FLIS: Clustered Federated Learning Via Inference Similarity for Non-IID Data Distribution

MAHDI MORAFAH, SAEED VAHIDIAN, WEIJIA WANG, AND BILL LIN (Member, IEEE)
Department of Electrical and Computer Engineering, University of California San Diego, San Diego, CA 92161 USA
Corresponding Author: BILL LIN, E-mail: billlin@eng.ucsd.edu.

Abstract Conventional federated learning (FL) approaches are ineffective in scenarios where clients have significant differences in the distributions of their local data. The Non-IID data distribution in the client data causes a drift in the local model updates from the global optima, which significantly impacts the performance of the trained models. In this article, we present a new algorithm called FLIS that aims to address this problem by grouping clients into clusters that have jointly trainable data distributions. This is achieved by comparing the inference similarity of client models. Our proposed framework captures settings where different groups of users may have their own objectives (learning tasks), but by aggregating their data with others in the same cluster (same learning task), superior models can be derived via more efficient and personalized federated learning. We present experimental results to demonstrate the benefits of FLIS over the state-of-the-art approaches on the CIFAR-100/10, SVHN, and FMNIST datasets. Our code is available at https://github.com/MMorafah/FLIS.

Index Terms Clustering, data heterogeneity, federated learning, inference similarity, Non-IID data distribution, personalization.

I. INTRODUCTION

Federated learning (FL) is a recently proposed distributed training framework that enables distributed users to collaboratively train a shared model under the orchestration of a central server without compromising the data privacy of users [1]. Some FL approaches aim to learn a single common model for all clients [2], [3], [4], but these approaches lack flexibility and personalization, which leads to poor performance [5], [6], [7] in practical use cases in which there is statistical heterogeneity (Non-IIDness) in the distribution of the local data. Depending on the Non-IIDness, training a global shared model can be even less accurate than a local model trained just on the local data [8], [9], especially for clients that have enough private local data [7]. Alternatively, approaches have been proposed to derive personalized models that are based on either the fine-tuning of a global model via a few rounds of local training [5], [6] or interpolations between global and local models [10], [11]. However, fine-tuning approaches often fail to derive models that generalize well to local data distributions, and interpolation approaches often degenerate to clients training on its local data.

In this article, we propose a clustered federated learning algorithm where the clients are dynamically partitioned into different clusters based on their data distributions. Our goal is to group clients with similar data distributions into the same cluster without requiring access to their private data and then train models for every cluster of clients. The main idea of our algorithm is a strategy that alternates between estimating the cluster identities and maximizing the inference similarity on the server side. Our main contributions are as follows:

- We propose the idea of inference similarity as a way for the central server to identify clusters of clients that have similar data distributions without requiring any access to the private data of clients. This way, clients in the same cluster can benefit from each other’s training without the corruptive influence of clients with unrelated data distributions.
• Our algorithm can constitute joint and disjoint clusters and does not require the number of clusters to be known a priori. Further, it is effective both in Non-IID and IID regimes. In contrast, prior clustered FL approaches [12, 13] consider a pre-defined number of clusters (models) on the server and assign a hard membership ID to the clients. In such settings, these prior approaches could perform poorly under scenarios with highly skewed Non-IID data that would require a greater number of clusters, or scenarios with slightly skewed Non-IID data that would require fewer clusters, since we cannot know how many unique data distributions the client datasets are drawn from.

• Our proposed algorithm can also elegantly handle newcomer clients unseen at training time by matching them with a cluster model that the client can further personalize with local training.

• We perform extensive experimental studies to evaluate FLIS and verify its performance for FL under Non-IID data. In particular, we demonstrate that the proposed approach can significantly outperform the existing state-of-the-art (SOTA) global model FL benchmarks by up to ∼40%, and the SOTA personalized FL baselines by up to ∼30%.

II. RELATED WORK

Federated learning: FL was introduced in [2] as an efficient method for training models in a distributed fashion. Then, inspired by the FedAvg method [2], there have been some SOTA FL studies trying to develop modified versions of FedAvg algorithm under Non-IID data including FedProx [14], SCAFFOLD [3], and FedNova [4]. However, it has been shown that under Non-IID settings the performance of the global model drops around 51% on CIFAR-10 and 55% for keyword spotting (KWS) datasets [15] for neural networks trained for highly skewed Non-IID data. This accuracy reduction can be explained by the weight divergence across clients [15, 16]. In addition, a single global model lacks personalization, which may result in poor performance on local client test sets [7, 11, 15]. Therefore, in many practical applications, training a single global model for all clients under highly skewed non-IID data may not be optimal.

Personalized federated learning: A number of approaches have been proposed to address Non-IID distribution of data in FL from clients’ perspective [6, 7]. In particular, [6] proposed a representation learning method for FL by aggregating part of a model. This method requires clients to maintain local models across rounds, which is problematic when sampling clients from large populations. The authors in [7] proposed Sub-FedAvg (Hy) and Sub-FedAvg (Un) algorithms to realize personalization. They leverage pruning for finding a small similar subnetwork for clients with similar data distributions.

In addition, personalized FL under data heterogeneity was realized via performing clustering [12, 13]. Clustered-FL addresses this problem by grouping clients into separate clusters based on geometric properties of the FL loss surface [13]. [17] does clustering based on weights of models or model update comparisons at the server side. However, in FL, since the clients have limited amount of data, and they update their models for a few number of local epochs, clustering based on the model update would not be advisable.

The main advantages of our algorithm compared to the prior works is that our algorithm can constitute joint and disjoint clusters and does not require the number of clusters to be known a priori. Further, it is effective both in Non-IID and IID regimes. In contrast, [12] considers a pre-defined number of clusters (models) on the server. In such setting, the proposed method could perform poorly for many of the clients under both highly skewed Non-IID and slightly skewed Non-IID data.

III. FEDERATED LEARNING WITH CLUSTERING

A. PRELIMINARIES

FedAvg: In FL, several clients (end users) jointly train a single deep learning model on their own private data in a communication-efficient manner. In a traditional FL method, we sample $n$ clients out of $N$ to take part in each round of federation. Each client, $k$, has $m_k$ local data samples and communicating their learning regularly to a central server to reach global consensus about the whole data composed of $M = \sum m_k$. The federation is realized via an iterative three-step protocol where in each round of FL, the clients first synchronize with the server by downloading the latest global model, $\theta$. Every client then performs multiple iterations of stochastic gradient descent with mini-batches sampled from its local data to improve the downloaded model resulting in a weight update vector $\Delta \theta_{t+1}^k = SGD_k(\theta^t, D_i) - \theta^t$, $i = 1, \ldots, b$, where $D$ is the client data. Finally, the central server aggregates these local models into a new global model as $\theta^t+1 = \theta^t + \sum_{i=1}^{b} D_i \Delta \theta_{t+1}^k$.

Hierarchical clustering: When forming disjoint clusters where the number of clusters is not known in advance, hierarchical clustering (HC) [18] is of interest. Agglomerative HC is one of the well-known clustering techniques in machine learning that takes a proximity (adjacency) matrix as input and groups similar objects into clusters. HC starts by treating each client as a separate cluster. At each step of the clustering, the pairwise $L_2$ (Euclidean) distance between all clusters is computed to identify their similarity. The two clusters that are the closest ones are merged. This iterative process continues till all the clusters are merged into one. And finally, a distance threshold is defined to determine when to stop merging clusters. In our setting, the server applies hierarchical clustering on the adjacency matrix $A$ and forms $C$ disjoint clusters. In this article the distance threshold is called clustering threshold and is shown by $\beta$.

B. OVERVIEW OF FLIS ALGORITHM

In this section, we provide details of our algorithm. We name this algorithm Federated Learning by Inference Similarity (FLIS). FLIS is able to form both joint dynamic clusters with
FIGURE 1. A toy example showing the overview of FLIS algorithm. (a) The server sends the initial global model to the clients at Round 1. The clients update the received model using their local data and send back their updated models to the server. (b) The server captures the inference results on its own small dataset. Then according to the similarity of the inference results, the clients are clustered. In this example, clients \{1, 2, 3\} are yielding more similar inference results compared to Client 4. (c) The server uses inference similarity results to constitute the adjacency matrix and identify their cluster IDs via hard thresholding or hierarchical clustering and does model averaging within each cluster. In the next round, each new client selects the best cluster out of the ones that has been formed in the previous round.

Algorithm 1: The FLIS (DC) Framework.

Require: Number of available clients \(N\), sampling rate \(R \in [0, 1]\), data on the server \(D_{Server}\), clustering threshold \(\beta\).
Init: Initialize the server model with \(\theta_g^0\).

Def FLIS_DC:
1 for each round \(t = 0, 1, 2, \ldots\) do
2 \(n \leftarrow \max(R \times N, 1)\)
3 \(S_t \leftarrow \{k_1, \ldots, k_n\}\) random set of \(n\) clients
4 for each client \(k \in S_t\) in parallel do
5 if \(t = 0\) then
6 download \(\theta_g^0\) from the server and start training, i.e., \(\theta_{g,j_t}^0 = \theta_g^0\)
7 else
8 download clusters \(\theta_{g,j_t}^t, j_t = 1, \ldots, T_t\) from the server and select the best cluster according to \(\theta_{g,j_t}^t = \arg\min_j L_k(D^{test}; \theta_{g,j_t}^t)\)
9 \(\theta_{g,j_t}^{t+1} \leftarrow \text{ClientUpdate}(C_k; \theta_{g,j_t}^t)\) // SGD training
10 \(\{C_{j_t+1}\}_{j_t+1=1}^{T_{j_t+1}} = \text{ISC}(D_{Server}, \{\theta_{g,j_t}^{t+1}\}_{j=1}^{k=1,\ldots,K})\) // dynamically clustering clients via inference similarity
11 \(\theta_{g,j_t+1}^t = \sum_{k \in C_{j_t+1}} \|D_k^{ext}; \theta_{g,j_t}^{t+1}\| / \sum_{k \in C_{j_t+1}} |D_k|\)
12 end

Algorithm 2: Inference Similarity Clustering (ISC).

Require: Data on the server \(D_{Server}\), \(\beta\).
Return: The formed clusters \(\{C_j\}_{j=1}^{J}\).

Function ISC(D_{Server}, \{\theta_{g,j_t}^{t+1}\}_{j=1}^{k=1,\ldots,K}):
1 \(B_k = F_k(D_{Server}; \theta_{g,j_t}^{t+1})\) // \(F_k\) is client model
2 \(A_{i,j} = \|B_i \odot B_j\|_F\) // \(i,j = 1, \ldots, n\) // Server constructs the adjacency matrix
3 \(A_{i,j} = \Gamma(A_{i,j}) = \text{Sign}(A_{i,j} - \beta)\) // Server applies hard thresholding and does joint clustering
4 Return \(\{C_{j+1}\}_{j+1=1}^{T_{j+1}}\)

soft membership ID, named as FLIS (DC) and disjoint hierarchically formed clusters with hard membership ID, named as FLIS (HC). The overview of FLIS (DC) which forms joint clusters is sketched in Figure 1 and presented in Algorithms 1, and 2. The overview of FLIS (HC) which forms disjoint clusters is presented in Algorithm 3.

The first round of the algorithm starts with a random initial model parameters \(\theta_k\). In the t-th iteration of FLIS, the central server samples a random subset of clients \(S_t \subseteq [N]\) (\(N\) is the total number of clients), and broadcasts the current model parameters \(\{\theta_{g,j_t}^t\}_{j_t=1}^{T_t}\) to the clients in \(S_t\). We recall that the local objective \(L_k\) is typically defined by the empirical loss over local data. Each client then estimates its cluster identity via finding the model parameter that yields minimum loss on its test data, i.e., \(\theta_{g,j_t}^t = \arg\min_j L_k(D^{test}; \theta_{g,j_t}^t)\). Then the clients perform \(T_t\) steps of stochastic gradient descent (SGD) updates, get the updated model, and send their model parameters, \(\{\theta_{g,j_t}^{t+1}\}_{j_t=1}^{T_{j_t+1}}\), to the server. After receiving the model parameters from all the participating clients, the server then leverages inference similarity method to form clusters of clients that have similar data distributions. Finally, the server collects all the parameters from clients who are in the same cluster and averages the model parameters of each cluster.
Algorithm 3: The FLIS (HC) Framework.

Require: Number of available clients $N$, sampling rate $R \in (0, 1]$, data on the server $D_{server}$, clustering threshold $\beta$

Init: Initialize the server model with $\theta^0_g$

Def FLIS_HC:

1. for each round $t = 0, 1, 2, \ldots$ do
2.   if $t = 0$ then
3.     All clients receive the initial server model $\theta^0_g$, perform local update and back the updated models to the server.
4.     $A \leftarrow$ server forms $A$ based on $A_{i,j}$ defined in Subsection C.
5.     $\{C_1, \ldots, C_j\} = HC(A, \beta)$ // performing hierarchical clustering to obtain the clusters
6.     $\theta^0_{g,j} \leftarrow \theta^0_g$ // initializing all clusters with $\theta^0_g$
7.   else
8.     $n \leftarrow \max(R \times N, 1)$
9.     $S_r \leftarrow \{k_1, \ldots, k_n\}$ random set of $n$ clients
10.    for each client $k \in S_r$ in parallel do
11.       Each client $k$ receives its cluster model from the server $\theta^0_{g,k,j}$, $j = 1, \ldots, T$.
12.       $\theta^{t+1}_{g,k,j} \leftarrow$ ClientUpdate($C_k; \theta^t_{k,j}$) // SGD training
13.       $\theta^{t+1}_{g,k,j} = \sum_{k \in C_j} |D_k| \theta^{t+1}_{k,j} / \sum_{k \in C_j} |D_k|$
14.   end for

C. CLUSTERING CLIENTS BY INFERENCE SIMILARITY

Herein, we are aiming to find the clients with similar data distributions without requiring any prior knowledge about the data distributions. In doing so, we assume that the server has some auxiliary real or synthetic data\(^1\) on its own.\(^2\) The server then performs inference on each client model and obtains a $M \times N$ matrix, $B_k = F_k(D_{server}; \theta^0_{g,k,j})$, $k = 1, \ldots, |S_r|$, where $N$ and $M$ are the number of final neurons of the last fully connected layer (classification layer), and the number of auxiliary samples at the server, respectively. Note that, the columns of $B_k$ can be one-hot or soft labels. Using the inference results of each client, $B_k$, the server constructs an adjacency matrix $A_{i,j} = ||B_i \odot B_j||_F$, where $i, j = 1, \ldots, |S_r|$, and $\odot$ stands for Hadamard product. Having the adjacency matrix $A_{i,j}$, as mentioned earlier, depending on whether forming joint clusters are of interest or the disjoint ones, we propose two different clustering approaches. For FLIS (DC) that constructing joint clusters on the server is of interest, we define a hard thresholding operator $\Gamma$ which is applied on $A_{i,j}$ and yields $\hat{A}_{i,j} = \Gamma(A_{i,j}) = \text{Sign}(A_{i,j} - \beta)$, with $\beta$ being a threshold value. Now, making use of $\hat{A}_{i,j}$, the server can form joint clusters of interest by putting indices of the positive entries in each row of $\hat{A}_{i,j}$ in the same cluster as is shown in the toy example in Fig 1. In FLIS (DC), in each round, 10 clusters are formed, which is equal to the number of participant clients in each round. For FLIS (HC), having $A_{i,j}$ in hand, the server can group the clients by employing hierarchical clustering (HC)\(^3\) as presented in Algorithm 3. It is noteworthy that in FLIS (HC) the number of formed clusters are fixed and depends upon the clustering threshold ($\beta$) of HC which is a hyperparameter.

IV. EXPERIMENTS

We now present experimental results on the impact of the proposed algorithm, FLIS. We analyze FLIS experimentally from three aspects: (i) performance (accuracy) comparison of FLIS versus the SOTA baselines (ii) the effect of $\beta$ and number of local epochs on the performance of FLIS (iii) communication efficiency.

A. EXPERIMENTAL SETTINGS

Datasets and Models: We conduct experiments on CIFAR-10, CIFAR-100, SVHN, and Fashion MNIST (FMNIST) datasets. For each dataset we considered three different federated heterogeneity settings as in [26]: Non-IID label skew (20%), Non-IID label skew (30%), and Non-IID Dir($\alpha$). For Non-IID label skew (20%) and (30%) we first randomly assigned 20% and 30% of the total classes to each client, respectively. Then, each class is randomly and equally partitioned and distributed amongst the clients who own that particular class. For Non-IID Dir($\alpha$), we get random samples for class $c$ from Dirichlet distribution according to $p_c \sim \text{Dir}($$\alpha$). and give each client $j$ random samples of class $c$ according to $p_c, j$ proportion. In this setup, heterogeneity can be controlled by the parameter $\alpha$ of Dirichlet distribution. Specifically, when $\alpha \to \infty$ the clients’ data partitions become IID, and when $\alpha \to 0$ they become extremely Non-IID. We used Lenet-5 architecture for CIFAR-10, SVHN, and FMNIST datasets, and ResNet-9 architecture for CIFAR-100 dataset.

Baselines: To show the effectiveness of the proposed method, we compare the results of our algorithm against SOTA personalized FL methods as well as methods targeting to learn a single global model. Among the personalized SOTA ones, LG-FedAvg [6], learns local representations and a global head. Per-FedAvg [5] uses meta-learning technique to learn a single model that can adapt to the local dataset by a few steps of SGD. IFCA [12] groups the users into a predefined number of clusters and alternately optimizes the clusters and model parameters via gradient descent. Clustered-FL [13] dynamically groups the clients into clusters based on the cosine similarities of their trainable parameters and the centralized learning is performed on a per-class basis. For global FL methods, we compare versus SOTA benchmarks including FedAvg [2], FedProx [14], FedNova [4], and SCAFFOLD [3]. We also compare our results with another baseline named

\(^1\)Further, we find it noteworthy to mention that along the years this assumption for the FL scenarios has been adopted in the literature [15], [19], [20], [21], [22], [23], [24], [25].

\(^2\)The number of auxiliary samples used for forming the clusters at the server is 2500.

\(^3\)Further, we find it noteworthy to mention that along the years this assumption for the FL scenarios has been adopted in the literature [15], [19], [20], [21], [22], [23], [24], [25].
TABLE 1. Test Accuracy Comparison Across Different Datasets for Non-IID Label Skew (20%), and (30%)

| Algorithm | FMNIST | CIFAR-10 | CIFAR-100 | SVHN |
|-----------|--------|----------|-----------|------|
| SOLO      | 95.32 ± 0.57 | 79.22 ± 1.67 | 32.28 ± 0.23 | 79.72 ± 1.37 |
| FedAvg    | 77.3 ± 4.9   | 49.8 ± 3.3  | 53.73 ± 0.50 | 80.3 ± 0.8  |
| FedProx   | 74.9 ± 2.6   | 50.7 ± 1.7  | 54.35 ± 0.84 | 79.3 ± 0.9  |
| FedNova   | 70.4 ± 5.1   | 46.3 ± 3.5  | 53.61 ± 0.42 | 75.4 ± 4.8  |
| Scaffold  | 42.8 ± 28.7  | 49.3 ± 1.7  | 54.15 ± 0.42 | 62.7 ± 11.6 |
| LG        | 96.90 ± 0.51 | 86.31 ± 0.82 | 45.98 ± 0.34 | 92.61 ± 0.45 |
| PerFedAvg | 95.95 ± 1.15 | 85.46 ± 0.56 | 60.19 ± 0.15 | 93.32 ± 2.05 |
| IFCA      | 97.15 ± 0.01 | 87.99 ± 0.15 | 71.84 ± 0.23 | 95.42 ± 0.06 |
| CFL       | 77.33 ± 2.19 | 51.11 ± 1.01 | 40.29 ± 2.23 | 73.62 ± 1.76 |
| FLIS (DC) | 97.64 ± 0.38 | 89.47 ± 0.92 | 73.91 ± 0.29 | 95.65 ± 0.17 |
| FLIS (HC) | 97.45 ± 0.08 | 89.35 ± 0.46 | 73.20 ± 0.31 | 95.48 ± 0.21 |

TABLE 2. Test Accuracy Comparison for Non-IID Dir(0.1)

| Algorithm | FMNIST | CIFAR-10 | CIFAR-100 | SVHN |
|-----------|--------|----------|-----------|------|
| SOLO      | 93.33 ± 0.10 | 65 ± 0.65  | 22.95 ± 0.81 | 68.70 ± 3.13 |
| FedAvg    | 80.7 ± 1.9   | 58.3 ± 1.2  | 54.73 ± 0.41 | 82.0 ± 0.7  |
| FedProx   | 82.5 ± 1.9   | 57.1 ± 1.2  | 53.31 ± 0.48 | 82.1 ± 1.0  |
| FedNova   | 78.5 ± 3.0   | 54.4 ± 1.1  | 54.62 ± 0.91 | 80.5 ± 1.2  |
| Scaffold  | 77.7 ± 3.8   | 57.8 ± 1.4  | 54.90 ± 0.42 | 77.2 ± 2.0  |
| LG        | 94.21 ± 2.10 | 75.94 ± 0.16 | 56.91 ± 0.20 | 87.69 ± 0.77 |
| PerFedAvg | 92.87 ± 2.67 | 77.67 ± 0.19 | 56.42 ± 0.41 | 91.25 ± 1.47 |
| IFCA      | 95.22 ± 0.03 | 80.95 ± 0.29 | 67.39 ± 0.27 | 93.02 ± 0.15 |
| CFL       | 78.44 ± 0.23 | 52.57 ± 3.09 | 35.23 ± 2.23 | 73.97 ± 4.77 |
| FLIS (DC) | 95.95 ± 0.51 | 82.25 ± 1.12 | 68.36 ± 0.12 | 93.68 ± 0.22 |
| FLIS (HC) | 95.35 ± 0.16 | 82.17 ± 0.22 | 67.51 ± 0.23 | 93.10 ± 0.20 |

SOLO, where each client trains a model on its own local data without taking part in FL. Our code is available at https://github.com/MMorafah/FLIS.

Performance Comparison: Tables 1 and 2, show the average final top-1 test accuracy of all clients for all the SOTA algorithms under Non-IID label skew (20%, 30%), and Non-IID Dir(0.1) settings, respectively. In these tables we report the results of the two proposed clustering approaches i.e., FLIS (DC) (presented in Algorithm 1) as well as FLIS (HC) (presented in Algorithm 3). Under Non-IID settings, SOLO with zero communications cost demonstrates much better accuracy than all the global FL baselines including FedAvg, Fedprox, FedNova, and SCAFFOLD. On the other hand, each client itself may not have enough data and thus we need to better exploit the similarity among the users by clustering. This further explains the benefits of personalization and clustering in Non-IID settings. Comparing different FL approaches, we can see that FLIS (DC) consistently yields the best accuracy results among all tasks. It can outperform FedAvg by up to ~40%.

The accuracy of FedProx, is very close to FedAvg. This shows that the regularizer term in FedProx is not that beneficial in the federated training for small values of the regularizer’s coefficient. For large values of the regularizer’s coefficient, the convergence of FedProx is very slow and the accuracy results are poor. This is consistent with the observations in [16]. Unlike the paper introducing LG-FedAvg, in which the initialization of the algorithm is the solution of many rounds of FedAvg, for the sake of fair comparison, we initialized the model randomly and reported the average final local test accuracy over the entire federated training process rather than the average of the maximum local test accuracies for each client.

It is apparent from Table 2 for Non-IID Dir(0.1) that LG-FedAvg and Per-FedAvg perform even worse than FedAvg. The performance of CFL benchmark is close to that of FedAvg in most cases, and even worse. IFCA (with two clusters, C=2) obtained the closest results to FLIS, but FLIS consistently beats IFCA especially in Non-IID Dir(0.1) by a large margin. FLIS shows superior learning performance over the SOTA on more challenging tasks. For instance, FLIS, is noticeably better than IFCA for CIFAR-10 which is a harder task compared to FMNIST and SVHN by up to ~10% in Non-IID Dir(0.1).

As a final note, we also studied the impact of constructing disjoint clusters. HC by extracting disjoint clusters, seems to be slightly deteriorating the performance of FLIS, even though it still beats all SOTA baselines.

B. COMMUNICATION EFFICIENCY

What is the Required Communication Cost/Round to Reach a Target Test Accuracy? We additionally compare the SOTA baselines in terms of the number of communication rounds/cost that is required to reach a specific target accuracy. Table 3 report the required number of communication rounds to reach the designated target test accuracies for Non-IID label skew (20%). Further, Tables 4 and 5 report the required communication cost to reach the designated target test accuracies for Non-IID label skew (30%), and Non-IID Dir (0.1), respectively. As is observed from the Table 3, in all scenarios, FLIS has the minimum number of communication rounds, except for LG. As a final note, we also studied the impact of constructing disjoint clusters. HC by extracting disjoint clusters, seems to be slightly deteriorating the performance of FLIS, even though it still beats all SOTA baselines.
We attribute this to the fact that by grouping the clients with similar data distributions in the same clusters, the clusters tend to mimic the IID setting, which means faster convergence in fewer communication rounds. Note that “−−” means the baseline was not able to reach the target accuracy. This characteristic of FLIS (HC) is desirable in practice as it helps to reduce the communication overhead in FL systems in two ways: first, it converges fast and second, rather than communicating all clusters (models) with the clients, the server will receive the cluster ID from each client and then only send the corresponding cluster to each client.

### C. LEARNING WITH LIMITED COMMUNICATION

We further consider circumstances that frequently happen in practice, where a limited budget of communication rounds is allowed for federation under a heterogeneous setting. Herein, we compare the performance of FLIS with the rest of SOTA. We allocate limited communication rounds budget of 80 for all personalized baselines and report the average final test accuracy over all clients versus number of communication rounds for Non-IID label skew (20%), and (30%) in TABLE 4. Comparing Different FL Approaches for Non-IID Label Skew (30%) in Terms of the Required Communication Cost in Mb to Reach the Target top-1 Average Local Test Accuracy. We Evaluate on FMNIST, CIFAR-10, CIFAR-100, and SVHN.

| Algorithm  | FMNIST | CIFAR-10 | CIFAR-100 | SVHN |
|------------|--------|----------|-----------|------|
| Target     | 80%    | 70%      | 50%       | 80%  |
| FedAvg     | 79.36  | --       | 4237.37   | 71.43|
| FedProx    | 71.43  | --       | 4237.37   | 71.43|
| FedNova    | --     | --       | 3601.98   | 79.36|
| Scaffold   | --     | --       | 3305.11   | --   |
| LG         | 1.26   | 2.11     | --        | 1.76 |
| PerFedAvg  | 7.54   | 23.81    | 6356.06   | 18.65|
| IFCA       | 11.30  | 16.66    | 3495.19   | 10.71|
| CFL        | --     | --       | --        | --   |
| **FLIS (HC)** | **7.53** | **10.31** | **1991.60** | **8.73** |

Figs. 2 and 3, respectively. We can see that our proposed method requires only 30 communication rounds to converge in CIFAR-10, SVHN, and FMNIST datasets. CFL yields the worst performance on all benchmarks across all datasets, except for CIFAR-100. Per-Fedavg seems to benefit more from higher communication rounds. IFCA, and LG are the closest lines to ours for CIFAR-10, SVHN and FMNIST. FLIS consistently outperforms the SOTA in different communication rounds.

### D. GENERALIZATION TO UNSEEN CLIENTS

As mentioned earlier, FLIS allows new clients arriving after the distributed training to learn their personalized models. In our framework, the unseen client simply provides its model which trained for some epochs on its own data to the server. The server then obtains the inference results of the unseen client model and extends the adjacency matrix and applies HC. Therefore, the server via inference similarity analysis identifies which existing cluster model would be most suitable, or the server can inform the client that it should train on its own local data to form a new cluster if the client’s data distribution is not sufficiently similar to the distributions of the existing clusters. It is not clear how the other personalized FL algorithms should be extended to handle unseen clients during federation. In order to evaluate the performance of new clients’ personalized models, we run an experiment where only 80% of the clients participate to the training. The remaining 20% join the network at the end of the federation and receive the model from the server and personalize it for only 5 epochs. The average local test accuracy of the new clients is reported in Table 6. Table 6 demonstrates that FLIS allows unseen clients during the training to learn their personalized model with high test accuracy.

### E. IMPACT OF IMPORTANT HYPER-PARAMETERS

Herein, we study the impact of a few important hyper-parameters on the performance of FLIS as in the following.

The influence of the clustering threshold $\beta$: We investigate the effect of the clustering threshold $\beta$ on the final test
accuracy. Figs. 4 and 5 visualize the accuracy performance behavior of FLIS under different values of β, as well as the local epochs for several datasets for Non-IID (20%) and Non-IID (30%), respectively. We vary β from 0 to 1. The parameter β controls the similarity of the data distribution of clients within a cluster. Therefore, β achieves a trade-off between a purely local and global model and provides a trade-off between generalization and distribution heterogeneity. To delineate, when β = 0, FLIS groups all the clients into 1 cluster and the scenario reduces to FedAvg baseline. This is the reason for the significant accuracy drop at β = 0 as it is also evident from these figures, by increasing β, FLIS becomes more strict in grouping the clients. It means FLIS only groups the clients with more amount of label/feature overlap into a cluster leading to a more personalized FL. The optimal performance on CIFAR-10, SVHN, and FMNIST for Non-IID (20%) are achieved at β = 0.3, 0.3, 0.5, respectively. Finally, when β is 1, the scenario degenerates to SOLO baseline where each client receives the model from the server and trains it on its own local data without averaging with anyone. According to our experiments, on all datasets, all clients benefit from some level of globalization. In general, finding the optimal

| Algorithm | FMNIST | CIFAR-10 | CIFAR-100 | SVHN |
|-----------|--------|----------|-----------|------|
| SOLO      | 95.13 ± 0.42 | 82.30 ± 1.00 | 27.26 ± 0.98 | 91.5 ± 0.64 |
| FedAvg    | 77.61 ± 3.78 | 31.01 ± 1.83 | 32.19 ± 0.32 | 71.78 ± 3.43 |
| FedProx   | 74.30 ± 4.70 | 27.56 ± 3.24 | 32.41 ± 1.17 | 74.30 ± 4.70 |
| FedNova   | 74.66 ± 2.81 | 31.48 ± 1.49 | 33.18 ± 0.80 | 73.04 ± 3.65 |
| Scaffold  | 73.57 ± 1.68 | 37.22 ± 1.34 | 23.90 ± 2.61 | 64.96 ± 4.74 |
| LG        | 94.58 ± 0.33 | 77.98 ± 1.61 | 10.63 ± 0.21 | 89.48 ± 0.65 |
| PerFedAvg | 89.88 ± 0.38 | 73.79 ± 0.51 | 30.09 ± 0.35 | 67.48 ± 2.88 |
| FLIS (DC) | 97.50 ± 1.30 | 94.45 ± 1.76 | 58.35 ± 1.46 | 94.87 ± 0.34 |
| FLIS (HC) | 96.30 ± 0.20 | 85.34 ± 0.11 | 59.11 ± 0.52 | 95.11 ± 0.18 |

FIGURE 4. Evaluating FLIS (DC)’s accuracy performance versus the clustering threshold β, and number of local epochs for Non-IID label skew (20%) on CIFAR-10, FMNIST, and SVHN datasets. FLIS (DC) benefits from larger numbers of local training epochs. The optimal values of β are different for different datasets.

FIGURE 3. Test accuracy versus number of communication rounds for Non-IID (30%). FLIS converges fast to the desired accuracy and consistently outperforms strong competitors, except in SVHN.
Evaluating FLIS (DC)'s accuracy performance versus the clustering threshold $\beta$, and number of local epochs for Non-IID label skew (30%) on CIFAR-10, SVHN, and FMNIST datasets. FLIS (DC) benefits from larger numbers of local training epochs. The optimal values of $\beta$ are different for different datasets.

The trade-off between globalizaton and distribution heterogeneity depends on the level of heterogeneity of tasks, the intra-class distance of the dataset, as well as the data partitioning across the clients. This is precisely what our novel FLIS (HC) is designed to do, to easily find this trade-off via the proximity matrix at the server.

Benefit of more local updates: The benefits of FLIS can be further pronounced by increasing the number of local epochs. The results are shown in Figs. 4 and 5. When the number of local epoch is 1, the clients' local updates are very small. Therefore, the training will be slow and the accuracy becomes lower compared to the bigger number of local epochs given a fixed number of communication rounds. Also, when the clients have not been trained enough, their inference results at server side would be erroneous which further causes less accurate clustering. For Non-IID (30%) as shown in Fig. 5 the results when the number of local epoch are 10, and 20 are comparable but still are better that that of 1 local epoch. These figures show the performance of FLIS is coupled with local training epochs specially on more challenging tasks. In contrast, it was shown in [26] when the number of local epochs is too large, the accuracy of all non-personalized models drop which is due to severe-side averaged models drift from the clients’ local models [4].

F. THE INFLUENCE OF $\beta$ ON CLUSTERING ERROR

We investigate the effect of the clustering threshold $\beta$ on the clustering error. Figs. 6 and 7 visualize the clustering error behaviour of FLIS versus clustering threshold $\beta$ and number of local epochs for Non-IID (20%), and Non-IID (30%) on CIFAR-10, SVHN, and FMNIST datasets, respectively. We define clustering error as the summation of false positives (FP) and false negatives (FN) w.r.t. the ground-truth. Depending on the dataset, at some optimal $\beta$ we should expect minimum clustering error. For Non-IID (20%) on CIFAR-10, SVHN, and FMNIST the minimum clustering error occurs at $\beta = 0.1$, $\beta = 0.3$, and $\beta = 0.5$ which is reflected in a shorter error bar. When $\beta$ is less than the optimal one, a large FP causes bigger error and when $\beta$ is bigger than the optimal value, a large FN is the reason of bigger error. Another noticeable observation is that the accuracy peak of SVHN and FMNIST in Fig. 4 and their corresponding minimum clustering error in Fig. 6 occur at the same $\beta$. While this is not the case for CIFAR-10. Indeed, it is not required that these $\beta$ values match. This can be explained by the fact that, FLIS groups the clients based on the inference similarity/response of neural networks. This means FLIS selects a subset of the most similar clients out of the set of similar clients. This way, FLIS sacrifices some clustering error by accepting more FN in order to improve the accuracy. Similar discussions can be made for Fig. 7.

G. THE IMPACT OF VARIOUS SETS OF AUXILIARY DATA ON THE INFERENCE SIMILARITY MEASURE

To better understand the inference behavior of clients which is leveraged by the server to group the clients with similar distributions in a Non-IID setting, we did some ablation studies. It is worth noting that, according to our experiments we do not necessarily need an auxiliary dataset on the server-side that is drawn from the same dataset used to train or test the clients, nor do we need an auxiliary dataset on the server-side that is representative of all the clients involved. These observations, while not intuitive at first, are not surprising given the well-established results in the pseudo-labeling and unsupervised domain adaptation literature [27], [28], [29], [30], [31], [32] that has already shown the effective use of labeled data from another source domain to evaluate neural nets intended for a different target domain. For example, in our experiments with CIFAR-10 as the target domain, we can just use samples drawn from another source domain like SVHN as the auxiliary dataset to perform inference similarity analysis on the server-side, and the results are just as effective as using samples drawn from the same CIFAR-10 dataset. We further conducted experiments with just using randomly generated images as the auxiliary dataset, and the results are also just as effective. We observe the same phenomenon when training for SVHN, but using another domain like CIFAR-10 or randomly generated images as the auxiliary dataset, where the results achieved are similar.

Figs. 8, 9, and 10 sketch the ablation study of inference results of clients with similar distribution (label overlap) and dissimilar distribution (without label overlap) for different numbers of local epochs, and $\alpha$ (the Dirichlet distribution parameter). Herein, every client data is drawn independently with class labels following a categorial distribution over $P$ classes parameterized by a vector $v(v_i \leq 0; i \in [1, P])$. The benefits of FLIS can be further pronounced by increasing the number of local epochs. The results are shown in Figs. 4 and 5. When the number of local epoch is 1, the clients’ local updates are very small. Therefore, the training will be slow and the accuracy becomes lower compared to the bigger number of local epochs given a fixed number of communication rounds. Also, when the clients have not been trained enough, their inference results at server side would be erroneous which further causes less accurate clustering. For Non-IID (30%) as shown in Fig. 5 the results when the number of local epoch are 10, and 20 are comparable but still are better that that of 1 local epoch. These figures show the performance of FLIS is coupled with local training epochs specially on more challenging tasks. In contrast, it was shown in [26] when the number of local epochs is too large, the accuracy of all non-personalized models drop which is due to severe-side averaged models drift from the clients’ local models [4].
FIGURE 6. Evaluating the clustering error behavior of FLIS (DC) for Non-IID (20%) versus the clustering threshold $\beta$, and number of local epoch on Left: CIFAR-10, Middle: SVHN, and Right: FMNIST datasets.

FIGURE 7. Evaluating the clustering error behavior of FLIS (DC) for Non-IID (30%) versus the clustering threshold $\beta$, and number of local epoch on Left: CIFAR-10, Middle: SVHN, and Right: FMNIST datasets.

FIGURE 8. Understanding the impact of similarity and heterogeneity of the clients' data on their inference result at the server side for different degrees of Non-IIDness controlled by $\alpha$, and number of local epochs per communication round on CIFAR-10 (Left) and on SVHN (Right). 20 clients out of 100 are randomly selected and trained for certain number of epochs (1, 5, 10, and 20). The figures visualize the adjacency matrices obtained based on the inference results of some auxiliary data sampled from CIFAR-10 (Left) and that of some auxiliary data sampled from SVHN (Right) as outlined in line 3 of Algorithm 2.
We use $\alpha$ to control the Non-IID degree of the local data. The smaller $\alpha$, the larger heterogeneity between local data distributions of clients. Big values of $\alpha$, e.g., $\alpha = 100$ mimics identical local data distributions (IID). We visualize the inference similarity result of the clients on CIFAR-10, SVHN, and FMNIST, as well as on some synthetic randomly generated data. We sketch the adjacency matrix for the extreme cases of IID-ness and Non-IID-ness with $\alpha = \{100, 0.1, 0.01\}$. In the figures, we move from extremely Non-IID with $\alpha = 0.01$ to extremely IID with $\alpha = 100$. In all cases, the entries on the main diagonal are all 1. In the extremely Non-IID case, there should be more difference between diagonal and off-diagonal entries which is reflected in the figures. As expected, as we move to extremely IID case, the difference between the diagonal and off-diagonal should decrease. That is because the clients have similar distribution and should produce similar inference results. Therefore, inference similarity can be a metric to measure the IID-ness and Non-IID-ness of the distributions across clients by looking at the difference between diagonal and off-diagonal entries of the adjacency matrix.

For producing these figures, 20 clients out of 100 are randomly selected and trained for certain number of epochs (1, 5, 10, and 20). The figures visualize the adjacency matrices obtained based on the inference results of some auxiliary data sampled from CIFAR-10, FMNIST, SVHN, and synthetic...
randomly generated data (vectors) as outlined in line 3 of Algorithm 2. As can be seen from the figures, regardless of the sort of the auxiliary data used at the server, when the clients’ data distributions are similar (Dir (100)), their inference results are also similar and vice versa.

V. CONCLUSION

In this article, we studied the FL problem in the context of clients with limited amounts of Non-IID data. We proposed the idea of inference similarity as a way for the server to construct clusters of clients that have similar data distributions without requiring any access to sensitive on-device data. The server can evaluate inference similarity on the locally updated models that it receives from the participating clients in each round. As a result, clients in the same cluster can benefit from each other’s training task without the corruptive influence of clients with unrelated data distributions. Our experimental evaluations show that our proposed method substantially outperforms the SOTA while significantly reducing the number of communication rounds.

REFERENCES

[1] B. McMahan and D. Ramage, “Federated learning: Collaborative machine learning without centralized training data,” Google Res. Blog, vol. 3, 2017.
[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in Proc. Artif. Intell. Statist., 2017, pp. 1273–1282.
[3] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, “SCAFFOLD: Stochastic controlled averaging for federated learning,” in Proc. 37th Int. Conf. Mach. Learn., 2020, pp. 5132–5143.
[4] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, “Tackling the objective inconsistency problem in heterogeneous federated optimization,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 7611–7623.
[5] A. Fallah, A. Mokhtari, and A. Ozdaglar, “Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 3557–3568.
[6] P. P. Liang, T. Liu, L. Ziyin, R. Salakhutdinov, and L.-P. Morency, “Think locally, act globally: Federated learning with local and global models,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 27–34.
[7] F. Hanzely and P. Richtárik, “Federated learning of a mixture of global and local models,” 2020, arXiv:2002.05516.
[8] T. Yu, E. Bagdasaryan, and V. Shmatikov, “Salvaging federated learning by local adaptation,” 2020, arXiv:2002.04758.
[9] Y. Deng, M. M. Kamani, and M. Mahdavi, “Adaptive personalized federated learning,” 2020, arXiv:2003.13461.
[10] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, “Three approaches for personalization with applications to federated learning,” 2020, arXiv:2002.10619.
[11] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, “An efficient framework for clustered federated learning,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 19586–19597.
[12] 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.
[13] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “Federated optimization in heterogeneous networks,” in Proc. Mach. Learn. Syst., 2020, pp. 429–450.
[14] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, “Federated learning with non-IID data,” 2018, arXiv:1806.00582.
[15] Q. Li, B. He, and D. Song, “Model-contrastive federated learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 10713–10722.
[16] C. Briggs, Y. Fan, and P. Andras, “Federated learning with hierarchical clustering of local updates to improve training on non-IID data,” in Proc. IEEE Int. Joint Conf. Neural Netw., 2020, pp. 1–9.
[17] W. H. Day and H. Edelsbrunner, “Efficient algorithms for agglomerative hierarchical clustering methods,” J. Classification, vol. 1, no. 1, pp. 7–24, 1984.
[18] D. Li and J. Wang, “FEDMD: Heterogeneous federated learning via model distillation,” 2019, arXiv:1910.03581.
[19] H. Chang, V. Shejwalkar, R. Shokri, and A. Housmansadr, “CRONUS: Robust and heterogeneous collaborative learning with black-box knowledge transfer,” 2019, arXiv:1912.11279.
[20] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, “Ensemble distillation for robust model fusion in federated learning,” in Proc. Adv. Neural Inf. Process. Syst. Annu. Conf. Neural Inf. Process. Syst., 2020, pp. 2351–2363.
[21] S. Itahara, T. Nishio, Y. Koda, M. Morikura, and K. Yamamoto, “Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-IID private data,” IEEE Trans. Mobile Comput., vol. 22, no. 1, pp. 191–205, 2023.
[22] C. He, M. Annavaram, and S. Avetisian, “Group knowledge transfer: Federated learning of large CNNs at the edge,” in Proc. Adv. Neural Inf. Process. Syst. Annu. Conf. Neural Inf. Process. Syst., 2020, pp. 14068–14080.
[23] Y. Ge, D. Chen, and H. Li, “Mutual mean-teaching: Pseudo label refinement for unsupervised domain adaptation on person re-identification,” in Proc. 8th Int. Conf. Learn. Represent., 2020.
[24] K. Sohn et al., “Fixmatch: Simplifying semi-supervised learning with inter-client consistency & disjoint learning,” in Proc. 9th Int. Conf. Learn. Representations, 2021.
[25] Q. Li, Y. Diao, Q. Chen, and B. He, “Federated learning on non-IID data silos: An experimental study,” IEEE 38th Int. Conf. Data Eng., pp. 965–978, 2022.
[26] D. Hyun Lee, “Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks,” in Workshop Challenges Represent. Learn., vol. 3, no. 2, p. 896, 2013.
[27] Y. Ge, D. Chen, and H. Li, “Communication-efficient collaborative training with non-IID private data,” in Proc. 9th Int. Conf. Learn. Representations, 2020.
[28] K. Sohn et al., “Fixmatch: Simplifying semi-supervised learning with consistency and confidence,” in Proc. Adv. Neural Inf. Process. Syst. Annu. Conf. Neural Inf. Process. Syst., 2020, pp. 6256–6268.
[29] D. Hyun Lee, “Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks,” in Workshop Challenges Represent. Learn., vol. 3, no. 2, p. 896, 2013.
[30] X. Dai, E. H. Hovy, T. Luong, and Q. Le, “Unsupervised data augmentation for consistency training,” in Proc. Adv. Neural Inf. Process. Syst. Annu. Conf. Neural Inf. Process. Syst., 2020, pp. 6256–6268.
[31] K. Sohn et al., “Fixmatch: Simplifying semi-supervised learning with consistency and confidence,” in Proc. Adv. Neural Inf. Process. Syst. Annu. Conf. Neural Inf. Process. Syst., 2020, pp. 596–608.
[32] J. Yang, S. Shi, Z. Wang, H. Li, and X. Qi, “ST3D: Self-training for unsupervised domain adaptation on 3D object detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2022, pp. 10368–10378.

MAHDI MORAFAH received the B.S. degree in electrical engineering from the Tehran Polytechnic University, Tehran, Iran, in 2019, and the M.S. degree in electrical and computer engineering from the University of California, San Diego, La Jolla, CA, USA, in 2021, where he is currently working toward the Ph.D. degree in electrical and computer engineering. His research interests include machine learning and deep learning, distributed learning, federated learning, continual learning, and optimization.
SAEED VAHIDIAN was born in Ghochan, Iran. He received the B.Sc. degree in electrical and computer engineering from the Ferdowsi University of Mashhad, Mashhad, Iran, in 2012, and the M.Sc. degree in electrical and computer engineering from the K. N. Toosi University of Technology, Tehran, Iran, in 2014. He is currently working toward the Ph.D. degree in electrical and computer engineering with the University of California San Diego, La Jolla, CA, USA. Between 2015 and 2016, he was an Optimization Specialist with Huawei Company and Ericsson Company, Tehran. In Summer 2021, he was an Intern with Qualcomm AI. His research interests include federated learning, meta learning, continual learning, column subset selection, and convex and non-convex optimization.

WEIJIA WANG received the B.S. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2016, and the M.S. degree in electrical and computer engineering from the University of California, San Diego, La Jolla, CA, USA, in 2018, where he is currently working toward the Ph.D. degree in electrical and computer engineering. His research interest include machine learning and deep learning, including the compression and acceleration of deep convolutional neural networks, algorithms of meta-learning and federated learning, and explainable artificial intelligence.

BILL LIN (Member, IEEE) received the B.S., M.S., and Ph.D. degrees, all in electrical engineering and computer sciences from the University of California, Berkeley, Berkeley, CA, USA, in 1985, 1988, and 1991, respectively. He is currently a Professor of electrical and computer engineering with the University of California, San Diego, La Jolla, CA, where he is actively involved with the Center for Wireless Communications, Center for Networked Systems, and Qualcomm Institute in industry-sponsored research efforts. His research has led to more than 200 journal and conference publications, including a number of Best Paper awards and nominations. He was the recipient of the five patents. He was the General Chair and served on the executive and technical program committee of many IEEE and ACM conferences, and was an Associate or Guest Editor for several IEEE and ACM journals.