Federated Learning With Taskonomy for Non-IID Data

Jamali-Rad, Hadi; Abdizadeh, Mohammad; Sing, Anuj

DOI
10.1109/TNNLS.2022.3152581

Publication date
2022

Document Version
Accepted author manuscript

Published in
IEEE Transactions on Neural Networks and Learning Systems

Federated Learning With Taskonomy for Non-IID Data

Hadi Jamali-Rad, *Senior Member, IEEE*, Mohammad Abdizadeh, and Anuj Singh

***Abstract*— Classical federated learning approaches incur significant performance degradation in the presence of non-independent and identically distributed (non-IID) client data. A possible direction to address this issue is forming clusters of clients with roughly IID data. Most solutions following this direction are iterative and relatively slow, also prone to convergence issues in discovering underlying cluster formations. We introduce federated learning with taskonomy (`FLT`) that generalizes this direction by learning the task relatedness between clients for more efficient federated aggregation of heterogeneous data. In a one-off process, the server provides the clients with a pretrained (and fine-tunable) encoder to compress their data into a latent representation and transmit the signature of their data back to the server. The server then learns the task relatedness among clients via manifold learning and performs a generalization of federated averaging. `FLT` can flexibly handle a generic client relatedness graph, when there are no explicit clusters of clients, as well as efficiently decompose it into (disjoint) clusters for clustered federated learning. We demonstrate that `FLT` not only outperforms the existing state-of-the-art baselines in non-IID scenarios but also offers improved fairness across clients. Our codebase can be found at: https://github.com/hjraad/FLT/**

***Index Terms*— Federated learning, non-independent and identically distributed (non-IID) client data.**

I. Introduction

FEDERATED learning is a new paradigm that offers significant potential in elevating edge-computing capabilities in modern massive distributed networks. While presenting great potential, federated learning also comes with its own unique challenges in practical settings [1], [2]. Recent studies focus on systemic heterogeneity [3], communication efficiency [2], [4], [5], privacy concerns [6], [7], and more recently on fairness [8], [9] and robustness across the network of clients [10], [11]. A defining characteristic of massive decentralized networks is stochastic heterogeneity of client data, i.e., clients possess non-independent and identically distributed (non-IID) data. Non-IIDness in client data can be due to several factors, including the following two most commonly considered aspects: 1) pathological non-IIDness, where different clients can see different target classes, and 2) quantity skew, where different clients can have imbalanced number of samples to train on [4]. Li *et al.* [12] identifies statistical heterogeneity as the root cause for tension between fairness and robustness constraints in federated optimization. McMahan *et al.* [4], Li *et al.* [13] investigate the impact of heterogeneous data distributions on the performance of federated averaging algorithm, `FedAvg` [4], and demonstrate significant performance degradation in non-IID settings. Several avenues have been explored in the literature to tackle the problem of statistical heterogeneity in federated learning settings. Personalized federated learning tackles data heterogeneity by forming personalized models for clients via meta-learning or multi-task learning [12], [14]–[17]. Clustered federated learning addresses this problem by iterative (or recursive) assignment of clients to separate clusters based on model or model update comparisons at the server side [18]–[22]. The effectiveness of clustering approaches hinges upon the quality of cluster formation through this iterative assignment process. Besides, clustered federated learning approaches are sensitive to initialization, as we will demonstrate later on. More details on the related work will be provided in the next section.

Inspired by the idea of "taskonomy" [23], we explore the task relatedness across client data distributions and cast it in the form of a client relatedness graph. This is accomplished in a *one-off* fashion based on contractive encoding of client data followed by manifold learning (with `UMAP`) at the server side. The proposed approach (`FLT`) can flexibly handle a range of possibilities in incorporating this client relatedness graph for federated averaging. It can be used in generic form as an extension of `FedAvg` for non-IID data when there are no explicit clusters of clients with similar data distributions or, if need be, can be decomposed with hierarchical clustering (`HC`) to disjoint clusters and transform into clustered federated learning. Our main contributions can be summarized as follows: 1) we propose federated learning with taskonomy (`FLT`), which learns the task relatedness among clients and uses it at the server side for federated averaging of non-IID data, without requiring any prior knowledge about data distribution correlations among clients or the number of clusters they belong to; 2) `FLT` can flexibly discover generic client relatedness and an underlying clustered formation in non-IID scenarios; 3) we empirically show that `FLT` offers

Manuscript received May 14, 2021; revised October 8, 2021 and January 2, 2022; accepted February 15, 2022. An earlier version of this paper is presented (and has won the Best Paper Award) at ICLR 2021 Workshop on Distributed and Private Machine Learning (DPML). *(Corresponding author: Hadi Jamali-Rad.)*

Hadi Jamali-Rad is with Shell Global Solutions International B.V., 1031HW Amsterdam, The Netherlands, and also with the Department of Intelligent Systems (INSY), Delft University of Technology (TU Delft), 2628 CD Delft, The Netherlands (e-mail: h.jamalirad@tudelft.nl).

Mohammad Abdizadeh is with Myant Inc, Toronto, ON M9W 1B6, Canada (e-mail: mohammad.abdizadeh@myant.ca).

Anuj Singh is with the Department of Intelligent Systems (INSY), Delft University of Technology (TU Delft), 2628 CD Delft, The Netherlands (e-mail: a.r.singh@student.tudelft.nl).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TNNLS.2022.3152581.

Digital Object Identifier 10.1109/TNNLS.2022.3152581

faster convergence compared with the existing state-of-the-art baselines; 4) we provide convergence guarantees on FLT under common assumptions required for convergence of FedAvg; 5) we demonstrate that FLT outperforms the existing state-of-the-art baselines across several (synthetic and realistic) federated learning settings by about 3%, 9%, 2%, and 40%, on MNIST, CIFAR10, FEMNIST, and a newly introduced "Structured Non-IID FEMNIST" dataset); and 6) finally, we show that FLT offers improved fairness (least variance in performance across clients), besides the improved accuracy.

II. Related Work

A. Federated Learning of Non-IID Data

An early proposition to handle non-IID data in federated learning was to create a globally shared dataset comprising a small subset of data from each client [24]. However, the validity of this approach has been debated from increased communication costs, privacy, and security perspectives [3]. Li *et al.* [13] proposed to add a proximal term to local optimization sub-problems to limit the impact of "variable" local updates. This problem is phrased as "client drift" in [25], and a set of control variates are communicated between the clients and server to correct for this drift. In a somewhat related context but on different type of data (such as medical), other studies propose changing the global optimization cost function or weighting the aggregation at the server side to account for dissimilarities among data distributions [26]–[28].

B. Personalized and Multi-Task Federated Learning

This line of research tackles data heterogeneity by forming personalized models for clients via meta-learning or multi-task learning [14]–[16]. The work presented in [14] extends multi-task learning to federated learning; however, it relies on alternating bi-convex optimization which limits its applicability to only convex objective functions. In meta-learning setting [15], [16], first a single global model is obtained at the server, based on which each client fine-tunes its model. However, this global model would not serve as a good initialization if client data distributions are considerably different.

C. Clustered Federated Learning

Akin to our high-level approach, two iterative approaches are proposed in [18] and [19] to assign clients to separate clusters and train sub-models per cluster. However, several rounds of communication are required until the formation of clusters is solidified. More specifically, in [18] per iteration the cluster sub-models have to be sent to all the other active clients in that iterations, which is demanding in terms of communication cost. The clustering methodologies proposed in [20], [21], and [29] are based on recursive bi-partitioning with cosine similarity between model updates as metric. Owing to the recursive nature, their compute and communication overhead can become a bottleneck in large-scale settings. A multicenter federated learning approach is proposed in [22] where clients are clustered based on their model parameter differences with randomly initialized cluster-level models. We argue that model parameters are just an implicit proxy of client data distributions, whereas our approach directly exploits the client data itself and distills it into its signature encoding for clustering. Notably, the approach of [22] is sensitive to model initialization and also requires prior knowledge about the number of clusters. The cited approaches are mostly iterative and either slow in convergence or costly from communication perspective. We instead propose a one-shot solution based on client relatedness. In a concurrent work, Dennis *et al.* [30] focuses on one-shot federated clustering. After projecting client data onto a selected subspace, the iterative Lloyd's k-means clustering [31] is applied and the outcomes is communicated to the server in a one-shot fashion for cluster assignment. First, Dennis *et al.* [30] does not study the impact of the proposed clustering on downstream federated learning applications. Besides, linear subspace decomposition is in practice less efficient than the non-linear (and fine-tuned) autoencoders we use at the client side. Moreover, our manifold learning approach with UMAP [32] at the server side has the potential to approximate the underlying manifold of data in a more generic fashion as we empirically demonstrate in the following.

Notation: In the following, we use $\|X\|_l$ to denote norm-l of matrix X and $|\mathcal{X}|$ to denote the cardinality of set $\mathcal{X}$. We denote the set $\{1, \ldots, n\}$ with $[n]$. We refer to the (i, j)th element of matrix X as $X_{i,j}$ and its ith row with X_i. $\mathbf{1}_n$ denotes a row vector of size n containing all 1s.

III. Federated Learning With Taskonomy

In this section, we first formalize the classical federated learning setting and objectives and then introduce FLT.

A. Preliminaries: Refresher on FedAvg

In a classical federated learning setting, we consider M clients (in practice, hundreds) with n_m local data samples and communicate their learning regularly to a central server to reach global consensus about the whole data composed of $N = \sum_m n_m$ samples. In most prior work, the goal is to solve

$$\min_w f(w) = \sum_{m=1}^{M} p_m F_m(w) \tag{1}$$

where $p_m = (n_m/N)$ is the fraction of the total data client m sees, and thus, $\sum_m p_m = 1$. The local objective F_m is typically defined by the empirical risk over local data $F_m = (1/n_m) \sum_{j=1}^{n_m} l_j(w)$. Federated learning is conducted in regular communication rounds (server to clients, and vice versa), and per round t typically a subset of clients $\mathcal{S}^t$ are randomly selected to run stochastic gradient descent (SGD) for a given number of local epochs. This *local updating* mechanism is shown to be more flexible and efficient than mini-batch methods [33]–[35]. Algorithm 1 summarizes FedAvg [4], a pioneering method to solve (1) in a non-convex setting. The protocol is simple: in T consecutive rounds, selected client m runs E epochs of SGD (with learning rate η) on local data and shares the local model w_m with the server to be averaged

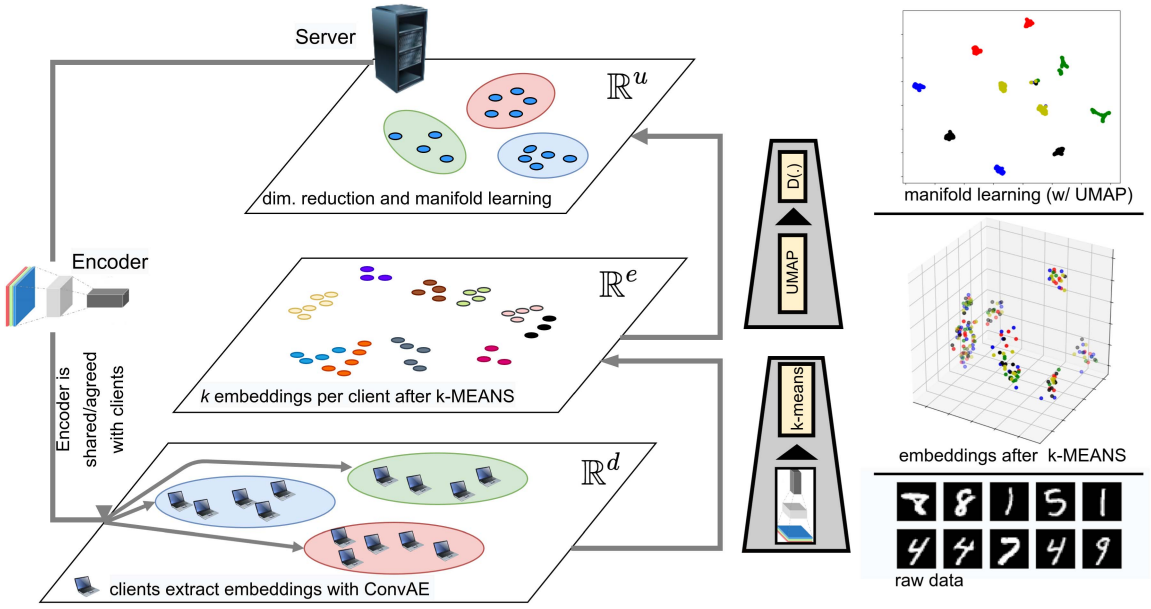


Fig. 1. High-level architecture of FLT. Clients extract their data signature using the agreed ConvAE. They then apply k-MEANS to further condense their signatures and transmit k embedding vectors to the server. The server then applies manifold learning using UMAP followed by a distance metric to form an adjacency matrix. Finally, HC [36] is applied if formation of disjoint clusters is of interest as shown in Fig. 2.

Algorithm 1 Federated Averaging (FedAvg)

Require: $\overline{M}$, T, w^0, p_m
for $t = 0, \ldots, T-1$ **do**
  Server selects a subset $\mathcal{S}^t$ of clients ($\overline{M} = |\mathcal{S}^t|$).
  Server sends w^t to all the selected clients.
  for each client m **do**
    for epoch $e = 1, \ldots, E$ **do**
      $w_m \leftarrow w_m - \eta \nabla F_m(w_m)$
    end
    $w_m^{t+1} \leftarrow w_m$
  end
  Each client m sends w_m^{t+1} to the server.
  Server aggregates w_ms and updates w:
    $w^{t+1} \leftarrow \sum_{m=1}^{\overline{M}} p_m \, w_m^{t+1}$
end

among $\overline{M}$ participating clients. For the sake of simplicity, the client participation fraction $\rho = \overline{M}/M = |\mathcal{S}^t|/|\mathcal{S}|$ is typically considered to be constant.

B. Mechanics of FLT

The majority of the clustered federated learning approaches [18]–[21], [29] enforce a hard membership constraint on the clients to form disjoint clusters where every client can belong to only one cluster. In contrast, we allow for an arbitrary symmetric task relatedness matrix with the possibility to be reordered and relapsed into disjoint clusters. To form clusters, these approaches mostly compare the clients based on their model parameters using a distance metric (such as L_2 norm or cosine similarity), which in practice does not capture the underlying manifold of data in the model parameter space or any other representation space. We instead propose a one-shot method for learning the task relatedness matrix (coined as FLT) that benefits from manifold approximation (metric learning with UMAP [32]) at the server side before applying a distance metric. In the following, we delve deeper into the mechanics of FLT.

1) Overview: A high-level sketch of the proposed approach is depicted in Fig. 1. As can be seen, we consider three abstraction levels: 1) data level, where data samples live in $\mathbb{R}^d$; 2) encoder level, where a contractive latent space representation of client data is extracted in an unsupervised fashion (samples are *nonlinearly* projected to $\mathbb{R}^e$); and 3) manifold approximation level with UMAP, where samples live in $\mathbb{R}^u$. The encoder is provided by the server to the clients. This allows them to apply one-shot contractive encoding on their local data, followed by k-means on the outcome and return the results to the server. At the server side, UMAP is applied to approximate the arriving clients' embeddings. This is followed by applying a distance threshold to determine client dependencies and form an adjacency matrix or a client (task) relatedness graph. If forming disjoint clusters is of interest, we then use HC [36] to efficiently reorder the adjacency matrix (or corresponding client relatedness graph) into disjoint clusters (see Fig. 2).

2) Learning Client (Task) Relatedness: The proposed approach, FCR, is described in Algorithm 2. The server broadcasts an encoder $G(\cdot)$ to all the M clients in $\mathcal{S}$. Note that this can also be an agreement (between the server and the clients) on using an encoder pretrained on a standard dataset (without virtually sending it), similar to the case of

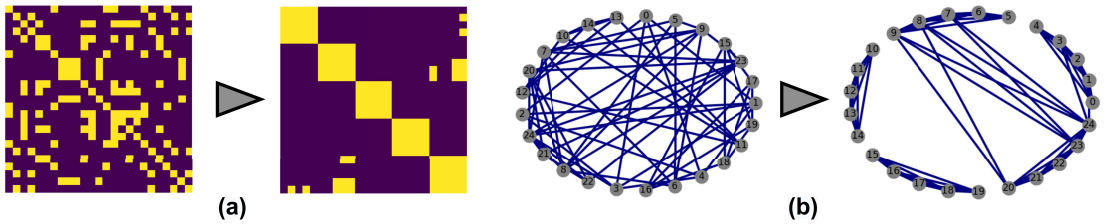


Fig. 2. Forming $C = 5$ clusters for a network of $M = 25$ clients with HC. (a) Adjacency matrix and (b) corresponding client relatedness graph (both reordered on the right).

Algorithm 2 Form Client Relatedness (FCR)

Require: MODE, $\mathcal{S}$, $G(\cdot)$, $\Gamma(\cdot)$, k, C
Server broadcasts $G(\cdot)$ to all clients in $\mathcal{S}$
for each client m in $\mathcal{S}$ **do**
  if MODE = fine-tune **then**
    Client runs F epochs to fine-tune
  end
  Client computes its own embedding:
    $\mathcal{E}_m \leftarrow G(\mathcal{D}_m)$
  Client applies k-means clustering to $\mathcal{E}_m$'s:
    $\mathcal{M}_m := \{\mu_1, \ldots, \mu_k\} \leftarrow \texttt{kMEANS}(\mathcal{E}_m)$
  Client sends $\mathcal{M}_m$ to the server
end
Server updates $\mathcal{M} := \{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$
Server (re)computes
  $\mathcal{Z} := \{\mathcal{Z}_1, \ldots, \mathcal{Z}_M\} \leftarrow \texttt{UMAP}(\mathcal{M})$
Server constructs the adjacency matrix:
$A_{i,j} := \min_{r,s} \|u_{i,r} - u_{j,s}\|_2,$
  $\forall i, j \in [M]$ & $\forall r, s \in [k]$
Server applies thresholding $\tilde{A} := \Gamma(A)$
Server forms C clusters with hiearchical clustering:
  $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_C\} \leftarrow \texttt{HC}(\tilde{A})$
Return: $\tilde{A}$ and $\mathcal{C}$

local models being used by clients in FedAvg. Nonetheless, this is *one-off*, and this downlink communication/agreement does not have to be repeated unless in an exceptional case where the server decides to change the architecture of the encoder. Note that extracting the signature of client data based on an encoder, instead of model weights for instance, is a key component of FLT, and it is inspired by taskonomy [23]. While being a significant contributor to the performance of FLT, the fact that in some scenarios this encoder has to be fine-tuned on the client data can impose an extra burden. On the other hand, since we only consider a *simple* convolutional autoencoder (ConvAE) with its frozen encoder section used for extracting latent embeddings, this fine-tuning is a straightforward process with negligible added complexity on the client side. In our experimentation (Section IV), this model is as simple as the local client models designed for the downstream task. ConvAE not only helps compressing the information that has to be sent to the server but also creates a less noisy representation of the client data. Upon receiving $G(\cdot)$ clients compute $\mathcal{E}_m := G(\mathcal{D}_m)$, where $\mathcal{D}_m$ denotes the dataset of client m ($\in [M]$) and $\mathcal{E}_m$ denotes its embedding set of size $|\mathcal{D}_m|$. The elements of $\mathcal{E}_m$ live in $\mathbb{R}^e$ with e referring to the latent embedding dimension. Even though $\mathcal{E}_m$ is compressed when compared with $\mathcal{D}_m$, it turns out that it can still be further distilled and yet capture enough information for our downstream federated learning purpose. Therefore, each client applies $\texttt{kMEANS}(\cdot)$ on $\mathcal{E}_m$ and sends the outcome $\mathcal{M}_m := \{\mu_1, \ldots, \mu_k\}$ (a set of size k) back to the server. The *fine-tune* mode is provisioned to accommodate encoders pretrained on a *totally different* dataset. In such a case, clients will be asked to run F epochs of SGD from their latest state on their most recent dataset. As we will demonstrate in Section IV, this is to establish that the choice of dataset for pretraining the encoder is not a bottleneck.

The server then constructs $\mathcal{M} := \{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$ and applies UMAP [32] to $\mathcal{M}$ and computes $\mathcal{Z} := \{\mathcal{Z}_1, \ldots, \mathcal{Z}_M\}$ with $\mathcal{Z}_m := \{u_{m,1}, \ldots, u_{m,k}\}$. $\mathcal{Z}$ contains $k \times M$ elements each living in $\mathbb{R}^u$, with u being typically 2 or 3. In most prior work, a distance metric (L_2 or cosine) is directly applied, which could be a limiting factor for non-convex risk functions and incongruent non-IID settings [20]. Instead, in FCR the server first learns the manifold in which the embeddings live using UMAP and then applies a distance metric to construct an adjacency matrix $A_{i,j} := \min_{r,s} \|u_{i,r} - u_{j,s}\|_2$, $\forall i, j \in [M]$ & $\forall r, s \in [k]$, where the minimum pairwise distance among the elements of $\mathcal{Z}_i$ and $\mathcal{Z}_j$ is taken into account. Here, for the sake of simplicity, we consider a *hard-thresholding* operator Γ applied on A leading to $\tilde{A}$, where $\tilde{A}_{i,j} = \Gamma(A_{i,j}) = \texttt{Sign}(A_{i,j} - \gamma)$ with γ a threshold value. In practice, γ can be tuned to return best performance in different settings. If constructing *explicit and disjoint* clusters is of interest, the servers apply HC [36] to reorder $\tilde{A}$ into $\tilde{A}^r$ with C disjoint clusters (diagonal blocks). Finally, the server extracts cluster membership in $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_C\}$ with $\mathcal{C}_c$s being a set of client IDs in cluster c. A flexibility that HC in [36] offers is to propose the best fitting number of clusters, if the maximum number of clusters is not specified. $\tilde{A}$ and $\tilde{A}^r$ for a toy setup with $M = 25$ clients are depicted in Fig. 2: a) adjacency matrix and b) client relatedness graph. On the left, the clients are arranged based on their IDs and on the right they are re-ordered with HC [36] to form $C = 5$ clusters. It is noteworthy that this one-shot client relatedness discovery of FCR can also happen in a few stages, in case all the clients are not available for cooperation at the initialization stage. Besides, excluding a few clients from the process (for any reason) will not impact the performance of FLT.

Algorithm 3 FLT

Require: $\mathcal{S}$, M, T, W^0, p_m
1 **Initialize Clustering:**
2 $\quad \tilde{A}, \mathcal{C} \leftarrow$ FCR(normal, $\mathcal{S}, G(\cdot), \Gamma(\cdot), k$)
3 **for** $t = 0, \ldots, T-1$ **do**
4 $\quad w_1^t, \ldots, w_M^t \leftarrow W^t$
5 $\quad$ Server selects a subset $\mathcal{S}^t$ of clients
6 $\quad$ Server sends w_m^t to all clients m in $\mathcal{S}^t$
7 $\quad$ **for** each client m *in* $\mathcal{S}^t$ **do**
8 $\quad\quad$ **for** epoch $e = 1, \ldots, E$ **do**
9 $\quad\quad\quad w_m \leftarrow w_m - \eta \nabla F_m(w_m)$
10 $\quad\quad$ **end**
11 $\quad\quad \bar{w}_m^{t+1} \leftarrow w_m$
12 $\quad$ **end**
13 $\quad$ Each client sends $\bar{w}_m^{t+1}$ and δ to the server
14 $\quad$ Server collects $\bar{W}^{t+1} = [\bar{w}_1^{t+1}, \ldots, \bar{w}_M^{t+1}]$
15 $\quad$ Server aggregates and updates $\bar{W}^{t+1}$:
16 $\quad$ **Case I**) Full adjacency matrix $\tilde{A}$:
17 $\quad\quad W^{t+1} \leftarrow \bar{W}^{t+1} \tilde{A}\, \mathtt{diag}(p_m / \|\tilde{A}_m\|_0)$
18 $\quad$ **Case II**) Disjoint clusters based on $\mathcal{C}$:
19 $\quad\quad w_c^{t+1} \leftarrow \frac{1}{|\mathcal{C}_c|} \sum_{m \in \mathcal{C}_c} p_m \bar{w}_m^{t+1}$, $\forall c \in [C]$
20 $\quad\quad W^{t+1} = [\mathbf{1}_{|\mathcal{C}_1|} w_1^{t+1}, \ldots, \mathbf{1}_{|\mathcal{C}_C|} w_C^{t+1}]$
21 $\quad$ {**Dynamic Clustering (optional):**
22 $\quad$ Server updates $\Delta = [\delta_1, \ldots, \delta_{M+M'}]$;
23 $\quad$ **if** $\sum_i \delta_i > \lambda$ **or** (t mod $\tau = 0$) **then**
24 $\quad\quad \mathcal{S} \leftarrow \mathcal{S} \cup \{M+1, \ldots, M+M'\}$
25 $\quad\quad \tilde{A}, \mathcal{C} \leftarrow$ FCR(fine-tune, $\mathcal{S}, G(\cdot), \Gamma(\cdot), k$)
26 $\quad\quad \Delta \leftarrow [0, \ldots, 0]$, $M \leftarrow M + M'$
27 $\quad$ **end**
28 $\quad$ }
29 **end**

3) Clustering Performance of FLT: Let us assume that the data of each client are generated from a mixture of L components, where L is the total number of target classes (labels). The goal of the encoder is to transform the data into a latent embedding such that separability among data labels is maximized for clustering. Let us denote the classification error of the encoder part of the pretrained ConvAE by P_e^{cae}. For sake of simplicity, assume that latent embeddings are a mixture of Gaussian components. In such a case, the following theorem provides an upper bound on the total clustering error of FLT:

Theorem 1: The clustering error of FLT is upper bounded by

$$P_{\text{err}}^{\text{tot}} \lessapprox \sum_{m=1}^{M} \sum_{l=1}^{L} \sum_{i=1}^{e} \exp\left(- \frac{\left(n_m^l \times Q^{-1}\left(P_{\text{err}}^{\text{cae}}\right)\right)^2}{2} \right) \qquad (2)$$

where, Q represents the Q-function, n_m^l denotes the total number of samples with label l for client m, and e is the dimension of the latent space.

This demonstrates that by increasing the sample size, the total clustering error of FLT vanishes, as desired. The detailed proof can be found in Appendix B-A in the supplementary material.

4) Federated Averaging With Taskonomy: FLT in Algorithm 3 starts with an initialization stage by calling the *normal* mode of FCR. This returns the adjacency (or client relatedness) matrix $\tilde{A}$ together with an optional cluster membership set $\mathcal{C}$ for the case of disjoint clustering. Next, the typical T rounds of communication akin to FedAvg will be run. The server sends across a model corresponding to each client in the (randomly) selected (time-varying) subset $\mathcal{S}^t$ (line 6). Similar to FedAvg, the client participation rate is defined as $\rho = |\mathcal{S}^t|/|\mathcal{S}|$. In case of disjoint clustering, all the clients in cluster $\mathcal{C}_c$ will be given the same model w_c^t. Each client runs F epoch on its local data and sends back the updated *local* model $\bar{w}_m^{t+1}$. The top bar notation is used to denote client-side models. The server collects all the local models in $\bar{W}^{t+1} = [\bar{w}_1^{t+1}, \ldots, \bar{w}_M^{t+1}]$ and updates them according to two possible cases. The first option (Case I) uses the full client relatedness matrix $\tilde{A}$ to benefit from the all the related client models (line 17). In this case, the server updates the local model weights using $W^{t+1} = \bar{W}^t \tilde{A}\, \mathtt{diag}(p_m / \|\tilde{A}_m\|_0)$. For notation simplicity, each model parameter set w_m^{t+1} is assumed to be reshaped into a column vector. The second option (case II) is to define an update rule per disjoint cluster according to cluster membership in $\mathcal{C}$. In that case, standard FedAvg will be applied per cluster (line 19) and the aggregated model of the cluster, w_c^{t+1}, will be sent to all the clients in the cluster in the next round (line 20).

Finally, we reflect on an *optional* dynamic clustering functionality of FLT in lines 13 and 21–28 of Algorithm 3. Here, we consider two possible circumstances: 1) the data distribution of (some) clients varies suddenly or with time, e.g., new classes are introduced and 2) newcomer clients join the network. In such cases, the clients can *optionally* send a flag of state change ($\delta := 1$) along with their updated model parameters (line 13) to notify the server of the change in their data or the state of being a newcomer. The server keeps track of these flags in Δ, and once a certain number of clients have raised such flags (say λ clients), it calls for repeating the cluster formation process. This needs further client cooperation and is prone to byzantine attacks. There are two other possibilities to do this without any client contribution. This can happen every τ round as denoted in line 18 or can happen on the server side, by monitoring client model parameter change as determining factor, similar to the main clustering approach of [21] and [22]. In any case, if (e.g. M') new clients are introduced and/or some clients have experienced data change, the server will update $\mathcal{S}$ accordingly (line 24) and calls FCR with MODE set to fine-tune, possibly resulting in a new client relatedness graph and cluster membership $(\tilde{A}, \mathcal{C})$. The new clients will then be incorporated in the next communication round. Another possibility to handle newcomer clients is to assign them to the existing cluster formation $\mathcal{C}$ based on their distance to the (average) latent embeddings of the clusters, i.e., w.r.t. an averaged representative cluster embedding. In favor of limited space, these directions are left as our future work.

5) Convergence of FLT: The following theorem generalizes convergence characteristics of FedAvg to that of FLT under same regularity assumptions. A detailed proof of this theorem is provided in Appendix B-B in the supplementary material.

Theorem 2: Let $G_{\mathbb{W}}(W^t) \triangleq \|W^t - W^{t-1}\|^2$ represent the optimality gap for the stationary solution of FLT, where W^t stands for the model parameters in communication round t. Under common regularity assumptions [37], the iterative gradient descent solution for FLT satisfies

$$\frac{1}{TM}\sum_{i=1}^{M}\sum_{t=1}^{T}\mathbb{E}\big[G_{\mathbb{W}}(W^t)\big] \leq \frac{1}{T\times M}\frac{F(W^1)-F(W^T)}{\left(\frac{1}{2\eta}-\frac{L_W}{2}-\frac{\eta}{2}L_W^2\left\|I_M-\bar{A}\right\|^2\right)} \quad (3)$$

where the cost function F is assumed to be L_W-smooth, I_M is the $M \times M$ identity matrix, and $\bar{A}$ and η denote the row-normalized adjacency matrix and the step size, respectively.

This theorem proves that akin to FedAvg, the convergence rate of FLT for a regular cost function is $\mathcal{O}(1/T)$ for T communication rounds. In Appendix B-B in the supplementary material besides the detailed proof of this theorem, we also provide a more elaborate convergence analysis for FLT where extra hyper-parameters such as the number of local epochs E are incorporated.

IV. Evaluation

We now present empirical evidence on the impact of the proposed approach, FLT. We first describe our experimental setup covering a suite of synthetic and realistic (large-scale) non-IID scenarios. We then compare the performance (accuracy and fairness) of FLT against the existing state-of-the-art baselines.

A. Experimental Setup

1) Datasets: We opt for image classification task as our downstream application of federated learning. For performance evaluation, we use four different datasets: 1) MNIST [38]; 2) CIFAR10 [39]; 3) Federated EMNIST (FEMNIST) of LEAF [40] which is made out of Extended MNIST (EMNIST) [41]; and 4) and our newly designed *Structured Non-IID FEMNIST*. The latter is a resampled version of EMNIST to assess the performance in structured non-IID scenarios; see the supplementary materials for more detailed description of this new dataset. For experiments on MNIST and CIFAR10, we, respectively, use 60 000 and 50 000 samples for training and 10 000 for testing. MNIST and CIFAR10 contain ten classes of handwritten digits and objects with 6000 and 5000 samples per class, respectively. FEMNIST is a standard federated learning image classification dataset with 805 263 samples that can accommodate up to 3550 clients. Based on EMNIST and similar to FEMNIST, we build a new dataset with (more pronounced) structured non-IID conditions and call it Structured Non-IID FEMNIST. To do so, we consider the "balanced" dataset of EMNIST, containing 131 600 samples on 47 classes. We use 112 800 for training (2400 samples per class) and the remainder 18 800 for testing. Depending on the scenarios explained later on, we partition the data into $C = 5$ or 10 clusters. For experiments on FEMNIST, there is no predefined partitioning (or clustering) and we follow the standard definitions of LEAF [40].

2) Encoders: To investigate the impact of encoder initialization, we also make use of three other datasets: a subset of CIFAR100 [42] (without any overlap with 10 classes of CIFAR10), a 20-class dataset composed of CIFAR10 and 10 classes of CIFAR100 (we refer to this as "CIFAR20"), and Fashion MNIST [43]. More concretely, we evaluate the performance of the proposed method in two encoder scenarios.

1) *Enc1:* The encoder provided to the clients is pretrained on a large set of targets that covers the client target classes. This case is hoping for a holistic encoder on the server side. More specifically, for federated learning on MNIST we have used an encoder pretrained on EMNIST, and for CIFAR10 we use CIFAR20 introduced earlier.
2) *Enc2:* The encoder is pretrained on a totally different dataset, and an initial fine-tuning per client would be required. This case demonstrates that lacking the holistic encoder (Enc1) is not a bottleneck for FLT. Note that in this scenario the encoder (ConvAE) is fine-tuned (also pre-trained) in an *unsupervised* way and independent of the downstream federated learning task for J more epochs based on a mean-squared-error (MSE) loss. In this case, for federated learning on MNIST we use an encoder pretrained on Fashion MNIST, and for CIFAR10 we use CIFAR100, which has no overlap with CIFAR10. For experiments on EMNIST/FEMNIST and Structured Non-IID FEMNIST, we only considered Enc2 pretrained on Fashion MNIST.

3) Network Parameters: We consider two model architectures for local client training, a multilayer perceptron (MLP) and a convolutional neural network (CNN). We use an MLP with ReLU activation and a single hidden layer of size 200. We use the CNN of LEAF [40] with two convolutional layers followed by two fully connected layers. The encoder section of our frozen ConvAE has two convolutional layers followed by a single fully connected layer. Note that the encoder is as simple as the local client model. See the supplementary materials for more details on model architectures. We set the number of local epochs to $E = 5$, and the total communication rounds to $T = 100$, unless otherwise mentioned. The local training is a mini-batch SGD with a batch size of 10 and a learning rate of $\eta = 0.01$. For FLT, the size of the latent embedding is $e = 128$ and k in kMEANS is set to 5. Even though on the client side k can be adjusted according to the number of client classes. The number of fine-tuning epochs for Enc2 is set to $J = 5$, and $\gamma = 1$. The client participation fraction $\rho = 20\%$, unless otherwise mentioned. For HC, we use Ward's method for linkage creation [36].

B. Evaluation Scenarios

According to [3] and [4], there are several possible sources of non-IIDness in client data distributions. Among those, label distribution skew, or the so-called "pathological" partitioning, is the most commonly adopted approach in literature. In this case, different clients will have different class labels, which

together with quantity skew (varying number of samples across clients) leads to most destructive impact on FedAvg [44]. We build the following scenarios upon these angles.

1) Scenario 1 [MNIST, CIFAR10]: We consider a network of $M = |\mathcal{S}| = 100$ clients clustered into $C = 5$ clusters. The training data samples will be evenly distributed among these clients, 600 samples each (500 for CIFAR10), and the clients in each cluster will have samples only from two distinct classes. For instance, for MNIST, clients in cluster 1 ($\mathcal{C}_1$) will have only samples from digits "0" and "1" and those in $\mathcal{C}_2$ from "4" and "7" without overlap with $\mathcal{C}_1$.

2) Scenario 2 [MNIST, CIFAR10]: This scenario is the same as Scenario 1, except that the clients in two different clusters can have one similar label/class. As an example, for MNIST, clients in cluster 1 ($\mathcal{C}_1$) will have only samples from digits "0", "1," and "2," and those in $\mathcal{C}_2$ can have samples from "2," "3," and "7." This is to investigate the performance in less extreme non-IID conditions.

3) Scenario 3 [FEMNIST]: We import the standard FEMNIST dataset of LEAF and construct a network of 200 clients according to train and test data distributions defined in [40]. We run our experiments for a total of $T = 100$, 1000, and 1500 communication rounds. There are no predefined (or structured) clusters in FEMNIST, and it is up to the federated learning method to form clusters, if need be. Here, we consider both the cases in Algorithm 3 and assess the performance of FLT in the generic form (no clusters, case I) and with $C = 2$, 3, 5, and 7 clusters (case II).

4) Scenario 4 [Structured Non-IID FEMNIST]: As mentioned earlier, we introduced this dataset with the purpose of imposing (more extreme) structured non-IIDness in FEMNIST. To this aim, we impose label distribution skew (across clusters) and quantity skew following a power law for the number of samples per client in each cluster, akin to [13]. We consider $C = 10$ clusters, each containing five distinct character classes (a total of 12 000 data samples per cluster), except the last one containing two classes (4800 samples), resulting in a total of 47 classes and 112 800 samples. We also consider even a larger network than Scenario 3 with $M = 2400$ clients (240 clients per cluster) for MLP and $M = 600$ clients (100 clients per cluster) for CNN. Notably, as a result of our random sampling strategy, the clients in each cluster will have a random subset of the labels assigned to that cluster.

C. Baselines and Competitors

1) Baselines: We consider five baselines as described in the following: 1) FedAvg [4] where a *single global model* is trained for the whole network; 2) local where each client trains its own model with its own local data; and 3) PCA + kM + HC Inspired by [30] (focused on federated clustering), this method applies linear principle component analysis (PCA) followed by kMEANS (at the client side) and HC at the server side. The goal is to illustrate the impact of the nonlinear encoder (convAE) and manifold learning (UMAP) components of FLT for discovering task relatedness. iv, v) We also compare our performance with two of the most recent state-of-the-art clustered federated learning approaches called IFCA [18] and FedSEM [22]. Notably, FedSEM already outperforms other recent baselines such as FedProx [13] and CFL [20], and thus has been selected as the outstanding approach. For the sake of reference though, we also report the result of CFL in Scenario 3. FedSEM constructs multiple clusters each building its own local cluster-level model. For each cluster, a virtual cluster center is defined and its parameters are updated in an iterative fashion. The key idea behind cluster formation is measuring the distance between clients and virtual cluster center model parameters. IFCA tries to construct multiple clusters iteratively by alternating between cluster identification estimation and loss function minimization. It starts with initializing model parameters for cluster centers which are then broadcast to the randomly participating clients per communication round (a costly communication overhead). The clients estimate their cluster identity by finding the model that returns the lowest loss and send their cluster identity along with the updated model. For both the methods: 1) the number of clusters has to be known a priori; 2) a good initialization of cluster center model parameters is key for convergence; and 3) they require several communication rounds before clusters are formed. We will demonstrate in Section IV-F that these characteristics can lead to slow convergence and suboptimal performance in *structured non-IID* scenarios.

TABLE I
TEST ACCURACIES (%± STD. ERROR) FOR SCENARIO 1

	MNIST		CIFAR10	
Method	acc.	var.	acc.	var.
FedAvg	82.73±0.38	44.65	33.29±0.47	108.20
Local	97.41±0.16	3.44	79.81±0.40	115.35
PCA+kM+HC	83.82±0.37	41.13	75.35±0.40	589.6
FedSEM	98.01±0.14	2.12	78.23±0.41	695.01
FLT (Enc1)	97.98±0.14	2.17	87.11±0.33	88.80
FLT (Enc2)	97.97±0.14	2.17	87.17±0.33	84.62

2) Fairness in Federated Learning: Recently, *fairness* in performance has become an important concern in federated learning. In this context, being "fair" is to avoid disproportionately advantaging or disadvantaging some of the clients. Among several interesting approaches to fairness, two recent ones stand out in the federated learning literature: 1) best worst case performance [8], [45], and 2) least variance across clients [9]. We focus on the latter and report the variance of model accuracies across clients as a measure of fairness.

D. Evaluation Results for Scenarios 1 and 2

Average model accuracies on *test* data (and their standard errors) for Scenarios 1 and 2 after $T = 100$ communication rounds are shown in Tables I and II, respectively. These are accompanied by the variance of model accuracies across clients which is serving as our fairness metric [9]. For MNIST, the number of local epochs is set to $E = 1$ and for CIFAR10 is set to $E = 5$. Note that irrespective of the choice of the encoder (Enc1 or Enc2), the proposed method (FLT) achieves

TABLE II
TEST ACCURACIES (%± STD. ERROR) FOR SCENARIO 2

	MNIST		CIFAR10	
Method	acc.	var.	acc.	var.
FedAvg	86.08±0.35	17.55	52.77±0.50	19.18
Local	96.58±0.17	1.81	70.44±0.46	72.29
PCA+kM+HC	85.45±0.35	18.42	61.75±0.49	312.22
FedSEM	94.14±0.23	1.20	73.26±0.44	339.39
FLT (Enc1)	97.37±0.16	1.07	79.971±0.40	146.84
FLT (Enc2)	97.37±0.16	1.07	80.00±0.40	78.66

TABLE III
TEST ACCURACIES (%± STD. ERROR) FOR SCENARIO 3

	MLP		CNN	
Method	acc.	var.	acc.	var.
FedAvg	72.76±0.76	202.61	81.64±0.66	147.03
PCA+kM+HC$_{(C=3)}$	71.94±0.77	194.75	80.07±0.68	160.32
IFCA [18]	61.24±0.84	176.38	81.47±0.66	118.71
CFL [20]	65.35±0.81	180.23	76.68±0.73	132.42
FedSEM [22]	72.45±0.76	185.96	79.99±0.68	156.27
FLT	74.11±0.74	171.31	82.14±0.65	145.03
FLT$_{(C=2)}$	72.83±0.76	175.55	80.53±0.68	150.26
FLT$_{(C=3)}$	72.61±0.76	184.59	79.72±0.69	170.10
FLT$_{(C=5)}$	72.05±0.77	180.32	78.71±0.70	169.22

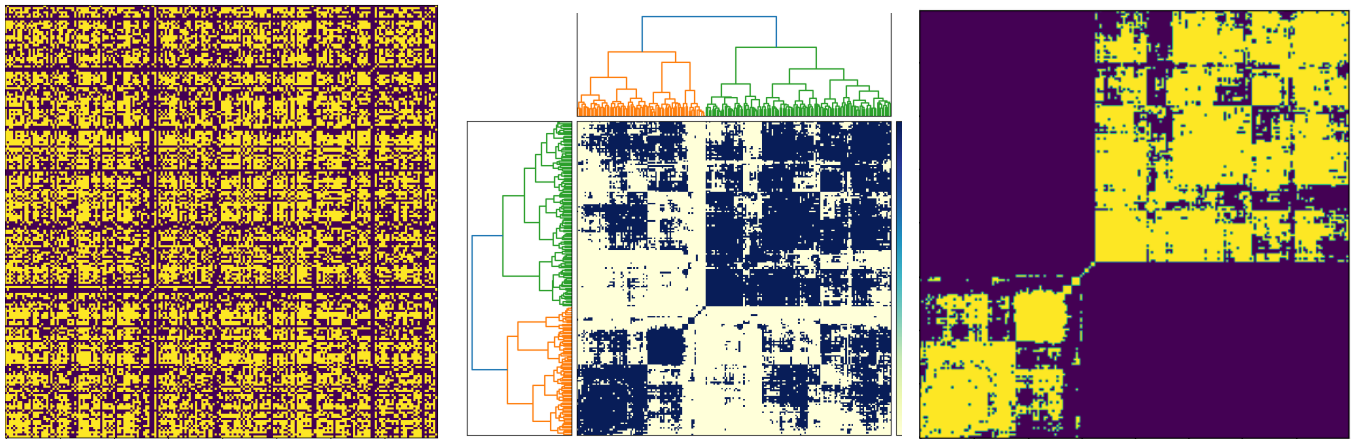
roughly the same performance in terms of accuracy in both the scenarios. This highlights that our method is reasonably robust against the choice of the encoder and the fine-tuning idea (starting from a totally different dataset and finetuning only for $J = 5$ epochs) for Enc2 is functioning. As can be seen in both the tables, even with $E = 1$, the differences between FLT and FedSEM are relatively smaller on MNIST but this discrepancy grows for CIFAR10. In summary, on MNIST and CIFAR10, we improve upon the state-of-the-art approach FedSEM by about 3% and 9% respectively, and way beyond that for the case of FedAvg, local, and PCA + kM + HC. On top of accuracy, FLT returns the least test accuracy variance (best fairness) for almost all the settings in the two scenarios.

There are two reasons behind achieving reasonable performance by local models: 1) in these two scenarios clients have a reasonably high number of samples (600 for MNIST and 500 for CIFAR10) and 2) the target classes in different clusters are almost independent. Note that extreme non-IIDness (total class independence across clusters) in Scenario 1 helps local models score better than in Scenario 2 where some correlation between labels is allowed. A counter-argument applies to our method FLT because hard thresholding (mapping cluster membership to 0s and 1s) for Γ in Algorithm 2 seems to be limiting the performance in handling inter-cluster dependencies. Local models do not illustrate competitive performance. Therefore, we omit them in the following experiments.

E. Evaluation Results for Scenarios 3

Predefined data distribution of FEMNIST in LEAF [40] and its adoption by other state-of-the-art methods helped us to also benchmark our approach against IFCA [18]. Here, we present the performance results based on both MLP and CNN networks for $M = 200$ clients, a setting commonly adopted in literature. For FLT, we only present the more challenging Enc2 case where the encoder is pretrained on Fashion MNIST, and in a *one-off process* each client fine-tunes the encoder (for only $J = 5$ epochs). Ghosh *et al.* [18] use a slightly modified data sampling strategy; however, we use standard data distribution of FEMNIST as in [40]. They also run for a total of $T = 6000$ communication rounds to reach the nominal performance, 2000 of which is initialization with FedAvg for "weight sharing" for a smaller client participation rate ($\rho = 3\%$). For the sake of fair comparison, we run 500 rounds of initialization (with FedAvg) followed by another 1000 rounds of IFCA itself, i.e., a total of 1500 rounds. All the MLP experiments are run for $T = 1000$ communication rounds, and those for CNN are run for only $T = 100$ rounds, except for IFCA which is again run for 1500 rounds for both MLP and CNN. Remember that FEMNIST does not introduce any predefined cluster structure, and thus non-IIDness in the structured form. As such, this is up to the methodology to define clusters. Both IFCA and FedSEM must define clusters ($C > 1$) or with $C = 1$ they would degenerate to FedAvg; however, thanks to FCR in Algorithm 2, this is not the case for FLT. We adopt the setting leading to the best performance reported by both IFCA and FedSEM with $C = 3$ and $C = 5$ clusters, respectively. For FLT, we present the results for both the cases described in Section III: 1) where the full client relatedness matrix will be used and 2) where we use HC to specifically extract disjoint clusters (only if need be, as a special case). This is shown in Fig. 3 where (a) shows the raw client relatedness matrix, (b) illustrates how HC would reorder this to extract cluster-level dependencies (dendrogram), and (c) shows the flattened version with $C = 2$ disjoint clusters.

The *test* accuracies and fairness measures are summarized in Table III. Using the full client relatedness matrix, FLT beats all the competitors (by +2%, +12%, +1.5% for PCA + kM + HC, IFCA, FedSEM) for MLP and (by +2%, +0.5%, +2% for PCA + kM + HC, IFCA, FedSEM) in case of CNN local models. As also reported in [22], FedSEM already beats CFL, so does FLT by a siginificant margin of +6 (for CNN) to +9% (for MLP). Notably, FLT significantly outperforms IFCA in MLP setting and marginally outperforms in the case of CNN. However, in case of CNN, all the models except IFCA are run for only 100 communication rounds. In practice, it took IFCA 1500 communication rounds to reach a comparable performance regime. For this reason, we omit IFCA in our next experiments. Note that PCA + kM + HC (inspired by the line of thought in [30] and embedded in an end-to-end federated learning setting) also performs in par with the best methods and considerably better than IFCA in MLP setting. For the sake of fair comparison, HC with $C = 3$ clusters is applied here. The convergence graphs of average *test* accuracies are illustrated in Fig. 4(a) and (b) where FLT is the fastest in terms of convergence. We also investigated the impact of creating disjoint clusters with FLT



(a) (b) (c)

Fig. 3. Hierarchical clustering (HC) is used to reorder client relatedness graph ($\tilde{A} \rightarrow \tilde{A}^r$) and extract near-optimal block diagonal structures for disjoint cluster formation. In (b), the dendrogram highlights the client dependencies leading to $C = 2$ disjoint clusters. (a) Raw adjacency matrix. (b) Reordered with HC. (c) Flattened to $C = 2$ clusters.

TABLE IV
TEST ACCURACIES (%± STD. ERROR) FOR SCENARIO 4

	MLP		CNN	
Method	acc.	var.	acc.	var.
FedAvg	46.50±0.36	100.27	53.58±0.36	883.47
PCA+kM+HC	48.75±0.36	467.92	59.87±0.36	912.39
FedSEM [22]	43.53±0.36	406.60	76.22±0.29	1264.38
FLT	86.51±0.24	207.60	93.69±0.10	98.65

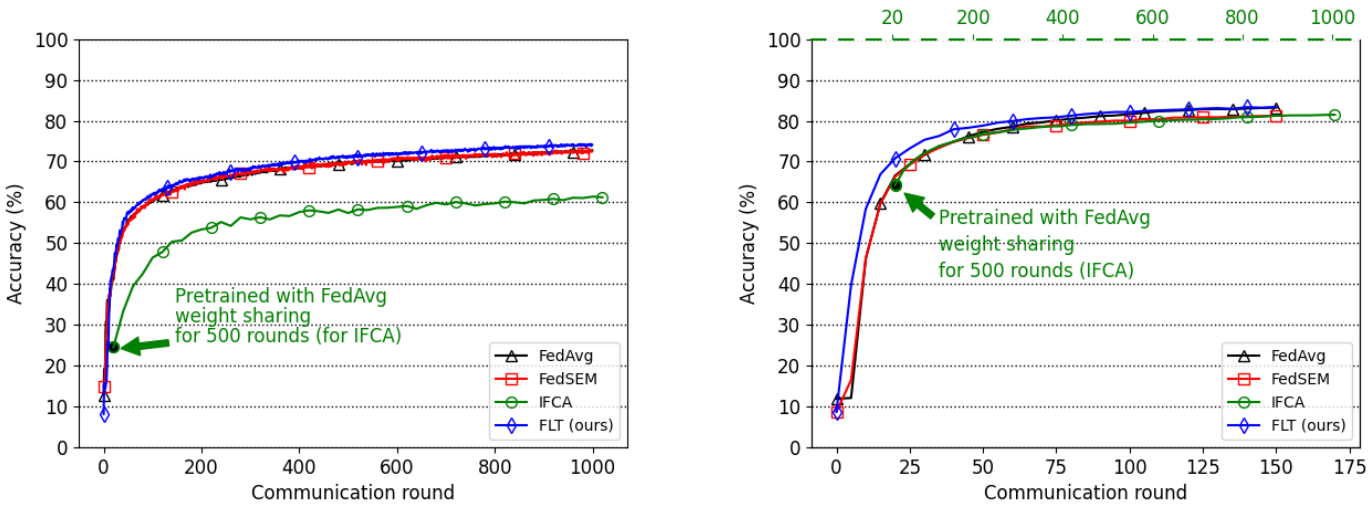
TABLE V
ABLATION STUDY OF *Test* ACC. (%± STD.) FOR SCN. 2

	FedSEM [22]	FLT (ours)	
(M, ρ, E, T)		Enc1	Enc2 ($J = 5$)
(100, 0.2, 1, 50)	64.32±0.48	66.23±0.47	65.93±0.47
(100, 0.2, 5, 50)	71.85±0.45	79.45±0.40	78.64±0.41
(100, 0.2, 1, 100)	69.33±0.46	72.44±0.45	72.62±0.45
(100, 0.2, 5, 100)	73.42±0.44	79.95±0.40	79.62±0.40
(100, 0.5, 1, 50)	72.94±0.44	73.78±0.44	73.17±0.44
(100, 0.5, 5, 50)	74.29±0.44	80.02±0.40	79.22±0.41
(100, 0.5, 1, 100)	77.37±0.42	79.30±0.41	78.23±0.41
(100, 0.5, 5, 100)	74.15±0.44	79.72±0.40	79.45±0.40
(500, 0.2, 1, 50)	64.88±0.48	66.69±0.47	66.74±0.47
(500, 0.2, 5, 50)	71.24±0.44	80.27±0.40	79.77±0.40
(500, 0.2, 1, 100)	70.65±0.46	72.55±0.46	72.84±0.44
(500, 0.2, 5, 100)	73.21±0.45	81.77±0.39	81.00±0.39
(500, 0.5, 1, 50)	72.26±0.45	73.80±0.44	73.88±0.44
(500, 0.5, 5, 50)	76.79±0.42	81.97±0.37	81.08±0.39
(500, 0.5, 1, 100)	76.82±0.42	79.66±0.40	79.01±0.41
(500, 0.5, 5, 100)	76.75±0.42	81.57±0.39	81.22±0.39

(case II in Algorithm 3) in this setting. Introducing clusters ($C = 2$, 3, and 5) in this dataset seems to be slightly degrading the performance of FLT, even though it still remains to be in par with the best performing models. See the supplementary materials for more detailed results. Reflecting on the relatively smaller performance margin in this scenario, we argue that FEMNIST may not have a clear cluster *structure*, and thus, cluster-based methods might not offer a significant gain. This also concurs with that FLT using the full adjacency matrix slightly outperforms its clustered variants. This was the main motivation behind designing Scenario 4 to assess the potential of these algorithms in a more challenging large-scale setting: Structured Non-IID FEMNIST.

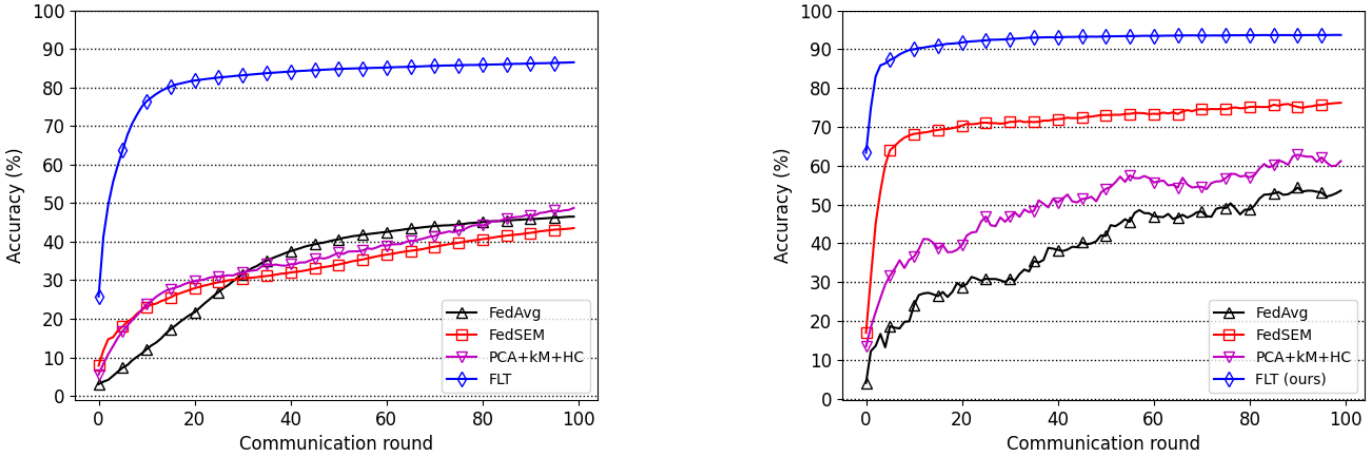
F. Evaluation Results for Scenarios 4

As explained earlier, Scenario 4 presents a large-scale federated learning setting with structured non-IIDness involving both quantity and label distribution skews. For FLT, we only present the more challenging Enc2 case where the encoder is pretrained on Fashion MNIST, and in a *one-off process* each client fine-tunes the encoder (for only $J = 5$ epochs). The convergence graphs of average *test* accuracies are shown in Fig. 5(a) and (b), for MLP and CNN networks, respectively. The *test* accuracies (and their standard error) together with the fairness measure (variance across clients) for the last communication round $T = 100$ are summarized in Table IV. Interestingly, the state-of-the-art competitor, FedSEM, suffers in the MLP scenario and roughly scores as good as the barebone FedAvg, which is not made to cope with structured non-IID scenarios. The performance of FedSEM is noticeably improved in CNN setting, but accordingly FLT is boosted as well, reaching +93% *test* accuracy in this challenging scenario. Here, we beat FedSEM by +40% and +17% in MLP and CNN scenarios, respectively. The main reason behind this downgrade in performance of FedSEM (compared with Scenario 3) is the struggle to discover and form clusters by relying on model parameter comparisons. This is in turn due to two main factors: 1) significantly larger number of models (per cluster and across) for this extreme non-IID scenario leads to tremendous heterogeneity in model space and thus considerable increase in complexity of pairwise model comparisons and 2) the problem is exacerbated due to the limited number of samples provided to some clients (down



(a) (b)

Fig. 4. Convergence graph of *test* accuracies for Scenario 3, FEMNIST, $M = 200$ (left: MLP, right, CNN). On the right for CNN, note the different range of communication rounds on the top horizontal axis associated with `IFCA`. (a) Test accuracies for MLP. (b) Test accuracies for CNN.



(a) (b)

Fig. 5. Convergence graph of *test* accuracies for Scenario 4, Structured Non-IID FEMNIST, $C = 10$. (a) Test accuracies for MLP, $M = 2400$. (b) Test accuracies for CNN, $M = 600$.

to 20 samples because of the power law) resulting in lower quality local model training. An evidence confirming this hypothesis is that these problems are even more pronounced in the case of a simpler local model (MLP), where `FedSEM` falls almost back to `FedAvg`. On the other hand, thanks to `FCR` on client side, and `UMAP` and `HC`) on the server side, `FLT` manages to automatically detect $C = 10$ clusters (see Appendix A in the supplementary material for more results). Both `FedAvg` and `FedSEM` can benefit from more communication rounds in this scenario. Nonetheless, the main message of this experiment is clear: the one-shot taskonomy-based client relatedness helps `FLT` converge already in roughly 20 communication rounds and outperform the other baselines by a significant margin, while also offering a lower variance across clients (best fairness) compared with most competitors.

G. Impact of Hyper-Parameter Changes

We investigate the impact of a few remaining important hyper-parameters on the performance of `FedSEM` and `FLT` with both Enc1 and Enc2 initialization. We ran this experiment on Scenario 2 due to being an intermediate case where label overlap is allowed. Following this scenario, we set $C = 5$, and focus on CIFAR10, the more challenging dataset. The results are summarized in Table V. Here, we consider $M = 100$ and 500, client participation fraction per communication round $\rho = 20\%$ and 50% (denoted as 0.2 and 0.5), $E = 5$, and $J = 5$ (number of fine-tuning steps in Enc2 initialization), and we report the *test* accuracy after $T = 50$ and 100 communication rounds. As can be seen in the table, changing the total number of clients M does not change the performance significantly (by looking at the top and bottom halves of the table), whereas increasing the communication rounds T leads to better performance. Besides, for the same M, increasing the client participation fraction ρ improves the overall performance of both the methods. Furthermore, for the same M and ρ, increasing the number of local epochs E results in a superior performance, all of which are in line with the more elaborate version of Theorem 2, in Appendix B-B in the supplementary material. Without exception, our method outperforms `FedSEM` by roughly 2%–8%.

TABLE VI
COMMUNICATION COMPLEXITY ANALYSIS

FLT	FedSEM [22]	IFCA [18]
$\underbrace{M * W_{\text{enc}} + k\,M\,e}_{\text{one-off}} + 2\rho M W_{\text{local}} T$	$2\rho M W_{\text{local}} T$	$\rho M W_{\text{local}} T(C+1)$

V. CONCLUSION

A. Summary and Future Directions

We proposed FLT that comes with the following notable advantages. First, it is one-shot and considerably faster in convergence compared with its competitors, especially in structured non-IID scenarios. Second, the problem formulation of FLT can handle both generic client relatedness (no specific clustering) and disjoint clusters, if need be. Third, in contrast to most existing baselines, it does not require prior knowledge about the number of clusters to form them. Fourth, it performs slightly better than the state-of-the-art baselines in standard federated learning settings and significantly outperforms them in structured non-IID scenarios. Finally, FLT offers improved fairness (least performance disparity among clients) compared with the existing baselines in most presented scenarios. Finally, in Appendix B in the supplementary material, we also provide a detailed convergence proof for FLT under common assumptions required for the convergence of FedAvg. We are currently working on extending FLT to handle dynamic clustering in the presence of newcomers or time-varying data distributions, as is briefly discussed in Algorithm 3. Another avenue to explore is removing the hard threshold Γ and modifying HC to work with soft (non-binary) adjacency matrices.

B. Complexity and Practical Considerations

FLT introduces a *one-off* overhead due to the client relatedness discovery process (FCR, Algorithm 2). However, owing to FCR, it is faster than the existing iterative baselines and less prone to convergence issues, as we have demonstrated throughout Section IV and Appendix A in the supplementary material. One can argue that this step can be prone to security issues during uplink communication (akin to standard FedAvg communications). A possible solution to address this is adding encryption and client ID verification processes, which are outside the scope of our work [6], [46], [47]. From communication complexity perspective, this overhead requires the server to send an encoder model (W_{enc}) to the clients and the clients to send an array of size ke (with k in k-means and e denoting the latent embedding dimension of the encoder) to the server. A rough estimate of the communication complexity of the proposed FLT and two discussed state-of-the-art competitors (FedSEM and IFCA) is summarized in Table VI. As can be seen, the communication complexity of FLT and FedSEM is essentially the same except for the first two one-off terms (without T for total communication rounds) which could be neglected. Note that this is an initialization step and it can also happen in multiple steps. Excluding a few clients from this process due to, for instance, their unavailability does not impact the performance of FCR and in turn FLT. On the other hand, IFCA mandates roughly $(C+1)/2$ times (C being the number of clusters) more communication complexity. This is because in every communication round, C virtual center models will have to be sent to all the participating clients. From compute complexity perspective, possible *fine-tuning* of the encoder is only for a small number of epochs ($J = 5$ in our experiments) on an encoder which is as simple as the local client models, and this is yet another one-off process that can be neglected over long runs. When the number of clients grows to tens of thousands (in very large-scale networks), FLT has the flexibility to decompose the client relatedness graph into disjoint clusters and degenerate to the same complexity level its competitors inflict.

ACKNOWLEDGMENT

The authors would like to thank Shell Global Solutions International B.V. and Delft University of Technology (TU Delft) for the support and for the permission to publish this work. They would also like to thank Attila Szabó from the University of Amsterdam (UvA) and Ahmad Beirami from Meta AI for helpful discussions.

REFERENCES

[1] J. Konečnỳ, B. McMahan, and D. Ramage, "Federated optimization: Distributed optimization beyond the datacenter," 2015, *arXiv:1511.03575*.
[2] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016, *arXiv:1610.05492*.
[3] E. B. P. Kairouz and H. B. McMahan, "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, no. 1, pp. 1–210, 2021.
[4] 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.
[5] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-IID data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3400–3413, Sep. 2019.
[6] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," 2017, *arXiv:1712.07557*.
[7] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2938–2948.
[8] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 4615–4625.
[9] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–27.
[10] A. T. Suresh, B. McMahan, P. Kairouz, and Z. Sun, "Can you really backdoor federated learning?" 2019, *arXiv:1911.07963*.
[11] H. Wang *et al.*, "Attack of the tails: Yes, you really can backdoor federated learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 16070–16084.
[12] 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.

[13] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2018, *arXiv:1812.06127*.

[14] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 4424–4434.

[15] Y. Jiang, J. Konečnỳ, K. Rush, and S. Kannan, "Improving federated learning personalization via model agnostic meta learning," 2019, *arXiv:1909.12488*.

[16] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," in *Advances in Neural Information Processing Systems*, vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds. Red Hook, NY, USA: Curran Associates, 2020, pp. 3557–3568.

[17] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 10713–10722.

[18] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," in *Advances in Neural Information Processing Systems*, vol. 33. Red Hook, NY, USA: Curran Associates, 2020, pp. 19586–19597.

[19] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," 2020, *arXiv:2002.10619*.

[20] 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.

[21] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-IID data," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–9.

[22] M. Xie, G. Long, T. Shen, T. Zhou, X. Wang, and J. Jiang, "Multi-center federated learning," 2020, *arXiv:2005.01026*.

[23] A. Zamir, A. Sax, W. Shen, L. Guibas, J. Malik, and S. Savarese, "Taskonomy: Disentangling task transfer learning," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 3712–3722.

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

[25] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 5132–5143.

[26] Y. Yeganeh, A. Farshad, N. Navab, and S. Albarqouni, "Inverse distance aggregation for federated learning with non-IID data," in *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning*. Cham, Switzerland: Springer, 2020, pp. 150–159.

[27] Y. Laguel, K. Pillutla, J. Malick, and Z. Harchaoui, "Device heterogeneity in federated learning: A superquantile approach," 2020, *arXiv:2002.11223*.

[28] Z. Zhao *et al.*, "Federated learning with non-IID data in wireless networks," *IEEE Trans. Wireless Commun.*, early access, Sep. 3, 2021, doi: 10.1109/TWC.2021.3108197.

[29] M. Duan *et al.*, "FedGroup: Efficient clustered federated learning via decomposed data-driven measure," 2020, *arXiv:2010.06870*.

[30] D. K. Dennis, T. Li, and V. Smith, "Heterogeneity for the win: One-shot federated clustering," in *Proc. 38th Int. Conf. Mach. Learn.*, vol. 139, Jul. 2021, pp. 2611–2620.

[31] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. IT-28, no. 2, pp. 129–137, Mar. 1982.

[32] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform manifold approximation and projection for dimension reduction," 2018, *arXiv:1802.03426*.

[33] S. U. Stich, "Local SGD converges fast and communicates little," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–17.

[34] J. Wang and G. Joshi, "Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms," 2018, *arXiv:1808.07576*.

[35] B. E. Woodworth, J. Wang, A. Smith, B. McMahan, and N. Srebro, "Graph oracle models, lower bounds, and gaps for parallel stochastic optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 8496–8506.

[36] D. Müllner, "Modern hierarchical, agglomerative clustering algorithms," 2011, *arXiv:1109.2378*.

[37] S. Wang and T.-H. Chang, "Federated matrix factorization: Algorithm design and application to data clustering," 2020, *arXiv:2002.04930*.

[38] Y. LeCun. (1998). *The MNIST Database of Handwritten Digits*. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[39] A. Krizhevsky, V. Nair, and G. Hinton. (2014). *The CIFAR-10 Dataset*. [Online]. Available: http://www.cs.toronto.edu/kriz/cifar.html

[40] S. Caldas *et al.*, "LEAF: A benchmark for federated settings," 2018, *arXiv:1812.01097*.

[41] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, "EMNIST: Extending MNIST to handwritten letters," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 2921–2926.

[42] A. Krizhevsky *et al.*, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep. 2009-TR, 2009.

[43] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.

[44] T.-M. Harry Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," 2019, *arXiv:1909.06335*.

[45] T. Hashimoto, M. Srivastava, H. Namkoong, and P. Liang, "Fairness without demographics in repeated loss minimization," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1929–1938.

[46] K. Bonawitz *et al.*, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2017, pp. 1175–1191.

[47] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 5, pp. 1333–1345, May 2018.

[48] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *J. Amer. Stat. Assoc.*, vol. 58, no. 301, pp. 13–30, 1963.

[49] J. Bolte, S. Sabach, and M. Teboulle, "Proximal alternating linearized minimization for nonconvex and nonsmooth problems," *Math. Program.*, vol. 146, nos. 1–2, pp. 459–494, Aug. 2014.

[50] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*, vol. 87. London, U.K.: Springer, 2003.