without any transformation (0° rotation), and clients in cluster 1 rotate every image in the trainset and testset for 90°, clients in cluster 2 for 180°, and clients in cluster 3 for 270°. The labels of images remain unchanged, and each client’s class distributions are balanced.

**Swapping Labels:** There are two settings for swapping labels. First is forming two clusters by swapping labels: for clients in cluster 0, images labeled as “0” are relabeled as “1” and images labeled as “1” are relabeled as “0”; while for clients in cluster 1, images labeled as “6” are relabeled as “7” and images labeled as “7” are relabeled as “6”. The second is forming four clusters by swapping labels: (1) for clients in cluster 0, images labeled as “0” and “1” are swapped by labels; (2) for clients in cluster 1, images labeled as “2” and “3” are swapped; (3) for clients in cluster 2, images labeled as “4” and “5” are swapped; (4) for clients in cluster 3, images labeled as “6” and “7” are swapped. Note that the class distributions in clients are balanced.

**Real-World Clustered Heterogeneity.** We also use Digit-five [39], [40], [41] to validate the algorithms. Digit-five is a collection of five digital recognition datasets, namely handwritten digits (MNIST) [42], digits with colored backgrounds (MNIST-M) [43], street images of digits (SVHN) [44], synthetic digits (Synthetic Digits) [45], and digits from postal services (USPS) [46]. These datasets are of different domains and modalities, and assigning these datasets to clients can inherently realize clustered heterogeneity. USPS is much smaller than others, so we exclude USPS and select 50000 samples from each of the other four domains as the FL trainset. As a result, there exist four clusters among clients. To enable the training under the same model architecture, we transform the images in different domains to the same size and number of channels.

5.2 Details of Implementations

**Implementation Environment.** All the experiments are implemented in PyTorch 1.7.1. We have several GPUs for training, including Tesla P40 GPU with 24451MB memory, Quadro RTX 8000 GPU with 48601MB memory, Tesla P100 GPU with 16280MB memory, and Tesla V100 GPU with 16130MB memory.

**Clients’ Models.** A three-layer MLP with ReLU activations is adopted as the model for training on MNIST and FMNIST. For CIFAR10, a five-layer convolution neural network model (three convolutional layers followed by two fully connected layers) is used. For Digit-five, we use LeNet-5 [46]. We do not use data augmentation techniques like flipping and random cropping.

**Hyperparameters.** We set the batch size for all experiments to 128, and the number of local epochs in each round is 3. We adopt the learning rate decay strategy used in IFCA [11]. The decay step size is 0.99, which means for each round, the learning rate is set to the learning rate in the last round × 0.99. The initial learning rate is set to 0.08 in the first round. We use the SGD optimizer and set SGD momentum to 0.9. We set T₁ = 100, T₂ = 200 in all experiments. For “the expected amount of times” in PENS, we set this hyperparameter to an appropriate value ceil(T₁ × (l + k)/n) for fair comparison. For centralized FL: IFCA and FedAvg, we set full participation of clients in each round. For PANMGrad, we set α = 0.5 in all experiments.

**Result Presentation.** In every setting, we conduct experiments with different random initialization three times and average the results, and the mean results and the standard deviations are shown in the tables and figures. All experimental results are shown in percentage value (%).

5.3 Results on Different Datasets

Table 4 shows the test accuracies of all methods under different datasets. Although FixTopology and Random improve accuracy compared with local training, in contrast to Oracle, the fixed or random topology will impede performance gains.

Recall Oracle has perfect information about cluster identities which is impossible in real FL scenarios. Notably, our methods surpass Oracle in FMNIST and CIFAR10. We show the accuracy and loss curves of results in CIFAR10 in (a) and (b) of Fig. 5. It is demonstrated that PANMLoss achieves high accuracy and fast convergence in the first stage, which means the NSMC algorithm is effective. We explain that the Monte Carlo method enables the clients to collaborate with peers with maximal similarities, so the performance will be better than randomly sampled from same-cluster peers (Oracle). This explanation is also validated in the heatmap of aggregation records between clients in

| Methods         | MNIST | FMNIST | CIFAR10 |
|-----------------|-------|--------|---------|
| Local           | 82.57 ± 0.28 | 76.24 ± 0.22 | 25.27 ± 1.21 |
| FixTopology     | 94.71 ± 0.09  | 85.86 ± 0.17  | 39.74 ± 2.27  |
| Random          | 95.12 ± 0.04  | 85.94 ± 0.27  | 42.96 ± 1.42  |
| Oracle          | 95.87 ± 0.08  | 87.01 ± 0.26  | 49.11 ± 0.48  |
| PENNS           | 96.15 ± 0.16  | 86.82 ± 0.11  | 44.78 ± 1.12  |
| PANMLoss        | 95.63 ± 0.12  | 87.33 ± 0.17  | 49.19 ± 0.79  |
| PANMGrad        | 95.65 ± 0.09  | 86.88 ± 0.34  | 48.83 ± 0.39  |

*The top two are in bold. For all datasets: trainset size is 200, two rotations (0°, 180°), l = 10, k = 5, n = 100 for CIFAR10 and FMNIST, n = 200 for MNIST.*
Fig. 4. In PANM, the neighbors are pure compared with PENS and clients are prone to communicate with several peers instead of all true neighbors.

Additionally, we notice that PANM performs normally in MNIST, and we reckon this is because the clustered heterogeneity $\Delta$ is not dominant in rotated MNIST. One example is that if we rotate the images of ‘0’ for 180°, the images represent the same distribution. PANM will not take full advantage when $\Delta$ is small. First, as we have discussed in Section 4.5, smaller $\Delta$ will make the similarity metrics less effective. More importantly, in this case, the different clustered optimums have close distances, which means that the negative transfer between clusters is weak, so there is less necessity for clustering.

5.4 Results Under Various Clustered Heterogeneity

Table 5 shows results under various heterogeneity: swapping labels and more rotations. For test accuracy, it is evident that our methods are robust in different Non-IID environments. It is notable that in some experiments (CIFAR10 Label-swap(2) and (4)), PANM even outperforms Oracle with a large margin. PANMGrad has similar performances to PANMLoss, but performances vary in different settings. This is due to their different perspectives of client similarity that the loss-based and gradient-based perspectives will take advantage in different scenarios.

For precision (the fraction of true neighbors in the neighbor bag) and recall (the fraction of estimated true neighbors among all true neighbors) of neighbor estimation in the second stage, our methods outperform PENS. It indicates our EM-based NAEM method is effective in enabling clients to match most of the true neighbors. We also demonstrate the neighbor topology in the second stage in Fig. 6. In PANM,

| METHODS | LABEL-SWAP(2) | LABEL-SWAP(4) | ROTATION(4) |
|---------|---------------|---------------|-------------|
| LOCAL   | 25.27 ± 1.21  | 25.27 ± 1.21  | 25.27 ± 1.21|
| FixTopology | 36.56 ± 1.58  | 35.08 ± 2.70  | 31.86 ± 0.47|
| Random  | 37.32 ± 0.26  | 37.14 ± 1.30  | 33.16 ± 0.55|
| Oracle  | 43.34 ± 1.29  | 43.32 ± 1.06  | 43.32 ± 0.85|
| PANM    | 45.72 ± 1.34  | 43.49 ± 0.28  | 36.64 ± 0.58|
| PANMLoss | 47.12 ± 1.30  | 45.78 ± 1.85  | 41.43 ± 1.83|
| PANMGrad | 45.84 ± 1.92  | 42.14 ± 1.34  | 43.99 ± 1.26|

| METHODS | LABEL-SWAP(2) | LABEL-SWAP(4) | ROTATION(4) |
|---------|---------------|---------------|-------------|
| PENS    | 100           | 100           | 68.19 ± 28.12|
| PANMLoss | 100           | 100           | 83.33 ± 28.87|
| PANMGrad | 100           | 100           | 100          |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
clients evolve to form the four-cluster structure without prior knowledge of cluster identities, whereas PENS and Random construct disordered topologies.

### 5.5 Impact of \( l \) and \( k \)

\( l \) is the size of neighbor candidate list, and \( k \) is the size of the aggregation neighbor list. The choices of \( l \) and \( k \) depend on communication budgets in the system, and they determine the network connectivity. Larger \( l \) and \( k \) will result in denser network connectivity and bring more communication costs. Besides, the ratio of \( l \) and \( k \) is also crucial (especially for PENS), and it decides the purity of neighbors as we have inferred in Theorem 1 and Corollary 1. We conduct experiments under different network setups by changing \( l, k \) as demonstrated in Table 6.

We notice if \( l \) and \( k \) are small (for example, \( l = 10, k = 3 \)), PENS performs well, but in other settings, PENS has poor results. As we have discussed in Section 4.5, if \( l, k \) are not set appropriately, PENS is prone to be noisy in neighbor matching.

For Oracle, the communication only relies on \( k \). As \( k \) increases, the network connection is denser, and the performance of Oracle increases. We explain that if the neighbor bag is pure and the number of aggregation neighbors increases, it is better for clients to reach partial consensus within clusters, and the model weights are more likely to be similar. It is worth mentioning that PANMLoss is very robust under various network setups, surpassing Oracle in most settings. PANMLoss benefits a lot when \( k \) increases.

### 5.6 Impact of Trainset Size and Number of Clients

In the left figure of Fig. 7, we compare the methods by varying the size of the local trainsets, and in the right figure of Fig. 7, we show the results of changing the number of clients. We see that PANM consistently outperforms all baselines (Local, PENS, Random, and FixTopology), and it has comparable performance with Oracle.

### 5.7 Comparison With Centralized Clustered FL

P2P FL takes advantage of bandwidth and reliability, as we addressed in Section 1. Besides, as for clustered FL, our P2P solution is more robust and can exploit the latent cluster structure in a self-evolved manner without assuming the number of clusters. We compare IFCA [11], the state-of-the-art centralized clustered FL and centralized FedAvg [5] with decentralized P2P methods, as shown in Table 7 and (c) of Fig. 5. In typical centralized FL algorithms, the central server randomly samples \( l \) clients for aggregation, but we notice this will result in bad convergence for IFCA. In our implementations, in a scenario where there are 100 clients with 4 clusters, and the central

---

**Table 6**

| Methods       | \( l, k \)  |
|---------------|-------------|
|               | 10,5        | 10,3        | 20,10       | 20,5        | 30,15       | 30,10       |
| Local         | 19.93 ±0.36 | 19.93 ±0.36 | 19.93 ±0.36 | 19.93 ±0.36 | 19.93 ±0.36 | 19.93 ±0.36 |
| FixTopology   | 36.60 ±2.58 | 31.54 ±0.18 | 38.31 ±1.45 | 34.94 ±0.84 | 38.88 ±0.52 | 38.77 ±3.80 |
| Random        | 39.19 ±1.04 | 38.54 ±2.24 | 40.82 ±1.09 | 37.34 ±0.80 | 42.22 ±2.25 | 39.91 ±0.60 |
| Oracle        | 43.83 ±0.80 | 42.28 ±1.25 | 43.39 ±2.80 | 43.57 ±0.93 | 44.79 ±2.36 | 44.16 ±1.32 |
| PENS          | 42.42 ±1.58 | 40.04 ±0.90 | 42.57 ±1.74 | 41.92 ±1.28 | 44.12 ±0.22 | 42.80 ±0.98 |
| PANMLoss      | 41.75 ±0.24 | 39.55 ±1.78 | 44.02 ±0.59 | 43.64 ±1.54 | 46.82 ±0.41 | 46.19 ±0.84 |
| PANMGrad      | 42.48 ±0.21 | 39.53 ±2.99 | 38.24 ±4.26 | 41.94 ±3.15 | 41.16 ±3.60 | 42.23 ±1.02 |

CIFAR10 with two rotations \([0^\circ, 180^\circ]\), \( n = 100 \), trainset size is 100 for all settings.
We have mentioned in Section 5.5 that when the number of clients is 400, the best assumed number of clusters is 40.64% while our method can dynamically form balanced in the curves of centralized counterparts.

Comparison Under Sparse Network. Generally, if $l$ and $k$ are large, the overall communication costs of P2P FL are larger than those of centralized FL methods. In Table 8, we conduct experiments where PANM will have comparable communication costs as the centralized methods. This setting depicts an extremely sparse network for PANM. The results show that PANM can have intimate performances to IFCA when the communication costs are similar, but PANM requires much lower bandwidth. We also find that the consequence of inappropriately estimating the number of clusters will be severe for IFCA (43.65% when $c = 4$; 41.05% when $c = 3$). On the contrary, PANM is flexible enough to explore the cluster structure under any hyperparameters, and the influence of communication budgets is subtle. Therefore, we reckon PANM is more robust and effective than the centralized IFCA.

### 5.8 Results Under Real-World Clustered Heterogeneity

We validate the algorithms on Digit-five benchmark to see how they perform under real-world clustered heterogeneity, and the results are demonstrated in Table 9. It shows PANM is robust and effective that it has high mean and low variance in accuracy even in the real-world clustered heterogeneity. When the number of clients is 400, PANM has even superior performance over Oracle.

Interestingly, we observe that even though the Digit-five dataset has four domains, the best assumed number of clusters is not necessarily four. For IFCA, when $n = 200$, the best $c$ is 3, and when $n = 400$, the best $c$ is 4. It reveals that in real-world data, the clustered structure is not absolute. When the number of clients changes that clients come in and out, the cluster relationship among them also varies. Therefore, it is not flexible of IFCA to assume the cluster number that inappropriate $c$ will cause poor and unstable performances. However, our method can dynamically form the clustered topology, which is more robust and effective.

### 6 Discussion on the Applicability of PANM

**Choice Between PANMLoss and PANMGrad.** For PANMLoss and PANMGrad, the metrics require different computation resources. As we have discussed in Section 4.4, generally, PANMGrad is more computationally efficient since it only requires several inner product calculations on sparse vectors (gradients), while PANMLoss needs inference on local datasets. Hence, under limited computation resources, PANMGrad is preferable. In addition, in conjunction with the results in Tables 6 and 8, we found PANMGrad has better performance under sparse communication while PANMLoss benefits more in dense connection. Therefore, under limited communication resources, PANMGrad is preferable. And when the computation and communication resources are sufficient, PANMLoss is the better choice with better performance.

**Client Accessibility.** We have mentioned in Section 5.5 that we can choose different $l$, $k$ by the communication budgets, server samples 10 clients in each round (where $l/n = 0.1$), IFCA has poor convergence of estimations. As a result, we have to set full aggregation participation of clients in IFCA (where $l/n = 1.0$), but we remind this will cause large communication burdens, and it is unfair to the P2P setting (where we set $l = 100$, $n = 1000$). Even if, in Table 7, our P2P method PANM also achieves proportionate performances as a contrast to IFCA. What's more, IFCA requires the assumption on the number of clusters ($c$), and we find that if $c$ is set inappropriately, the performance will be poor. According to Table 7, in CIFAR10 with 4 rotations setting, if set $c = 2$, the accuracy of IFCA is 40.64% while our PANMLoss and PANMGrad reach 41.43% and 43.99%. Moreover, from the learning curves in Figure (c) of Fig. 5, it is apparent that our PANM keeps a more steady and robust learning process while there are some disturbances in the curves of centralized counterparts.

| Methods       | CIFAR10(4) | FMNIST(4) | FMNIST(2) |
|---------------|------------|-----------|-----------|
| Local         | 25.27 ± 1.21 | 76.24 ± 0.22 | 76.24 ± 0.22 |
| FedAvg        | 37.03 ± 0.74 | 83.54 ± 0.08 | 86.86 ± 0.16 |
| IFCA(2)       | 40.64 ± 2.18 | 86.19 ± 0.04 | 88.06 ± 0.20 |
| IFCA(3)       | 41.05 ± 1.09 | 86.78 ± 0.36 | /          |
| IFCA(4)       | 43.65 ± 0.77 | 86.50 ± 0.07 | /          |
| Oracle        | 43.32 ± 0.85 | 85.45 ± 0.38 | 87.01 ± 0.26 |
| PENS          | 36.64 ± 0.58 | 84.68 ± 0.27 | 86.82 ± 0.11 |
| PANMLoss      | 41.43 ± 1.83 | 86.09 ± 0.31 | 87.33 ± 0.17 |
| PANMGrad      | 43.99 ± 1.26 | 85.64 ± 0.25 | 86.88 ± 0.34 |

$n = 100$, $l = 10$, $k = 5$, trainset size is 200. Best performances in the centralized and decentralized are in bold. Clusters are generated by rotations: 2 for $\{0^\circ, 180^\circ\}$, 4 for $\{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$.

| Methods       | Comm. costs | Max. req. band. | Test acc. |
|---------------|-------------|-----------------|-----------|
| FedAvg        | 600Ω        | 1Ω              | 37.03 ± 0.74 |
| IFCA(2)       | 900Ω        | 2Ω              | 40.64 ± 2.18 |
| IFCA(3)       | 1200Ω       | 3Ω              | 41.05 ± 1.09 |
| IFCA(4)       | 1500Ω       | 4Ω              | 43.65 ± 0.77 |
| PANMLoss(k=2) | 1118Ω       | 0.06Ω           | 43.36 ± 0.64 |
| PANMGrad(k=2) | 1118Ω       | 0.06Ω           | 42.78 ± 1.68 |
| PANMLoss(k=3) | 1397Ω       | 0.07Ω           | 43.30 ± 1.32 |
| PANMGrad(k=3) | 1397Ω       | 0.07Ω           | 43.34 ± 0.85 |

We use CIFAR-10 with 4 rotations. For PANM, we set $l = 4$, $r = 10$.

| Methods       | Number of Clients (n) |
|---------------|------------------------|
| Local         | 16.58 ± 0.14           | 18.38 ± 0.92 |
| FedAvg        | 80.94 ± 7.66           | 71.92 ± 15.8 |
| IFCA(2)       | 50.23 ± 1.04           | 66.34 ± 23.59 |
| IFCA(3)       | 91.34 ± 0.44           | 52.56 ± 8.82 |
| IFCA(4)       | 64.94 ± 28.33         | 78.73 ± 13.09 |
| FixTopology   | 73.44 ± 25.14         | 39.29 ± 5.13 |
| Random        | 65.95 ± 19.46         | 41.82 ± 9.04 |
| Oracle        | 91.48 ± 0.44          | 83.05 ± 12.41 |
| PENS          | 85.57 ± 1.05          | 86.52 ± 1.03 |
| PANMLoss      | 88.6 ± 1.63           | 87.04 ± 0.31 |
| PANMGrad      | 89.66 ± 0.51          | 90.24 ± 1.51 |

TABLE 9

Results on Digit-Five, $l = 10$, $k = 5$, and Trainset Size is 200
and we have shown PANM is still effective under low communication budgets. Furthermore, in our implementations, we assume fully connected communication accessibility, which means that each client can communicate with any other client in the system. The fully connected accessibility is not often satisfied in realistic scenarios, but we state that PANM is also applicable under limited accessibility. In PANM, we hold loose assumptions about accessibility, that clients can purify their neighbors within the scope of accessible peers. Thus, PANM is flexible in practice.

7 Conclusion
This article studies the clustered Non-IID problem in FL under P2P communication and develops PANM that enables clients to match neighbors with similar objectives. PANM is more flexible and effective than the centralized clustered FL methods because it does not require the assumption of the number of clusters. Specifically, in PANM, we present two novel metrics for measuring client similarity based on loss and gradient, respectively. Then, we propose a two-stage algorithm. In the first stage, an effective method based on Monte Carlo is proposed to enable clients to match neighbors with maximally high similarities. Then in the second stage, a method based on Expectation Maximization under the Gaussian Mixture Model assumption of similarities is used for clients to discover more neighbors with similar objectives. We have conducted theoretical analyses of PANM on the probability of neighbor estimation and the error gap to the clustered optimum. We have also implemented extensive experiments under both synthetic and real-world clustered heterogeneity. Theoretical analysis and empirical experiments show that the proposed algorithm is superior to the P2P FL counterparts and achieves better performance than the centralized cluster FL method. PANM is effective even under extremely low communication budgets.

References
[1] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, “A survey on Internet of Things: Architecture, enabling technologies, security and privacy, and applications,” IEEE Internet of Things J., vol. 4, no. 5, pp. 1125–1142, Oct. 2017.
[2] A. Imteaj, U. Thakker, S. Wang, J. Li, and M. H. Amini, “A survey on federated learning for resource-constrained IoT devices,” IEEE Internet of Things J., vol. 9, no. 1, pp. 1–24, Jan. 2022.
[3] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, “Federated learning for Internet of Things: A comprehensive survey,” IEEE Commun. Surveys Tutts., vol. 23, no. 3, pp. 1622–1658, Third Quarter 2021.
[4] Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated machine learning: Concept and applications,” ACM Trans. Intell. Syst. Technol., vol. 10, no. 2, pp. 1–19, 2019.
[5] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Arcas, “Communication-efficient learning of deep networks from decentralized data,” in Proc. Int. Conf. Artif. Intell. Statist., 2017, pp. 1273–1282.
[6] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, “Federated learning: Challenges, methods, and future directions,” IEEE Signal Process. Mag., vol. 37, no. 3, pp. 50–60, May 2020.
[7] P. Kairouz et al., “Advances and open problems in federated learning,” Found. Trends Mach. Learn., vol. 14, no. 1/2, pp. 1–210, 2021.
[8] H. Zhu, J. Xu, S. Liu, and Y. Jin, “Federated learning on non-IID data: A survey,” Neurocomputing, vol. 465, pp. 371–390, 2021.
[9] B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Recommender systems for large-scale E-commerce: Scalable neighborhood formation using clustering,” in Proc. 5th Int. Conf. Comput. Inf. Technol., 2002, pp. 291–324.
[10] Q. Li and B. M. Kim, “Clustering approach for hybrid recommender system,” in Proc. IEEE/WIC Int. Conf. Web Intell., 2003, pp. 33–38.
[11] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, “An efficient framework for clustered federated learning,” in Proc. Int. Conf. Inf. Process. Syst., 2020, pp. 1–9.
[12] M. Xie et al., “Multi-center federated learning,” 2020, arXiv:2005.01026.
[13] F. Sattler, K.-R. Müller, and W. Samek, “Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints,” IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 8, pp. 3710–3722, Aug. 2021.
[14] M. Duan et al., “Flexible clustered federated learning for client-level data distribution shift,” IEEE Trans. Parallel Distrib. Syst., vol. 31, no. 11, pp. 2661–2674, Nov. 2020.
[15] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, “Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent,” in Proc. Int. Conf. Neural Inf. Process. Syst., 2017, pp. 5336–5346.
[16] A. Lalitha, O. C. Kilinc, T. Javidl, and F. Koushanfar, “Peer-to-peer federated learning on graphs,” 2019, arXiv:1901.11173.
[17] A. Lalitha, S. Shekhar, T. D. Tran, and F. Koushanfar, “Fully decentralized federated learning,” in Proc. 3rd Workshop Bayesian Deep Learn., 2018.
[18] T. Lin, S. P. Karimireddy, S. U. Stich, and M. Jagg, “Quasi-global momentum: Accelerating decentralized deep learning on heterogeneous data,” 2021, arXiv:2102.04761.
[19] L. Kong, T. Lin, A. Koloskova, M. Jagg, and S. Stich, “Consensus control for decentralized learning,” in Proc. Int. Conf. Mach. Learn., 2021, pp. 5686–5696.
[20] L. Cui, X. Su, Y. Zhou, and J. Liu, “Optimal rate adaption in federated learning with compressed communications,” in Proc. IEEE Conf. Comput. Commun., 2022, pp. 1459–1468.
[21] L. Cui, X. Su, Y. Zhou, and Y. Pan, “Slashing communication traffic in federated learning by transmitting clustered model updates,” IEEE Sel. Areas Commun., vol. 39, no. 8, pp. 2572–2589, Aug. 2021.
[22] M. N. Nguyen et al., “Self-organizing decentralized learning: Toward large-scale distributed learning systems,” IEEE Trans. Neural Netw. Learn. Syst., to be published, doi: 10.1109/TNNLS.2022.3170872.
[23] M. Duan et al., “FedGroup: Ternary cosine similarity-based clustered federated learning framework toward high accuracy in heterogeneous data,” 2020, arXiv:2010.06870.
[24] C. Brigo, Z. Fan, and P. Andras, “Federated learning with hierarchical clustering of local updates and sparse training on non-IID data,” in Proc. Int. Joint Conf. Neural Netw., 2020, pp. 1–9.
[25] S. Warnat-Herresthal et al., “Swarm learning for decentralized and confidential clinical machine learning,” Nature, vol. 594, no. 7862, pp. 265–270, 2021.
[26] I. Hegedüs, G. Danner, and M. Jelasity, “Decentralized learning works: An empirical comparison of gossip learning and federated learning,” J. Parallel Distrib. Comput., 2021, pp. 109041–124, 2021.
[27] A. Bellet, R. Guerraoui, M. Taziki, and M. Tommasi, “Personalized and private peer-to-peer machine learning,” in Proc. Int. Conf. Artif. Intell. Statist., 2018, pp. 473–481.
[28] C. Li, G. Li, and P. K. Varshney, “Decentralized federated learning via mutual knowledge transfer,” IEEE Internet of Things J., vol. 9, no. 2, pp. 1136–1147, Jan. 2022.
[29] A. Bellet, A. M. Kermarrec, and E. Lavoie, “D- Cliques: Compensating Non-IIDness in decentralized federated learning with topology,” 2021, arXiv:2104.07365.
[30] Z. Tang, S. Shi, and X. Chu, “Communication-efficient decentralized learning with sparsification and adaptive peer selection,” in Proc. IEEE 40th Int. Conf. Distrib. Comput. Syst., 2020, pp. 1207–1208.
[31] J. Jiang and L. Hu, “Decentralised federated learning with adaptive partial gradient aggregation,” CAAI Trans. Intell. Technol., vol. 5, no. 3, pp. 230–236, 2020.
[32] O. Marfoq, C. Xu, G. Neglia, and R. Vidal, “Throughput-optimal topology design for cross-silo federated learning,” in Proc. Int. Conf. Neural Inf. Process. Syst., 2020, pp. 19 478–19 487.
[33] N. Onoszko, G. Karlsson, O. Mogren, and E. L. Zec, “Decentralized federated learning of deep neural networks on non-IID data,” 2021, arXiv:2107.03997.
[34] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, “Deep domain confusion: Maximizing for domain invariance,” 2014, arXiv:1412.3474.
[35] M. Zhang, K. Sapra, S. Figler, S. Yeung, and J. M. Alvarez, “Personalized federated learning with first order model optimization,” in Proc. 9th Int. Conf. Learn. Representations, 2021.
Zexi Li received the BS degree in agriculture engineering from Zhejiang University, Hangzhou, China, in 2020, and the honorary degree from Chu Kochen Honors College, Zhejiang University. He is currently working toward the PhD degree in artificial intelligence with Zhejiang University. His main research interests include federated learning, optimization, and deep learning.

Jiaxun Lu (Member, IEEE) received the BS and PhD degrees from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2014 and 2019, respectively. From September 2017 to September 2018, he was a visiting student with Coordinate Science Laboratory, University of Illinois at Urbana-Champaign, Champaign, IL, USA. Currently, he is a senior researcher with Noah’s Ark Lab, Huawei Technologies. His research interests include statistical learning, information theory, and mobile wireless communication.

Shuang Luo received the BEng degree from the School of Data and Computer Science, Sun Yat-sen University, China, in 2019. She is currently working toward the PhD degree with the College of Public Affairs, Zhejiang University. Her current research interests include deep learning, reinforcement learning, and distributed artificial intelligence.

Didi Zhu received the BS degree in computer science and technology from the Beijing University of Chemical Technology, Beijing, China, in 2020. She is currently working toward the PhD degree with Zhejiang University. Her main research interests include federated learning and domain adaptation.