An Efficient Framework for Clustered Federated Learning

Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran, *Fellow, IEEE*

***Abstract*— We address the problem of federated learning (FL) where users are distributed and partitioned into clusters. This setup captures settings where different groups of users have their own objectives (learning tasks) but by aggregating their data with others in the same cluster (same learning task), they can leverage the strength in numbers in order to perform more efficient federated learning. For this new framework of clustered federated learning, we propose the *Iterative Federated Clustering Algorithm* (IFCA), which alternately estimates the cluster identities of the users and optimizes model parameters for the user clusters via gradient descent. We analyze the convergence rate of this algorithm first in a linear model with squared loss and then for generic strongly convex and smooth loss functions. We show that in both settings, with good initialization, IFCA is guaranteed to converge, and discuss the optimality of the statistical error rate. In particular, for the linear model with two clusters, we can guarantee that our algorithm converges as long as the initialization is slightly better than random. When the clustering structure is ambiguous, we propose to train the models by combining IFCA with the weight sharing technique in multi-task learning. In the experiments, we show that our algorithm can succeed even if we relax the requirements on initialization with random initialization and multiple restarts. We also present experimental results showing that our algorithm is efficient in non-convex problems such as neural networks. We demonstrate the benefits of IFCA over the baselines on several clustered FL benchmarks.**

***Index Terms*— Federated learning, clustering, alternating minimization.**

I. Introduction

IN MANY modern data-intensive applications such as recommendation systems, image recognition, and natural language processing, distributed computing has become a crucial component. In many applications, data are stored in end users' own devices such as mobile phones and personal computers, and in these applications, fully utilizing the on-device machine intelligence is an important direction for next-generation distributed learning. Federated learning (FL) [1]–[3] is a recently proposed distributed computing paradigm that is designed towards this goal, and has received significant attention. Many statistical and computational challenges arise in federated learning, due to the highly decentralized system architecture. In this paper, we propose an efficient algorithm that aims to address one of the major challenges in FL—dealing with heterogeneity in the data distribution.

In federated learning, since the data source and computing nodes are end users' personal devices, the issue of data heterogeneity, also known as non-i.i.d. data, naturally arises. Exploiting data heterogeneity is particularly crucial in applications such as recommendation systems and personalized advertisement placement, and it benefits both the users' and the enterprises. For example, mobile phone users who read news articles may be interested in different categories of news like politics, sports or fashion; advertisement platforms might need to send different categories of ads to different groups of customers. These indicate that leveraging the heterogeneity among the users is of potential interest—on the one hand, each machine itself may not have enough data and thus we need to better utilize the similarity among the users; on the other hand, if we treat the data from all the users as i.i.d. samples, we may not be able to provide personalized predictions. This problem has recently received much attention [4]–[6].

In this paper, we study one of the formulations of FL with non-i.i.d. data, i.e., the *clustered federated learning* [5], [7]. We assume that the users are partitioned into different clusters; for example, the clusters may represent groups of users interested in different categories of news, and our goal is to train models for every cluster of users. We note that cluster structure is very common in applications such as recommendation systems [8], [9]. The main challenge of our problem is that the *cluster identities of the users are unknown*, and we have to simultaneously solve two problems: identifying the cluster membership of each user and optimizing each of the cluster models in a distributed setting. In order to achieve this goal, we propose a framework and analyze a distributed method, named the *Iterative Federated Clustering Algorithm (IFCA)* for clustered FL. The basic idea of our algorithm is a strategy that alternates between estimating the cluster identities and minimizing the loss functions, and thus can be seen as an alternating minimization algorithm in a distributed setting. We compare with a simple one-shot clustering algorithm and

Manuscript received 19 October 2021; revised 22 April 2022; accepted 7 July 2022. Date of publication 19 July 2022; date of current version 22 November 2022. This work was supported in part by NSF under Grant CIF-1703678 and in part by the Machine Learning for Wireless Networking Systems (MLWiNS) under Grant 2002821. The work of Avishek Ghosh and Dong Yin was supported by the Ph.D. Students at UC Berkeley. An earlier version of this paper was presented in part at the Annual Conference on Neural Information Processing Systems (NeurIPS), 2020. *(Avishek Ghosh, Jichan Chung, and Dong Yin contributed equally to this work.) (Corresponding authors: Avishek Ghosh; Dong Yin.)*

Avishek Ghosh is with the Halicioglu Data Science Institute, UC San Diego, La Jolla, CA 92093 USA (e-mail: a2ghosh@ucsd.edu).

Jichan Chung and Kannan Ramchandran are with the EECS Department, UC Berkeley, Berkeley, CA 94720 USA.

Dong Yin is with DeepMind, Mountain View, CA 94043 USA (e-mail: yindong10@gmail.com).

Communicated by V. Y. F. Tan, Associate Editor for Machine Learning.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TIT.2022.3192506.

Digital Object Identifier 10.1109/TIT.2022.3192506

argue that one of the major advantages of our algorithm is that it does not require a centralized clustering algorithm, and thus significantly reduces the computational cost at the center machine. When the cluster structure is ambiguous, we propose to leverage the weight sharing technique in multi-task learning [10] and combine it with IFCA. More specifically, we learn the shared representation layers using data from all the users, and use IFCA to train separate final layers for each individual cluster.

We further establish convergence rates of our algorithm, for both linear models and general strongly convex losses under the assumption of good initialization. We prove exponential convergence speed, and for both settings, we can obtain *near optimal* statistical error rates in certain regimes. For the linear model that we consider, when there are two clusters, we show that IFCA is guaranteed to converge as long as the initialization is slightly better than random. We also present experimental evidence of its performance in practical settings: We show that our algorithm can succeed even if we relax the initialization requirements with random initialization and multiple restarts; and we also present results showing that our algorithm is efficient on neural networks. We demonstrate the effectiveness of IFCA on two clustered FL benchmarks created based on the MNIST and CIFAR-10 datasets, respectively, as well as the Federated EMNIST dataset [11] which is a more realistic benchmark for FL and has ambiguous cluster structure.[1]

Here, we emphasize that clustered federated learning is not the only approach to modeling the non-i.i.d. nature of the problem, and different algorithms may be more suitable for different application scenarios; see Section II for more discussions. That said, our approach to modeling and the resulting IFCA framework is certainly an important and relatively unexplored direction in federated learning. We would also like to note that our theoretical analysis makes contributions to statistical estimation problems with latent variables in distributed settings. In fact, both mixture of regressions [12] and mixture of classifiers [13] can be considered as special cases of our problem in the centralized setting. We discuss more about these algorithms in Section II.

Notation: We use $[r]$ to denote the set of integers $\{1, 2, \ldots, r\}$. We use $\|\cdot\|$ to denote the ℓ_2 norm of vectors. We use $x \lesssim y$ if there exists a sufficiently large constant $c > 0$ such that $x \leq cy$, and define $x \gtrsim y$ similarly. We use $\text{poly}(m)$ to denote a polynomial in m with arbitrarily large constant degree.

II. Related Work

During the preparation of the initial draft of this paper, we became aware of a concurrent and independent work by [7], in which the authors propose clustered FL as one of the formulations for personalization in federated learning. The algorithms proposed in our paper and by Mansour *et al.* are similar. However, our paper makes an important contribution by establishing the *convergence rate* of the *population loss function* under good initialization, which simultaneously guarantees both convergence of the training loss and generalization to test data; whereas in [7], the authors provided only *generalization* guarantees. We discuss other related work in the following.

[1]Implementation of our experiments is open sourced at https://github.com/jichan3751/ifca

A. Federated Learning and Non-I.I.D. Data

Learning with a distributed computing framework has been studied extensively in various settings [14]–[16]. As mentioned in Section I, federated learning [1]–[3], [17] is one of the modern distributed learning frameworks that aims to better utilize the data and computing power on edge devices. A central problem in FL is that the data on the users' personal devices are usually non-i.i.d. Several formulations and solutions have been proposed to tackle this problem. A line of research focuses on learning a single global model from non-i.i.d. data [18]–[23]. Other lines of research focus more on learning personalized models [4], [5], [24]. In particular, the MOCHA algorithm [4] considers a multi-task learning setting and forms a deterministic optimization problem with the correlation matrix of the users being a regularization term. Our work differs from MOCHA since we consider a statistical setting with cluster structure. Another approach is to formulate federated learning with non-i.i.d. data as a meta learning problem [6], [24], [25]. In this setup, the objective is to first obtain a single global model, and then each device fine-tunes the model using its local data. The underlying assumption of this formulation is that the data distributions among different users are similar, and the global model can serve as a good initialization.

B. Clustered Federated Learning

The formulation of clustered FL has been considered in a few recent works [5], [7], [26]. In comparison with some prior works, e.g., [5], our algorithm does not require a centralized clustering procedure, and thus significantly reduces the computational cost at the center machine. Also, as explained in the beginning of this section, the concurrent work of Mansour *et al.* [7] do not provide any convergence guarantees for their clustering algorithm. Furthermore, [27], [28] use ensemble methods (mixture of experts) and empirically obtain guarantees for clustering and personalization in Federated Learning. However, to the best of our knowledge, our work is the first in the literature that rigorously characterize the convergence behavior and statistical optimality for clustered FL problems.

C. Latent Variable Problems

As mentioned in Section I, our formulation can be considered as a statistical estimation problem with latent variables in a distributed setting, and the latent variables are the cluster identities. Latent variable problem is a classical topic in statistics and non-convex optimization; examples include Gaussian mixture models (GMM) [29], [30], mixture of linear regressions [12], [31], [32], and phase retrieval [33], [34]. Expectation maximization (EM) and alternating

minimization (AM) are two popular approaches to solving these problems. Despite the wide applications, their convergence analyses in the finite sample setting are known to be hard, due to the non-convexity nature of their optimization landscape. In recent years, some progress has been made towards understanding the convergence of EM and AM in the centralized setting [35]–[39]. For example, if started from a suitable point, they have fast convergence rate, and occasionally they enjoy super-linear speed of convergence [29], [40]. In this paper, we provide new insights to these algorithms in the FL setting.

III. Problem Formulation

We begin with a standard statistical learning setting of empirical risk minimization (ERM). Our goal is to learn parametric models by minimizing some loss functions defined by the data. We consider a distributed learning setting where we have one center machine and m worker machines (i.e., each worker machine corresponds to a user in the federated learning framework). The center machine and worker machines can communicate with each other using some predefined communication protocol. We assume that there are k different data distributions, $\mathcal{D}_1, \ldots, \mathcal{D}_k$, and that the m machines are partitioned into k disjoint clusters, $S_1^*, \ldots, S_k^*$. We assume no knowledge of the cluster identity of each machine, i.e., the partition $S_1^*, \ldots, S_k^*$ is not revealed to the learning algorithm. We assume that every worker machine $i \in S_j^*$ contains n i.i.d. data points $z^{i,1}, \ldots, z^{i,n}$ drawn from $\mathcal{D}_j$, where each data point $z^{i,\ell}$ consists of a pair of feature and response denoted by $z^{i,\ell} = (x^{i,\ell}, y^{i,\ell})$.

Let $f(\vartheta; z) : \Theta \to \mathbb{R}$ be the loss function associated with data point z, where $\Theta \subseteq \mathbb{R}^d$ is the parameter space. In this paper, we choose $\Theta = \mathbb{R}^d$. Our goal is to minimize the population loss function

$$F^j(\vartheta) := \mathbb{E}_{z \sim \mathcal{D}_j}[f(\vartheta; z)],$$

for all $j \in [k]$. For the purpose of theoretical analysis in Section V, we focus on the strongly convex losses, in which case we can prove guarantees for estimating the unique solution that minimizes each population loss function. In particular, we try to find solutions $\{\hat{\vartheta}_j\}_{j=1}^k$ that are close to

$$\vartheta_j^* = \operatorname{argmin}_{\vartheta \in \Theta} F^j(\vartheta), \quad j \in [k].$$

Since we only have access to finite data, we use empirical loss functions. In particular, let $Z \subseteq \{z^{i,1}, \ldots, z^{i,n}\}$ be a subset of the data points on the i-th machine. We define the empirical loss associated with Z as

$$F_i(\vartheta; Z) = \frac{1}{|Z|} \sum_{z \in Z} f(\vartheta; z).$$

When it is clear from the context, we may also use the shorthand notation $F_i(\vartheta)$ to denote an empirical loss associated with some (or all) data on the i-th worker.

IV. Algorithm

Since the users are partitioned into clusters, a natural idea is to use some off-the-shelf clustering approaches such as the Lloyd's algorithm (k-means) [41]. In this section, we first present a straightforward one-shot clustering algorithm that aims to estimate the cluster identities within one round of clustering at the center machine, and discuss the potential drawbacks of this method in the real-world scenario. Then in Section IV-B, we propose an iterative algorithm that alternates between estimating the cluster identities and minimizing the loss functions.

A. One-Shot Clustering

In our problem formulation, the data distribution on all the user devices has a cluster structure, a natural idea is to run an off-the-shelf clustering algorithm such as k-means to estimate the cluster identities. Since the data are all stored in the worker machines, it is not plausible to upload all the data to the center machine and cluster the raw data points due to the large communication cost. Instead, one straightforward approach is to let each worker machine train a model (ERM solution) $\hat{\vartheta}_i$ with its local data, send it to the center machine, and the center machine estimates the cluster identities of each worker machine by running k-means on $\hat{\vartheta}_i$'s. With the cluster identity estimations, the center machine runs any federated learning algorithm separately within each cluster, such as gradient-based methods [1], the Federated Averaging (FedAvg) algorithm [3], or second-order methods [42], [43]. This approach is summarized in Algorithm 1.

Algorithm 1 One-Shot Clustering Algorithm for Federated Learning

1: Worker machines send ERM $\hat{\vartheta}_i := \operatorname{argmin}_{\vartheta \in \Theta} F_i(\vartheta)$ (for all $i \in [m]$) to the center.
2: Center machine clusters $\{\hat{\vartheta}_i\}_{i=1}^m$ into $S_1, \ldots, S_k$.
3: Within each cluster S_j, $j \in [k]$, run federated learning algorithm to obtain final solution $\hat{\vartheta}_j$.

Note that Algorithm 1 is a modular algorithm. It provides flexibility in terms of the choice of clustering and optimization subroutines dependent on the problem instance. However, a major drawback is that Algorithm 1 clusters the user machines only once, and retains that throughout learning. Thus, if the clustering algorithm produces incorrect identity estimations, there is no chance that the algorithm can correct them in the last stage. Moreover, in Algorithm 1 the clustering is done in the center machine. If m is large, this job can be computationally heavy. Note that this is not desired in the setup of FL whose key idea is to reduce the computational cost of the center machine and leverage the end users' computing power (see e.g. [1]). Another note is that for most settings, in the first stage, i.e., computing the local ERMs, the worker machines can only use convex loss functions, rather than more complex models such as neural networks. The reason is that the ERM solutions $\hat{\vartheta}_i$ are used in distance-based clustering algorithm such as k-means, but for neural networks, two similar models can have parameters that are far apart due to, e.g., the permutation invariance of the model to the hidden units.

In order to address the aforementioned concerns, in the following sections, we propose and analyze an iterative clus-

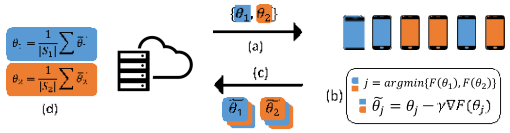

Fig. 1. An overview of IFCA (model averaging). (a) The server broadcast models. (b) Worker machines identify their cluster memberships and run local updates. (c) The worker machines send back the local models to server. (d) Average the models within the same estimated cluster S_j.

tering algorithm, where we improve the clustering of worker machines over multiple iterations.

B. Iterative Federated Clustering Algorithm (IFCA)

We now discuss details of our main algorithm, named *Iterative Federated Clustering Algorithm* (IFCA). The key idea is to alternatively minimize the loss functions while estimating the cluster identities. We discuss two variations of IFCA, namely gradient averaging and model averaging. The algorithm is formally presented in Algorithm 2 and illustrated in Figure 1.

Algorithm 2 Iterative Federated Clustering Algorithm (IFCA)

Input: number of clusters k, step size γ, $j \in [k]$, initialization $\vartheta_j^{(0)}$, $j \in [k]$
number of parallel iterations T, number of local gradient steps τ (for model averaging).
for $t = 0, 1, \ldots, T-1$ **do**
  center machine: broadcast $\vartheta_j^{(t)}$, $j \in [k]$
  $M_t \leftarrow$ random subset of worker machines (participating devices)
  for worker machine $i \in M_t$ in parallel **do**
    cluster identity estimate $\hat{j} = \operatorname{argmin}_{j \in [k]} F_i(\vartheta_j^{(t)})$
    define one-hot encoding vector $s_i = \{s_{i,j}\}_{j=1}^k$ with $s_{i,j} = \mathbf{1}\{j = \hat{j}\}$
    option I (gradient averaging):
      compute (stochastic) gradient: $g_i = \nabla F_i(\vartheta_j^{(t)})$, send s_i, g_i to center machine
    option II (model averaging):
      $\theta_i = \mathsf{LocalUpdate}(\vartheta_j^{(t)}, \gamma, \tau)$, send s_i, θ_i to center machine
  end for
  center machine:
  option I (gradient averaging): $\vartheta_j^{(t+1)} = \vartheta_j^{(t)} - \frac{\gamma}{m} \sum_{i \in M_t} s_{i,j} g_i$, $j \in [k]$
  option II (model averaging): $\vartheta_j^{(t+1)} = \sum_{i \in M_t} s_{i,j} \theta_i / \sum_{i \in M_t} s_{i,j}$, $j \in [k]$
end for
return $\vartheta_j^{(T)}$, $j \in [k]$
$\mathsf{LocalUpdate}(\theta^{(0)}, \gamma, \tau)$ at the i-th worker machine
for $q = 0, \ldots, \tau - 1$ **do**
  (stochastic) gradient descent $\theta^{(q+1)} = \theta^{(q)} - \gamma \nabla F_i(\theta^{(q)})$
end for
return $\theta^{(\tau)}$

The algorithm starts with k initial model parameters $\vartheta_j^{(0)}$, $j \in [k]$. In the t-th iteration of IFCA, the center machine selects a random subset of worker machines, $M_t \subseteq [m]$, and broadcasts the current model parameters $\{\vartheta_j^{(t)}\}_{j=1}^k$ to the worker machines in M_t. Here, we call M_t the set of *participating devices*. Recall that each worker machine is equipped with local empirical loss function $F_i(\cdot)$. Using the received parameter estimates and F_i, the i-th worker machine ($i \in M_t$) estimates its cluster identity via finding the model parameter with lowest loss, i.e., $\hat{j} = \operatorname{argmin}_{j \in [k]} F_i(\vartheta_j^{(t)})$ (ties can be broken arbitrarily). If we choose the option of gradient averaging, the worker machine then computes a (stochastic) gradient of the local empirical loss F_i at $\vartheta_j^{(t)}$, and sends its cluster identity estimate and gradient back to the center machine. After receiving the gradients and cluster identity estimates from all the participating worker machines, the center machine then collects all the gradient updates from worker machines whose cluster identity estimates are the same and conducts gradient descent update on the model parameter of the corresponding cluster. If we choose the option of model averaging (similar to FedAvg [3]), each participating device needs to run τ steps of local (stochastic) gradient descent updates, get the updated model, and send the new model and its cluster identity estimate to the center machine. The center machine then averages the new models from the worker machines whose cluster identity estimates are the same.

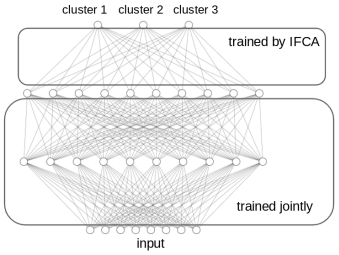

Fig. 2. Weight sharing scheme for IFCA with neural networks.

C. Practical Implementation of IFCA

We clarify a few issues regarding the practical implementation of IFCA. In some practical problems, the cluster structure may be ambiguous, which means that although the distributions of data from different clusters are different, there exists some common properties of the data from all the users that the model should leverage. For these problems, we propose to use the weight sharing technique in multi-task learning [10] and combine it with IFCA. More specifically, when we train neural network models, we can share the weights for the first a few layers among all the clusters so that we can learn a good representation using all the available data, and then run IFCA algorithm only on the last (or last few) layers to address the different distributions among different clusters. Using the notation in Algorithm 2, we run IFCA on a subset of the coordinates of $\vartheta_j^{(t)}$, and run vanilla gradient or model averaging on the remaining coordinates. Another benefit of this implementation is that we can reduce the communication cost: Instead of sending k models to all the worker machines, the center machine only needs to send k different versions of a subset of all the weights, and one single copy of the shared layers. We illustrate the weight sharing method in Figure 2.

Another technique to reduce communication cost is that when the center machine observes that the cluster identities of all the worker machines are stable, i.e., the estimates of their cluster identities do not change for several parallel iterations, then the center machine can stop sending k models to each worker machine, and instead, it can simply send the model corresponding to each worker machine's cluster identity estimate.

V. Theoretical Guarantees

In this section, we present convergence guarantees of IFCA. In order to streamline our theoretical analysis, we make several simplifications: we consider the IFCA with gradient averaging, and assume that all the worker machines participate in every rounds of IFCA, i.e., $M_t = [m]$ for all t. In addition, we also use the *re-sampling* technique for the purpose of theoretical analysis. In particular, suppose that we run a total of T parallel iterations. We partition the n data points on each machine into $2T$ disjoint subsets, each with $n' = \frac{n}{2T}$ data points. For the i-th machine, we denote the subsets as $Z_i^{(0)}, \ldots, Z_i^{(T-1)}$ and $Z_i^{(0)}, \ldots, Z_i^{(T-1)}$. In the t-th iteration, we use $Z_i^{(t)}$ to estimate the cluster identity, and use $Z_i^{(t)}$ to conduct gradient descent. As we can see, we use fresh data samples for each iteration of the algorithm. Furthermore, in each iteration, we use different set of data points for obtaining the cluster estimate and computing the gradient. This is done in order to remove the inter-dependence between the cluster estimation and the gradient computation, and ensure that in each iteration, we use fresh i.i.d. data that are independent of the current model parameter. We would like to emphasize that re-sampling is a standard tool used in statistics [35], [37], [40], [44], [45], and that it is for theoretical tractability only and is not required in practice as we show in Section VI.

Under these conditions, the update rule for the parameter vector of the j-th cluster can be written as

$$S_j^{(t)} = \{i \in [m] : j = \operatorname{argmin}_{j' \in [k]} F_i(\theta_{j'}^{(t)}; Z_i^{(t)})\},$$
$$\theta_j^{(t+1)} = \theta_j^{(t)} - \frac{\gamma}{m} \sum_{i \in S_j^{(t)}} \nabla F_i(\theta_j^{(t)}; Z_i^{(t)}),$$

where $S_j^{(t)}$ denotes the set of worker machines whose cluster identity estimate is j in the t-th iteration. In the following, we discuss the convergence guarantee of IFCA under two models: in Section V-A, we analyze the algorithm under a linear model with Gaussian features and squared loss, and in Section V-B, we analyze the algorithm under a more general setting of strongly convex loss functions. In all of our results, c, c_1, c_2, c_3, ... denote universal constants.

A. Linear Models With Squared Loss

In this section, we analyze our algorithm in a concrete linear model. This model can be seen as a warm-up example for more general problems with strongly convex loss functions that we discuss in Section V-B, as well as a distributed formulation of the widely studied mixture of linear regression problem [37], [45]. We assume that the data on the worker machines in the j-th cluster are generated in the following way: for $i \in S_j^*$, the feature-response pair of the i-th worker machine machine satisfies

$$y^{i,\ell} = \langle x^{i,\ell}, \theta_j^* \rangle + \varepsilon^{i,\ell},$$

where $x^{i,\ell} \in N(0, I_d)$ and the additive noise $\varepsilon^{i,\ell} \in N(0, \sigma^2)$ is independent of $x^{i,\ell}$. Furthermore, we use the squared loss function $f(\theta; x, y) = (y - \langle x, \theta \rangle)^2$. As we can see, this model is the mixture of linear regression model in the distributed setting. We observe that under the above setting, the parameters $\{\theta_j^*\}_{j=1}^k$ are the minimizers of the population loss function $F^i(\cdot)$.

We proceed to analyze our algorithm. We define $p_j := |S_j^*|/m$ as the fraction of worker machines belonging to the j-th cluster, and let $p := \min\{p_1, p_2, \ldots, p_k\}$. We also define the minimum separation Δ as $\Delta := \min_{j \neq j'} \|\theta_j^* - \theta_{j'}^*\|$, and $\rho := \frac{\Delta^2}{\sigma^2}$ as the signal-to-noise ratio. Before we establish our convergence result, we state a few assumptions. Here, recall that n' denotes the number of data that each worker uses in each step.

Assumption 1: The initialization of parameters $\theta_j^{(0)}$ satisfy $\|\theta_j^{(0)} - \theta_j^*\| \leq (\frac{1}{2} - \alpha_0)\Delta$, $\forall j \in [k]$, where the closeness parameter α_0 satisfies $0 < \alpha_0 < \frac{1}{2}$.

Assumption 2: Without loss of generality, we assume that $\max_{j \in [k]} \|\theta_j^*\| \geq 1$, and that $\sigma \geq 1$. We also assume that $n' \gtrsim (\frac{\rho+1}{\alpha_0 \rho})^2 \log m$, $d \gtrsim \log m$, $p \gtrsim \frac{\log m}{m}$, $pmn' \gtrsim d$, and $\Delta \gtrsim \frac{\sigma}{p}\sqrt{\frac{d}{mn'}} + \exp(-c(\frac{\alpha_0 \rho}{\rho+1})^2 n')$ for some universal constant c.

Assumption 1 states that the initialization condition is characterized by the closeness parameter α_0. If α_0 is close to $\frac{1}{2}$, the initialization is very close enough to θ_j^*. On the other hand, if α_0 is very small, Assumption 1 implies that the initial point is only *slightly* biased towards θ_j^*. We note that having an assumption on the initialization is standard in the convergence analysis of mixture models [38], [46], due to the non-convex optimization landscape of mixture model problems. Moreoever, when $k = 2$, and α_0 is small, this assumption means we only require the initialization to be slightly better than random. In Assumption 2, we put mild assumptions on n', m, p, and d. The condition that $pmn' \gtrsim d$ simply assumes that the total number of data that we use in each iteration for each cluster is at least as large as the dimension of the parameter space. The condition that $\Delta \gtrsim \frac{\sigma}{p}\sqrt{\frac{d}{mn'}} + \exp(-c(\frac{\alpha_0 \rho}{\rho+1})^2 n')$ ensures that the iterates stay close to θ_j^*.

We first provide a single step analysis of our algorithm. We assume that at a certain iteration, we obtain parameter vectors θ_j that are close to the ground truth parameters θ_j^*, and show that θ_j converges to θ_j^* at an exponential rate with an error floor.

Theorem 1: Consider the linear model and assume that Assumptions 1 and 2 hold. Suppose that in a certain iteration of the IFCA algorithm we obtain parameter vectors θ_j with

$$\|\theta_j - \theta_j^*\| \leq (\frac{1}{2} - \alpha)\Delta, \quad 0 < \alpha < \frac{1}{2}.$$

Let ϑ_j^+ be iterate after this iteration. Then, when we choose step size $\gamma = c_1/p$, with probability at least $1 - 1/\text{poly}(m)$, we have for all $j \in [k]$,

$$\|\vartheta_j^+ - \vartheta_j^*\| \le \frac{1}{2}\|\vartheta_j - \vartheta_j^*\| + c_2\frac{\sigma}{p}\sqrt{\frac{d}{mn^0}} + c_3\exp\left(-c_4\left(\frac{\alpha\rho}{\rho+1}\right)^2 n^0\right). \tag{1}$$

We prove Theorem 1 in Appendix IX. We see that provided that Assumptions 1 and 2 hold, the iterates of IFCA are contractive. However, the contraction depends on the closeness parameter α. From the third terms on the right hand side of (1), a smaller α implies slower convergence. If IFCA starts with a very small α_0, the convergence can be slow at the beginning. However, as we run more iterations, the closeness parameter α increases, and thus the algorithm gradually converges faster. We can also see from the proof in Appendix IX that the third term in (1) corresponds to the misclassification probability when estimating the cluster identities of the worker machines. Thus, an increasing sequence of α implies the improvement of the accuracy of cluster identity estimation.

With this theorem, we can now state the convergence guarantee of the entire algorithm.

Corollary 1: Consider the linear model and assume that Assumptions 1 and 2 hold. By choosing step size $\gamma = c_1/p$, with probability at least $1 - \frac{2+\log(\frac{\Delta}{4\varepsilon})}{\text{poly}(m)}$, after $T = 2 + \log\frac{\Delta}{4\varepsilon}$ parallel iterations, we have for all $j \in [k]$, $\|\vartheta_j - \vartheta_j^*\| \le \varepsilon$, where

$$\varepsilon = c_6\frac{\sigma}{p}\sqrt{\frac{d_0}{mn}} + c_7\exp\left(-c_8\left(\frac{\rho}{\rho+1}\right)^2 n^0\right).$$

We prove Corollary 1 in Appendix X. The basic proof idea is as follows: Although the initial closeness parameter α_0 can be small, as mentioned, it increases while we run the algorithm, and we can show that after a small number of iterations, denoted by T^0 in the proof, the closeness parameter can increase to a fixed constant, say 1/4. Afterwards, the third term in (1) does not explicitly depend on the initial closeness parameter α_0. Then by iterating (1) we obtain the final result, whose error floor ε does not explicitly depend on α.

Let us examine the final accuracy. Since the number of data points on each worker machine $n = 2n^0T = 2n^0\log(\Delta/4\varepsilon)$, we know that for the smallest cluster, there are a total of $2pmn^0\log(\Delta/4\varepsilon)$ data points. According to the minimax estimation rate of linear regression [47], we know that *even if we know the ground truth cluster identities*, we cannot obtain an error rate better than $O(\sigma\sqrt{\frac{d}{pmn^0\log(\Delta/4\varepsilon)}})$. Comparing this rate with our statistical accuracy ε, we can see that the first term $\frac{\sigma}{p}\sqrt{\frac{d}{mn^0}}$ in ε is equivalent to the minimax rate up to a logarithmic factor and a dependency on p, and the second term in ε decays exponentially fast in n^0, and therefore, our final statistical error rate is *near optimal*.

B. Strongly Convex Loss Functions

In this section, we study a more general scenario where the population loss functions of the k clusters are strongly convex and smooth. In contrast to the previous section, our analysis do not rely on any specific statistical model, and thus can be applied to more general machine learning problems. We start with reviewing the standard definitions of strongly convex and smooth functions $F : \mathbb{R}^d \to \mathbb{R}$.

Definition 1: F is λ-strongly convex if $\forall\vartheta, \vartheta^0$, $F(\vartheta^0) \ge F(\vartheta) + \langle\nabla F(\vartheta), \vartheta^0 - \vartheta\rangle + \frac{\lambda}{2}\|\vartheta^0 - \vartheta\|^2$.

Definition 2: F is L-smooth if $\forall\vartheta, \vartheta^0$, $\|\nabla F(\vartheta) - \nabla F(\vartheta^0)\| \le L\|\vartheta - \vartheta^0\|$.

In this section, we assume that the population loss functions $F^j(\vartheta)$ are strongly convex and smooth.

Assumption 3: The population loss function $F^j(\vartheta)$ is λ-strongly convex and L-smooth, $\forall j \in [k]$.

We note that we do not make any convexity or smoothness assumptions on the individual loss function $f(\vartheta; z)$. Instead, we make the following distributional assumptions on $f(\vartheta; z)$ and $\nabla f(\vartheta; z)$.

Assumption 4: For every ϑ and every $j \in [k]$, the variance of $f(\vartheta; z)$ is upper bounded by η^2, when z is sampled according to D_j, i.e., $\mathbb{E}_{z\sim D_j}[(f(\vartheta; z) - F^j(\vartheta))^2] \le \eta^2$

Assumption 5: For every ϑ and every $j \in [k]$, the variance of $\nabla f(\vartheta; z)$ is upper bounded by ν^2, when z is sampled according to D_j, i.e., $\mathbb{E}_{z\sim D_j}[\|\nabla f(\vartheta; z) - \nabla F^j(\vartheta)\|^2] \le \nu^2$

Bounded variance of gradient is very common in analyzing SGD [48]. In this paper we use loss function value to determine cluster identity, so we also need to have a probabilistic assumption on $f(\vartheta; z)$. We note that bounded variance is a relatively weak assumption on the tail behavior of probability distributions. In addition to the assumptions above, we still use some definitions from Section V-A, i.e., $\Delta := \min_{j\neq j'}\|\vartheta_j^* - \vartheta_{j'}^*\|$ and $p = \min_{j\in[k]} p_j$ with $p_j = |S_j^*|/m$. We make the following assumptions on the initialization, n^0, p, and Δ.

Assumption 6: Without loss of generality, we assume that $\max_{j\in[k]}\|\vartheta_j^{(0)} - \vartheta_j^*\| < 1$. We also assume that $\|\vartheta_j^{(0)} - \vartheta_j^*\| \le (\frac{1}{2} - \alpha_0)\sqrt{\frac{\lambda}{L}}\Delta$, $\forall j \in [k]$, $n^0 \gtrsim \frac{k\eta^2}{\alpha_0^2\lambda^2\Delta^4}$, $p \gtrsim \frac{\log(mn^0)}{m}$, and that

$$\Delta \gtrsim \mathcal{O}(\max\{\alpha_0^{-2/5}(n^0)^{-1/5}, \alpha_0^{-1/3}m^{-1/6}(n^0)^{-1/3}\}).$$

Here, for simplicity, the $\mathcal{O}$ notation omits any logarithmic factors and quantities that do not depend on m and n^0. As we can see, again we need to assume good initialization, due to the nature of the mixture model, and the assumptions that we impose on n^0, p, and Δ are relatively mild; in particular, the assumption on Δ ensures that the iterates stay close to an ℓ_2 ball around ϑ_j^*. Similar to Section V-A, we begin with the guarantee for a single iteration.

Theorem 2: Suppose Assumptions 3-6 hold. Choose step size $\gamma = 1/L$. Suppose that in a certain iteration of IFCA algorithm, we obtain the parameter vectors ϑ_j with $\|\vartheta_j - \vartheta_j^*\| \le (\frac{1}{2} - \alpha)\sqrt{\frac{\lambda}{L}}\Delta$ with $0 < \alpha < \frac{1}{2}$. Let ϑ_j^+ be the next iterate in the algorithm. Then, for $\delta \in (0, 1)$, we have for any fixed $j \in [k]$, with probability at least $1 - \delta$,

$$\|\vartheta_j^+ - \vartheta_j^*\| \le (1 - \frac{p\lambda}{8L})\|\vartheta_j - \vartheta_j^*\| + \varepsilon_0,$$

where

$$\varepsilon_0 \asymp \frac{\nu}{\delta L\sqrt{pmn_0}} + \frac{\eta^2}{\delta\alpha^2\lambda^2\Delta^4 n^0} + \frac{\nu\eta k^{3/2}}{\delta^{3/2}\alpha\lambda L\Delta^2\sqrt{mn_0}}.$$

We prove Theorem 2 in Appendix XI. With this theorem, we can then provide the convergence guarantee of the entire algorithm.

Corollary 2: Suppose Assumptions 3-6 hold. Choose step size $\gamma = 1/L$. Then, with probability at least $1 - \delta$, after $T = \frac{8L}{p\lambda} \log \frac{2\Delta}{\varepsilon}$ parallel iterations, we have for all $j \in [k]$, $\|\theta_j - \theta_j^*\| \le \varepsilon$, where

$$\varepsilon \asymp \frac{\nu k L \log(mn^0)}{p^{5/2}\lambda^2\delta\sqrt{mn^0}} + \frac{\eta^2 L^2 k \log(mn^0)}{p^2\lambda^4\delta\Delta^4 n^0} + \mathcal{O}\left(\frac{1}{n^0\sqrt{m}}\right).$$

We prove Corollary 2 in Appendix XII. The basic proof idea is to first show that after $T^0 = \frac{8L}{p\lambda}\log(4\sqrt{L/\lambda})$, the closeness parameter α_0 grows from α_0 to at least $1/4$. Then after another $T - T^0$ parallel iterations, the algorithm converges to the desired accuracy ε. Note that this ensures that there is no explicit dependence on the initial closeness parameter α_0 in ε.

To better interpret the result, we focus on the dependency on m and n and treat other quantities as constants. Then, since $n = 2n^0T$, we know that n and n^0 are of the same scale up to a logarithmic factor. Therefore, the final statistical error rate that we obtain is $\mathcal{O}(\nu\sqrt{\frac{1}{mn}} + \frac{1}{n})$. As discussed in Section V-A, $\nu\sqrt{\frac{1}{mn}}$ is the optimal rate even if we know the cluster identities; thus our statistical rate is near optimal in the regime where $n \gtrsim m$. In comparison with the statistical rate in linear models $\mathcal{O}(\nu\sqrt{\frac{1}{mn}} + \exp(-n))$, we note that the major difference is in the second term. The additional terms of the linear model and the strongly convex case are $\exp(-n)$ and $\frac{1}{n}$, respectively. We note that this is due to different statistical assumptions: in for the linear model, we assume Gaussian noise whereas here we only assume bounded variance.

VI. Experiments

In this section, we present our experimental results, which not only validate the theoretical claims in Section V, but also demonstrate that our algorithm can be efficiently applied beyond the regime we discussed in the theory. We emphasize that we do not re-sample fresh data points at each iteration in our empirical study. Furthermore, the requirement on the initialization can be relaxed. More specifically, for linear models, we observe that random initialization with a few restarts is sufficient to ensure convergence of Algorithm 2. In our experiments, we also show that our algorithm works efficiently for problems with non-convex loss functions such as neural networks.

A. Synthetic Data

We begin with evaluation of Algorithm 2 with gradient averaging (option I) on linear models with squared loss, as described in Section V-A. For all $j \in [k]$, we first generate $\theta_j^* \in$ Bernoulli(0.5) coordinate-wise, and then rescale their ℓ_2 norm to R. This ensures that the separation between the θ_j^*'s is proportional to R in expectation, and thus, in this experiment, we use R to represent the *separation* between the ground truth parameter vectors. Moreover, we simulate the scenario where all the worker machines participate in all iterations, and all the clusters contain same number of worker machines.

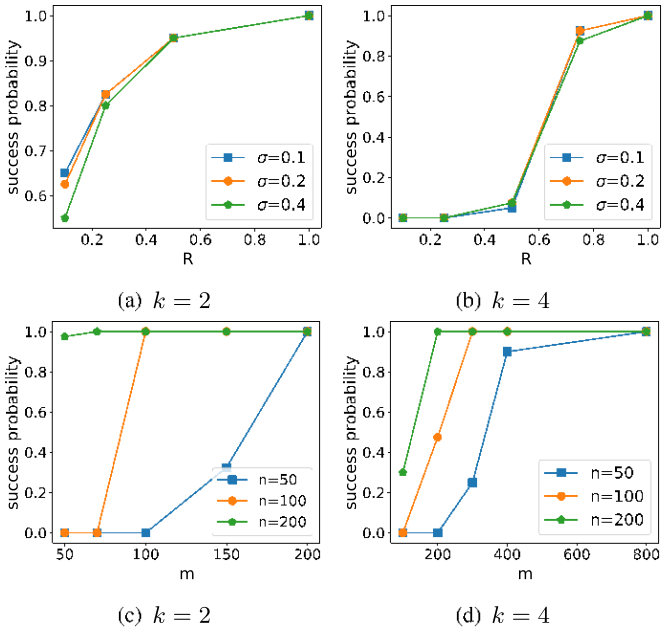

Fig. 3. Success probability with respect to: (a), (b) the separation scale R and the scale of additive noise σ; (c), (d) the number of worker machines m and the sample size on each machine n. In (a) and (b), we see that the success probability gets better with increasing R, i.e., more separation between ground truth parameter vectors, and in (c) and (d), we note that the success probability improves with an increase of mn, i.e., more data on each machine and/or more machines.

For each trial of the experiment, we first generate the parameter vectors θ_j^*'s, fix them, and then randomly initialize $\theta_j^{(0)}$ according to an independent coordinate-wise Bernoulli distribution. We then run Algorithm 2 for 300 iterations, with a constant step size. For $k = 2$ and $k = 4$, we choose the step size in $\{0.01, 0.1, 1\}, \{0.5, 1.0, 2.0\}$, respectively. In order to determine whether we successfully learn the model or not, we sweep over the aforementioned step sizes and define the following distance metric: $\text{dist} = \frac{1}{k}\sum_{j=1}^{k}\|\theta_j - \theta_j^*\|$, where $\{\theta_j\}_{j=1}^k$ are the parameter estimates obtained from Algorithm 2. A trial is dubbed *successful* if for a fixed set of θ_j^*, among 10 random initialization of $\theta_j^{(0)}$, at least in one scenario, we obtain $\text{dist} \le 0.6\sigma$.

In Fig. 3 (a-b), we plot the empirical success probability over 40 trials, with respect to the separation parame-ter R. We set the problem parameters as (a) $(m, n, d) = (100, 100, 1000)$ with $k = 2$, and (b) $(m, n, d) = (400, 100, 1000)$ with $k = 4$. As we can see, when R becomes larger, i.e., the separation between parameters increases, and the problem becomes easier to solve, yielding in a higher success probability. This validates our theoretical result that higher signal-to-noise ratio produces smaller error floor. In Fig. 3 (c-d), we characterize the dependence on m and n, with fixing R and d with $(R, d) = (0.1, 1000)$ for (c) and $(R, d) = (0.5, 1000)$ for (d). We observe that when we increase m and/or n, the success probability improves. This validates our theoretical finding that more data and/or more worker machines help improve the performance of the algorithm.

TABLE I
TEST ACCURACIES(%) ± STD ON ROTATED MNIST ($k = 4$) AND ROTATED CIFAR ($k = 2$)

	Rotated MNIST			Rotated CIFAR
m, n	4800, 50	2400, 100	1200, 200	200, 500
IFCA (ours)	**94.20 ± 0.03**	**95.05 ± 0.02**	**95.25 ± 0.40**	**81.51 ± 1.37**
global model	86.74 ± 0.04	88.65 ± 0.08	89.73 ± 0.13	77.87 ± 0.39
local model	63.32 ± 0.02	73.66 ± 0.04	80.05 ± 0.02	33.97 ± 1.19

B. Rotated MNIST and CIFAR

We also create clustered FL datasets based on the MNIST [49] and CIFAR-10 [50] datasets. In order to simulate an environment where the data on different worker machines are generated from different distributions, we augment the datasets using rotation, and create the Rotated MNIST [51] and Rotated CIFAR datasets. For **Rotated MNIST**, recall that the MNIST dataset has 60000 training images and 10000 test images with 10 classes. We first augment the dataset by applying 0, 90, 180, 270 degrees of rotation to the images, resulting in $k = 4$ clusters. For given m and n satisfying $mn = 60000k$, we randomly partition the images into m worker machines so that each machine holds n images *with the same rotation*. We also split the test data into $m_{\text{test}} = 10000k/n$ worker machines in the same way. The **Rotated CIFAR** dataset is also created in a similar way as Rotated MNIST, with the main difference being that we create $k = 2$ clusters with 0 and 180 degrees of rotation. We note that creating different tasks by manipulating standard datasets such as MNIST and CIFAR-10 has been widely adopted in the continual learning research community [51]–[53]. For clustered FL, creating datasets using rotation helps us simulate a federated learning setup with clear cluster structure.

For our MNIST experiments, we use the fully connected neural network with ReLU activations, with a single hidden layer of size 200; and for our CIFAR experiments, we use a convolution neural network model which consists of 2 convolutional layers followed by 2 fully connected layers, and the images are preprocessed by standard data augmentation such as flipping and random cropping.

We compare our IFCA algorithm with two baseline algorithms, i.e., the *global model*, and *local model* schemes. For **IFCA**, we use model averaging (option II in Algorithm 2). For MNIST experiments, we use full worker machines participation ($M_t = [m]$ for all t). For `LocalUpdate` step in Algorithm 2, we choose $\tau = 10$ and step size $\gamma = 0.1$. For CIFAR experiments, we choose $|M_t| = 0.1m$, and apply step size decay 0.99, and we also set $\tau = 5$ and batch size 50 for `LocalUpdate` process, following prior works [1]. In the **global model** scheme, the algorithm tries to learn single global model that can make predictions from all the distributions. The algorithm does not consider cluster identities, so model averaging step in Algorithm 1 becomes $\theta^{(t+1)} = \sum_{i \in M_t} \theta_i / |M_t|$, i.e. averaged over parameters from all the participating machines. In the **local model** scheme, the model in each node performs gradient descent only on local data available, and model averaging is not performed.

For IFCA and the global model scheme, we perform inference in the following way. For every test worker machine, we run inference on all learned models (k models for IFCA and one model for global model scheme), and calculate the accuracy from the model that produces the smallest loss value. For testing the local model baselines, the models are tested by measuring the accuracy on the test data with the same distribution (i.e. those have the same rotation). We report the accuracy averaged over all the models in worker machines. For all algorithms, we run experiment with 5 different random seeds and report the average and standard deviation.

Our experimental results are shown in Table I. We can observe that our algorithm performs better than the two baselines. As we run the IFCA algorithm, we observe that we can gradually find the underlying cluster identities of the worker machines, and after the correct cluster is found, each model is trained and tested using data with the same distribution, resulting in better accuracy. The global model baseline performs worse than ours since it tries to fit all the data from different distributions, and cannot provide personalized predictions. The local model baseline algorithm overfits to the local data easily, leading to worse performance than ours. In Figure 4, we illustrate the accuracy of cluster identity estimation during the algorithm. As we can see, the algorithm identifies all the clusters after a relatively small number of communication rounds (around 30 for Rotated MNIST and 10 for Rotated CIFAR).

C. Federated EMNIST

We provide additional experimental results on the Federated EMNIST (FEMNIST) [11], which is a realistic FL dataset where the data points on every worker machine are the handwritten digits or letters from a specific writer. Although the data distribution among all the users are similar, there might be ambiguous cluster structure since the writing styles of different people may be clustered. We use the weight sharing technique mentioned in Section IV-C. We use a neural network with two convolutional layers, with a max pooling layer after each convolutional layer, followed by two fully connected layers. We share the weights of all the layers, except the last layer which is trained by IFCA. We treat the number of clusters k as a hyper parameter and run the experiments with different values of k. We compare IFCA with the global model and local model approaches, as well as the one-shot centralized clustering algorithm described in Section IV-A. The test accuracies are shown in Table II, with mean and standard deviation computed over 5 independent runs. As we can see, IFCA shows clear advantage over the global model and local model approaches. The results of IFCA and the one-shot algorithm are similar. However, as we emphasized

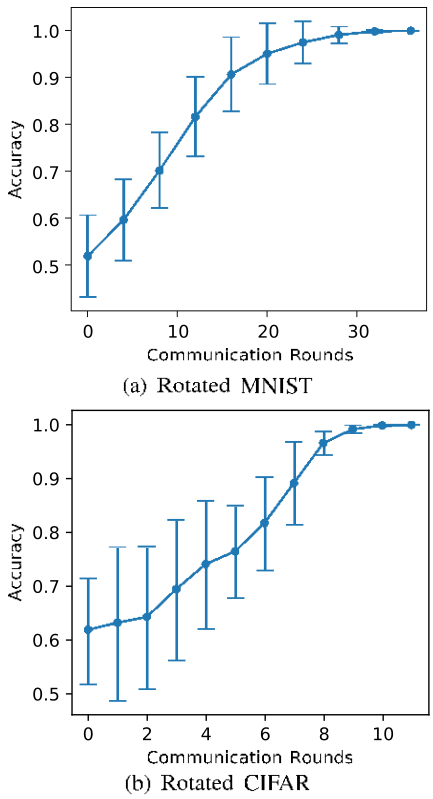

Fig. 4. Accuracy of cluster identity estimation at the end of each parallel iteration (communication round).

in Section IV, IFCA does not run a centralized clustering procedure, and thus reduces the computational cost at the center machine. As a final note, we observe that IFCA is robust to the choice of the number of clusters k. The results of the algorithm with $k = 2$ and $k = 3$ are similar, and we notice that when $k > 3$, IFCA automatically identifies 3 clusters, and the remaining clusters are empty. This indicates the applicability of IFCA in realistic problems where the cluster structure is ambiguous and the number of clusters is unknown.

VII. Differences and Improvements From [54]

A subset of the results in this paper have been published in the Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS, 2020) as a poster with the same title (see [54]). In this short (9 page) conference paper, we established the basic problem formulation for clustered federated learning, and proposed a clustering algorithm namely *Iterative Federated Clustering Algorithm* (IFCA). Theoretical analysis of IFCA was presented only when the initialization of parameter estimates is sufficiently close to the truth (warm start). Although we mention some practical implementation aspects of IFCA in the conference paper, we did not expand on that.

On the other hand, in this submission we have made several significant contributions towards strengthening [54]. First, we begin our discussion with a simple *one-shot* clustering algorithm for federated learning that uses the empirical risk minimizers (ERM) of the worker machines to cluster them. We discuss the potential drawbacks of this approach, which leads to the design of IFCA. We note that the study of the one-shot algorithm has appeared in our another previous paper "Robust Federated Learning in a Heterogeneous Environment", which is on arXiv at https://arxiv.org/pdf/1906.06629.pdf, and is presented at ICML Workshop on Security and Privacy of Machine Learning, 2019.

Second, we have provided an improved theoretical analysis on the initialization of IFCA, and in special cases, we show that initialization that is slightly better than a *random guess* is sufficient to ensure convergence. Defining an initialization parameter, we show that as we run more iterations, IFCA gradually converges faster. Furthermore, the accuracy of cluster identity estimation also improves with iterations. We have studied this for both the quadratic and the general (strongly convex) loss functions, and completely characterize the convergence rate and the final statistical accuracy. On the other hand, in [54], we assumed *sufficiently close* initialization, without any control parameter. Our analysis in this submission is more refined, and we analogously modified the requirements on the problem parameters and obtain a complete characterization of IFCA as a function of the initialization parameter, showing a trade-off between initialization and convergence.

Third, we have a more detailed discussion on the practical implementation aspects of IFCA. Via Figure 2 (of the submitted paper), we illustrate the *weight-sharing* scheme (popularly used in multi-task and representation learning) better. In particular, this technique further reduces the communication cost (which is often the bottleneck in federated learning) between the center and the worker machines. We also added Figure 4, which shows how the cluster identity estimation accuracy improves over the iterations.

VIII. Conclusion and Future Work

In this paper, we address the clustered FL problem. We propose an iterative algorithm and obtain convergence guarantees for strongly convex and smooth functions. In experiments, we achieve this via random initialization with multiple restarts, and we show that our algorithm works efficiently beyond the convex regime. Fufure directions include extending the analysis to weakly convex and non-convex functions, stochastic gradients on the worker machines, and small subset of participating devices.

Appendix

In our proofs, we use $c, c_1, c_2, \ldots$ to denote positive universal constants, the value of which may differ across instances. For a matrix A, we write $\|A\|_{op}$ and $\|A\|_F$ as the operator norm and Frobenius norm, respectively. For a set S, we use $S^{\overline{c}}$ to denote the complement of the set.

IX. Proof of Theorem 1

Since we only analyze a single iteration, for simplicity we drop the superscript that indicates the iteration counter. Suppose that at a particular iteration, we have model parameters $\vartheta_j, j \in [k]$, for the k clusters. We denote the *estimation* of

TABLE II
TEST ACCURACIES (%) ± STD ON FEMNIST

IFCA $(k=2)$	IFCA $(k=3)$	one-shot $(k=2)$	one-shot $(k=3)$	global	local
87.99 ± 0.35	87.89 ± 0.52	87.41 ± 0.39	87.38 ± 0.37	84.45 ± 0.51	75.85 ± 0.72

the set of worker machines that belongs to the j-th cluster by S_j, and recall that the true clusters are denoted by S_j^*, $j \in [k]$.

a) Probability of Erroneous Cluster Identity Estimation: We begin with the analysis of the probability of incorrect cluster identity estimation. Suppose that a worker machine i belongs to S_j^*. We define the event E_i^{j,j^0} as the event when the i-th machine is classified to the j^0-th cluster, i.e., $i \in S_{j^0}$. Thus the event that worker i is correctly classified is $E_i^{j,j}$, and we use the shorthand notation $E_i := E_i^{j,j}$. We now provide the following lemma that bounds the probability of E_i^{j,j^0} for $j^0 = j$.

Lemma 1: Suppose that worker machine $i \in S_j^*$. Let $\rho := \frac{\Delta^2}{\sigma^2}$. Then there exist universal constants c_1 and c_2 such that for any $j^0 = j$,

$$\mathsf{P}(E_i^{j,j^0}) \le c_1 \exp\left(-c_2 n^0 \left(\frac{\alpha\rho}{\rho+1}\right)^2\right),$$

and by union bound

$$\mathsf{P}(\overline{E_i}) \le c_1 k \exp\left(-c_2 n^0 \left(\frac{\alpha\rho}{\rho+1}\right)^2\right).$$

We prove Lemma 1 in Appendix IX-A.

Now we proceed to analyze the gradient descent step. Without loss of generality, we only analyze the first cluster. The update rule of ϑ_1 in this iteration can be written as

$$\vartheta_1^+ = \vartheta_1 - \frac{\gamma}{m} \sum_{i \in S_1} \nabla F_i(\vartheta_1; Z_i),$$

where Z_i is the set of the n^0 data points that we use to compute gradient in this iteration on a particular worker machine.

We use the shorthand notation $F_i(\vartheta) := F_i(\vartheta; Z_i)$, and note that $F_i(\vartheta)$ can be written in the matrix form as

$$F_i(\vartheta) = \frac{1}{n^0} \|Y_i - X_i \vartheta\|^2,$$

where we have the feature matrix $X_i \in \mathbb{R}^{n^0 \times d}$ and response vector $Y_i = X_i \vartheta_1^* + \epsilon_i$. According to our model, all the entries of X_i are i.i.d. sampled according to $N(0, 1)$, and $\epsilon_i \sim N(0, \sigma^2 I)$.

We first notice that

$$\|\vartheta_1^+ - \vartheta_1^*\| = \Big\| \underbrace{\vartheta_1 - \vartheta_1^* - \frac{\gamma}{m} \sum_{i \in S_1 \cap S_1^*} \nabla F_i(\vartheta_1)}_{T_1} - \frac{\gamma}{m} \sum_{i \in S_1 \cap S_1^{*c}} \nabla F_i(\vartheta_1) \Big\| \le \|T_1\| + \|T_2\|.$$

We control the two terms separately. Let us first focus on $\|T_1\|$.

b) Bound $\|T_1\|$: To simplify notation, we concatenate all the feature matrices and response vectors of all the worker machines in $S_1 \cap S_1^*$ and get the new feature matrix $X \in \mathbb{R}^{N \times d}$, $Y \in \mathbb{R}^N$ with $Y = X\vartheta_1^* + \epsilon$, where $N := n^0 |S_1 \cap S_1^*|$. It is then easy to verify that

$$\begin{aligned} T_1 &= \left(I - \frac{2\gamma_0}{mn} X^\top X\right)(\vartheta_1 - \vartheta_1^*) + \frac{2\gamma_0}{mn} X^\top \epsilon \\ &= \left(I - \frac{2\gamma_0}{mn} \mathsf{E}[X^\top X]\right)(\vartheta_1 - \vartheta_1^*) \\ &\quad + \frac{2\gamma_0}{mn} \left(\mathsf{E}[X^\top X] - X^\top X\right)(\vartheta_1 - \vartheta_1^*) + \frac{2\gamma_0}{mn} X^\top \epsilon \\ &= \left(1 - \frac{2\gamma N}{mn^c}\right)(\vartheta_1 - \vartheta_1^*) \\ &\quad + \frac{2\gamma_0}{mn} \left(\mathsf{E}[X^\top X] - X^\top X\right)(\vartheta_1 - \vartheta_1^*) + \frac{2\gamma_0}{mn} X^\top \epsilon. \end{aligned}$$

Therefore

$$\begin{aligned} \|T_1\| \le{}& \left(1 - \frac{2\gamma N}{mn^c}\right)\|\vartheta_1 - \vartheta_1^*\| \\ &+ \frac{2\gamma_0}{mn} \|X^\top X - \mathsf{E}[X^\top X]\|_{op} \|\vartheta_1 - \vartheta_1^*\| + \frac{2\gamma_0}{mn} \|X^\top \epsilon\|. \end{aligned} \tag{2}$$

Thus in order to bound $\|T_1\|$, we need to analyze two terms, $\|X^\top X - \mathsf{E}[X^\top X]\|_{op}$ and $\|X^\top \epsilon\|$. To bound $\|X^\top X - \mathsf{E}[X^\top X]\|_{op}$, we first provide an analysis of N showing that it is large enough. Using Lemma 1 in conjunction with Assumption 2, we see that the probability of correctly classifying any worker machine i, given by $\mathsf{P}(E_i)$, satisfies $\mathsf{P}(E_i) \ge \frac{1}{2}$. Hence, we obtain

$$\mathsf{E}[|S_1 \cap S_1^*|] \ge \mathsf{E}\left[\frac{1}{2}|S_1^*|\right] = \frac{1}{2} p_1 m,$$

where we use the fact that $|S_1^*| = p_1 m$. Since $|S_1 \cap S_1^*|$ is a sum of Bernoulli random variables with success probability at least $\frac{1}{2}$, we obtain

$$\begin{aligned} &\mathsf{P}\left(|S_1 \cap S_1^*| \le \frac{1}{4} p_1 m\right) \\ &\le \mathsf{P}\left(|S_1 \cap S_1^*| - \mathsf{E}[|S_1 \cap S_1^*|] \ge \frac{1}{4} p_1 m\right) \\ &\le 2\exp(-cpm), \end{aligned}$$

where $p = \min\{p_1, p_2, \ldots, p_k\}$, and the second step follows from Hoeffding's inequality. Hence, we obtain $|S_1 \cap S_1^*| \ge \frac{1}{4} p_1 m$ with high probability, which yields

$$\mathsf{P}\left(N \ge \frac{1}{4} p_1 m n^0\right) \ge 1 - 2\exp(-cpm). \tag{3}$$

By combining this fact with our assumption that $pmn^0 \gtrsim d$, we know that $N \gtrsim d$. Then, according to the concentration of the covariance of Gaussian random vectors [47], we know that with probability at least $1 - 2\exp(-\frac{1}{2} d)$,

$$\|X^\top X - \mathsf{E}[X^\top X]\|_{op} \le 6\sqrt{\frac{d}{N}} \cdot N. \tag{4}$$

We now proceed to bound $\|X^\top\|$. In particular, we use the following lemma.

Lemma 2: Consider a random matrix $X \in \mathbb{R}^{N\times d}$ with i.i.d. entries sampled according to $N(0,1)$, and $\in \mathbb{R}^N$ be a random vector sampled according to $N(0,\sigma^2 I)$, independently of X. Then we have with probability at least $1-2\exp(-c\max\{d,N\})$,

$$\|X\|_{op} \le c\max\{\sqrt{d},\sqrt{N}\},$$

and with probability at least $1-c_2\exp(-c_3\min\{d,N\})$,

$$\|X^\top\| \le c_4\sigma\sqrt{dN}.$$

We prove Lemma 2 in Appendix IX-B. Now we can combine (2), (4), (3), and Lemma 2 and obtain with probability at least $1-c_1\exp(-c_2\rho m)-c_3\exp(-c_4 d)$,

$$\|T_1\| \le (1-c_5\gamma\rho)\|\vartheta_1-\vartheta_1^\natural\| + c_6\gamma\sigma\sqrt{\frac{d_0}{mn}}. \tag{5}$$

Since we assume that $\rho \gtrsim \frac{\log m}{m}$ and $d \gtrsim \log m$, the success probability can be simplified as $1-1/\mathrm{poly}(m)$.

c) Bound $\|T_2\|$ We first condition on S_1. We have the following:

$$\nabla F_i(\vartheta_1) = \frac{2}{n}X_i^\top(Y_i - X_i\vartheta_1).$$

For $i \in S_1 \cap S_j^\natural$, with $j=1$, we have $Y_i = X_i\vartheta_j^\natural + \epsilon_i$, and so we obtain

$$n^0\nabla F_i(\vartheta_1) = 2X_i^\top X_i(\vartheta_j^\natural - \vartheta_1) + 2X_i^\top\epsilon_i,$$

which yields

$$n^0\|\nabla F_i(\vartheta_1)\| \le \|X_i\|_{op}^2\|X_i\| + \|X_i^\top\epsilon_i\|, \tag{6}$$

where we use the fact that $\|\vartheta_j^\natural - \vartheta_1\| \le \|\vartheta_j^\natural\| + \|\vartheta_1^\natural - \vartheta_1\| \le 1$. Then, we combine (6) and Lemma 2 and get with probability at least $1-c_1\exp(-c_2\min\{d,n^0\})$,

$$\|\nabla F_i(\vartheta_1)\| \le \frac{1}{n^0}(c_3\max\{d,n^0\} + c_4\sigma\sqrt{dn^0}) \le c_5\max\{1,\frac{d}{n^0}\}, \tag{7}$$

where we use our assumption that $\sigma \lesssim 1$. By union bound, we know that with probability at least $1-c_1 m\exp(-c_2\min\{d,n^0\})$, (7) holds for all $j \in \overline{S_1^\natural}$. In addition, since we assume that $n^0 \gtrsim \log m$, $d \gtrsim \log m$, this probability can be lower bounded by $1-1/\mathrm{poly}(m)$. This implies that conditioned on S_1, with probability at least $1-1/\mathrm{poly}(m)$,

$$\|T_2\| \le c_5\frac{\gamma}{m}|S_1\cap\overline{S_1^\natural}|\max\{1,\frac{d}{n^0}\}. \tag{8}$$

Since we choose $\gamma = \frac{c}{\rho}$, we have $\frac{\gamma}{m}\max\{1,\frac{d}{n^0}\} \lesssim 1$, where we use our assumption that $\rho mn^0 \gtrsim d$. This shows that with probability at least $1-1/\mathrm{poly}(m)$,

$$\|T_2\| \le c_5|S_1\cap\overline{S_1^\natural}|. \tag{9}$$

We then analyze $|S_1\cap\overline{S_1^\natural}|$. By Lemma 1, we have

$$\mathbb{E}[|S_1\cap\overline{S_1^\natural}|] \le c_6 m\exp(-c_7(\frac{\alpha\rho}{\rho+1})^2 n^0). \tag{10}$$

According to Assumption 2, we know that $n^0 \ge \frac{c}{\alpha^2}(\frac{\rho+1}{\rho})^2\log m$, for some constant c that is large enough. Therefore, $m \le \exp(\frac{1}{c}(\frac{\alpha\rho}{\rho+1})^2 n^0)$, and thus, as long as c is large enough such that $\frac{1}{c} < c_7$ where c_7 is defined in (10), we have

$$\mathbb{E}[|S_1\cap\overline{S_1^\natural}|] \le c_6\exp(-c_8(\frac{\alpha\rho}{\rho+1})^2 n^0). \tag{11}$$

and then by Markov's inequality, we have

$$\begin{aligned}\mathbb{P}\big(|S_1\cap\overline{S_1^\natural}| &\le c_6\exp(-\frac{c_8}{2}(\frac{\alpha\rho}{\rho+1})^2 n^0)\big)\\ &\ge 1-\exp(-\frac{c_8}{2}(\frac{\alpha\rho}{\rho+1})^2 n^0)) \ge 1-\mathrm{poly}(m).\end{aligned} \tag{12}$$

Combining (9) with (12), we know that with probability at least $1-1/\mathrm{poly}(m)$,

$$\|T_2\| \le c_1\exp(-c_2(\frac{\alpha\rho}{\rho+1})^2 n^0).$$

Using this fact and (5), we obtain that with probability at least $1-1/\mathrm{poly}(m)$,

$$\begin{aligned}\|\vartheta_1^+ - \vartheta_1^\natural\| &\le (1-c_1\gamma\rho)\|\vartheta_1-\vartheta_1^\natural\|\\ &\quad + c_2\gamma\sigma\sqrt{\frac{d_0}{mn}} + c_3\exp(-c_4(\frac{\alpha\rho}{\rho+1})^2 n^0).\end{aligned}$$

Then we can complete the proof for the first cluster by choosing $\gamma = \frac{1}{2c_1\rho}$. To complete the proof for all the k clusters, we can use union bound, and the success probability is $1-k/\mathrm{poly}(m)$. However, since $k \le m$ by definition, we still have success probability $1-1/\mathrm{poly}(m)$.

A. Proof of Lemma 1

Without loss of generality, we analyze $E_i^{1,j}$ for some $j=1$. By definition, we have

$$E_i^{1,j} = \{F_i(\vartheta_j;\mathcal{Z}_i) \le F_i(\vartheta_1;\mathcal{Z}_i)\},$$

where $\mathcal{Z}_i$ is the set of n^0 data points that we use to estimate the cluster identity in this iteration. We write the data points in $\mathcal{Z}_i$ in matrix form with feature matrix $X_i \in \mathbb{R}^{n^0\times d}$ and response vector $Y_i = X_i\vartheta_1^\natural + \epsilon_i$. According to our model, all the entries of X_i are i.i.d. sampled according to $N(0,1)$, and $\epsilon_i \sim N(0,\sigma^2 I)$. Then, we have

$$\mathbb{P}\{E_i^{1,j}\} = \mathbb{P}\big\{\|X_i(\vartheta_1^\natural - \vartheta_1) + \epsilon_i\|^2 \ge \|X_i(\vartheta_j^\natural - \vartheta_j) + \epsilon_i\|^2\big\}.$$

Consider the random vector $X_i(\vartheta_1^\natural - \vartheta_j) + \epsilon_i$, and in particular consider the ℓ-th coordinate of it. Since X_i and ϵ_i are independent and we resample (X_i, Y_i) at each iteration, the ℓ-th coordinate of $X_i(\vartheta^\natural - \vartheta_j) + \epsilon_i$ is a Gaussian random variable with mean 0 and variance $\|\vartheta_j - \vartheta^\natural\|^2 + \sigma^2$. Since X_i and ϵ_i contain independent rows, the distribution of $\|X_i(\vartheta_1^\natural - \vartheta_j) + \epsilon_i\|^2$ is given by $(\|\vartheta_j - \vartheta^\natural\|^2 + \sigma^2)u_j$, where u_j is a standard Chi-squared random variable n^0 degrees of freedom. We now calculate the an upper bound on the following probability:

$$\begin{aligned}&\mathbb{P}\big\{\|X_i(\vartheta_1^\natural - \vartheta_1) + \epsilon_i\|^2 \ge \|X_i(\vartheta^\natural - \vartheta_j) + \epsilon_i\|^2\big\}\\ &\le \mathbb{P}\big\{\|X_i(\vartheta^\natural - \vartheta_j) + \epsilon_i\|^2 \le t\big\}\\ &\quad + \mathbb{P}\big\{\|X_i(\vartheta_1^\natural - \vartheta_1) + \epsilon_i\|^2 > t\big\}\\ &\le \mathbb{P}\big\{(\|\vartheta_j - \vartheta_1^\natural\|^2 + \sigma^2)u_j \le t\big\}\\ &\quad + \mathbb{P}\big\{(\|\vartheta_j - \vartheta_1^\natural\|^2 + \sigma^2)u_j > t\big\},\end{aligned} \tag{13}$$

where (i) holds for all $t \geq 0$. For the first term, we use the concentration property of Chi-squared random variables. Using the fact that $\|\vartheta_j - \vartheta_1^*\| \geq \|\vartheta_i^* - \vartheta_1^*\| - \|\vartheta_j - \vartheta_j^*\| \geq \Delta - (\frac{1}{2} - \alpha)\Delta = (\frac{1}{2} + \alpha)\Delta$, we have

$$\begin{aligned} &\mathsf{P}\left(\|\vartheta_j - \vartheta_1^*\|^2 + \sigma^2\right)u_j \leq t \\ &\leq \mathsf{P}\left(\left(\left(\tfrac{1}{2} + \alpha\right)^2\Delta^2 + \sigma^2\right)u_j \leq t\right). \end{aligned} \tag{14}$$

Similarly, using the initialization condition, $\|\vartheta_1 - \vartheta_1^*\| \leq \frac{1}{4}\Delta$, the second term of equation (13) can be simplified as

$$\begin{aligned} &\mathsf{P}\left(\|\vartheta_1 - \vartheta_1^*\|^2 + \sigma^2\right)u_j > t \\ &\leq \mathsf{P}\left(\left(\left(\tfrac{1}{2} - \alpha\right)^2\Delta^2 + \sigma^2\right)u_j > t\right). \end{aligned} \tag{15}$$

Based on the above observation, we now choose $t = n^0((\frac{1}{4} + \alpha^2)\Delta^2 + \sigma^2)$. Recall that $\rho := \frac{\Delta^2}{\sigma^2}$. Then the inequality (14) can be rewritten as

$$\begin{aligned} &\mathsf{P}\left(\|\vartheta_j - \vartheta_1^*\|^2 + \sigma^2\right)u_j \leq t \\ &\leq \mathsf{P}\left(\frac{u_j}{n^0} - 1 \leq -\frac{4\alpha\rho}{(1+2\alpha)^2\rho + 4}\right). \end{aligned}$$

According to the concentration results for standard Chi-squared distribution [47], we know that there exists universal constants c_1 and c_2 such that

$$\begin{aligned} &\mathsf{P}\left(\|\vartheta_j - \vartheta_1^*\|^2 + \sigma^2\right)u_j \leq t \\ &\leq c_1 \exp\left(-c_2 n^0 \frac{(1+2\alpha)^2\rho + 1}{(\rho+1)^2}\right) \\ &\leq c_1 \exp\left(-c_2 n^0 \frac{\alpha^2\rho^2}{(\rho+1)^2}\right) \end{aligned} \tag{16}$$

where we use the fact that $\alpha < \frac{1}{2}$. Similarly, the inequality (15) can be rewritten as

$$\begin{aligned} &\mathsf{P}\left(\|\vartheta_1 - \vartheta_1^*\|^2 + \sigma^2\right)u_j > t \\ &\leq \mathsf{P}\left(\frac{u_j}{n^0} - 1 > \frac{4\alpha\rho}{(1-2\alpha)^2\rho + 4}\right) \end{aligned}$$

and again, according to the concentration of Chi-squared distribution, there exists universal constants c_3 and c_4 such that

$$\begin{aligned} &\mathsf{P}\left(\|\vartheta_1 - \vartheta_1^*\|^2 + \sigma^2\right)u_1 > t \\ &\leq c_3 \exp\left(-c_4 n^0 \frac{(1+2\alpha)^2\rho + 1}{(\rho+1)^2}\right) \\ &\leq c_3 \exp\left(-c_4 n^0 \frac{\alpha^2\rho^2}{(\rho+1)^2}\right), \end{aligned} \tag{17}$$

where we use the fact that $\alpha > 0$. The proof can be completed by combining (13), (16) and (17).

B. Proof of Lemma 2

According to Theorem 5.39 of [55], we have with probability at least $1 - 2\exp(-c_1 \max\{d, N\})$,

$$\|X\|_{op} \leq c \max\{\sqrt{d}, \sqrt{N}\},$$

where c and c_1 are universal constants. As for $\|X^\top\|$, we first condition on X. According to the Hanson-Wright inequality [56], we obtain for every $t \geq 0$

$$\mathsf{P}\left(\|X^\top\| - \sigma\|X^\top\|_F > t\right) \leq 2\exp\left(-c\frac{t^2}{\sigma^2\|X^\top\|_{op}^2}\right). \tag{18}$$

Using Chi-squared concentration [47], we obtain with probability at least $1 - 2\exp(-cdN)$,

$$\|X\|_F \leq c\sqrt{dN}.$$

Furthermore, using the fact that $\|X^\top\|_{op} = \|X\|_{op}$ and substituting $t = \sigma\sqrt{dN}$ in (18), we obtain with probability at least $1 - c_2\exp(-c_3\min\{d, N\})$,

$$\|X^\top\| \leq c_4\sigma\sqrt{dN}.$$

X. Proof of Corollary 1

Let $\alpha_t \in \mathbb{R}$ be the real number such that

$$\|\vartheta_j^{(t)} - \vartheta_j^*\| \leq \left(\frac{1}{2} - \alpha_t\right)\Delta, \ \forall j \in [k].$$

Recall that according to Theorem 1, we have

$$\begin{aligned} \|\vartheta_j^{(t+1)} - \vartheta_j^*\| \leq \ &\frac{1}{2}\|\vartheta_j^{(t)} - \vartheta_j^*\| + c_2\frac{\sigma}{p}\sqrt{\frac{d_0}{mn}} \\ &+ c_3\exp\left(-c_4\left(\frac{\alpha_t\rho}{\rho+1}\right)^2 n^0\right). \end{aligned}$$

Note that with Assumption 2 and the fact that $0 < \alpha_0 < \frac{1}{2}$, we can ensure that for all t, $\|\vartheta_j^{(t+1)} - \vartheta_j^*\| \leq \|\vartheta_j^{(t)} - \vartheta_j^*\|$. Hence the sequence $\{\alpha_0, \alpha_1, \ldots\}$ is non-decreasing. Let

$$\varepsilon_0 := c_2\frac{\sigma}{p}\sqrt{\frac{d_0}{mn}} + c_3\exp\left(-c_4\left(\frac{\alpha_0\rho}{\rho+1}\right)^2 n^0\right)$$

be the initial *error floor*. Now, we show that if IFCA is run for $T^0 = \log(C_1(1-2\alpha_0))$ iterations, we obtain a sufficiently large value of α_{T^0}, i.e., $\alpha_{T^0} \geq \frac{1}{4}$. After T^0 iterations, we have

$$\|\vartheta_j^{(T^0)} - \vartheta_j^*\| \leq \left(\frac{1}{2}\right)^{T^0}\left(\frac{1}{2} - \alpha_0\right)\Delta + 2\varepsilon_0,$$

and we hope to ensure that the right hand side of the above equation is upper bounded by $\frac{1}{4}\Delta$, yielding $\alpha_{T^0} \geq \frac{1}{4}$. Note that the conditions (a) $(\frac{1}{2})^{T^0}(\frac{1}{2} - \alpha_0)\Delta \leq \frac{1}{8}\Delta$ and (b) $2\varepsilon_0 \leq \frac{1}{8}\Delta$ suffice. Part (b) follows directly from the separation condition of Assumption 2. For part (a), observe that $\frac{1}{2} - \alpha_0 < \frac{1}{2}$, and thus it suffices to ensure that $(\frac{1}{2})^{T^0} \leq \frac{1}{4}$. Thus, after a constant number of iterations ($T^0 \geq 2$), we have $\alpha_t \geq \frac{1}{4}$. Afterwards, the iterates of IFCA satisfies $\|\vartheta_j^{(t)} - \vartheta_j^*\| \leq \frac{1}{4}\Delta$, for $t \geq T^0$. After additional T iterations, we obtain

$$\|\vartheta_j^{(T)} - \vartheta_j^*\| \leq \left(\frac{1}{2}\right)^{T^0}\frac{\Delta}{4} + 2\varepsilon,$$

with high probability. Finally observe that plugging $T = \log(\Delta/4\varepsilon)$, the final accuracy is $O(\varepsilon)$, and this completes the proof.

XI. Proof of Theorem 2

Suppose that at a certain step, we have model parameters ϑ_j $j \in [k]$ for the k clusters. Assume that $k\vartheta_j - \vartheta_j^* k \le (\frac{1}{2} - \alpha)\frac{\lambda}{L}\Delta$, for all $j \in [k]$.

a) Probability of Erroneous Cluster Identity Estimation: We first calculate the probability of erroneous estimation of worker machines' cluster identity. We define the events E_i^{j,j^0} in the same way as in Appendix IX, and have the following lemma.

Lemma 3: Suppose that worker machine $i \in S_j^*$. Then there exists a universal constants c_1 such that for any $j^0 = j$,

$$\mathsf{P}(E_i^{j,j^0}) \le c_1 \frac{\eta^2}{\alpha^2\lambda^2\Delta^4 n},$$

and by union bound

$$\mathsf{P}(\overline{E_i}) \le c_1 \frac{k\eta^2}{\alpha^2\lambda^2\Delta^4 n}.$$

We prove Lemma 3 in Appendix XI-A. Now we proceed to analyze the gradient descent iteration. Without loss of generality, we focus on ϑ_1. We have

$$k\vartheta_1^+ - \vartheta_1^* k = k\vartheta_1 - \vartheta_1^* - \frac{\gamma}{m}\sum_{i\in S_1} \nabla F_i(\vartheta_1) k,$$

where $F_i(\vartheta) := F_i(\vartheta; Z_i)$ with Z_i being the set of data points on the i-th worker machine that we use to compute the gradient, and S_1 is the set of indices returned by Algorithm 2 corresponding to the first cluster. Since

$$S_1 = (S_1 \cap S_1^*) \cup (S_1 \cap \overline{S_1^*})$$

and the sets are disjoint, we have

$$k\vartheta_1^+ - \vartheta_1^* k = k\vartheta_1 - \vartheta_1^* - \underbrace{\frac{\gamma}{m}\sum_{i\in S_1\cap S_1^*} \nabla F_i(\vartheta_1)}_{T_1} - \underbrace{\frac{\gamma}{m}\sum_{i\in S_1\cap \overline{S_1^*}} \nabla F_i(\vartheta_1)}_{T_2} k.$$

Using triangle inequality, we obtain

$$k\vartheta_1^+ - \vartheta_1^* k \le kT_1 k + kT_2 k,$$

and we control both the terms separately. Let us first focus on $kT_1 k$.

b) Bound $kT_1 k$: We first split T_1 in the following way:

$$T_1 = \underbrace{\vartheta_1 - \vartheta_1^* - \eta\nabla F^1(\vartheta_1)}_{T_{11}} + \eta\Big(\nabla F^1(\vartheta_1) - \frac{1}{|S_1\cap S_1^*|}\sum_{i\in S_1\cap S_1^*} \nabla F_i(\vartheta_1)\Big), \tag{19}$$

where $\eta := \gamma\frac{|S_1\cap S_1^*|}{m}$. Let us condition on S_1. According to standard analysis technique for gradient descent on strongly convex functions, we know that when $\eta \le \frac{1}{L}$,

$$kT_{11} k = k\vartheta_1 - \vartheta_1^* - \eta\nabla F^1(\vartheta_1) k \le \Big(1 - \frac{\eta L\lambda}{\lambda + L}\Big) k\vartheta_1 - \vartheta_1^* k. \tag{20}$$

Further, we have $\mathsf{E}[kT_{12}k^2] = \frac{\nu^2}{n^0|S_1\cap S_1^*|}$, which implies $\mathsf{E}[kT_{12}k] \le \frac{\nu}{\sqrt{n^0|S_1\cap S_1^*|}}$, and thus by Markov's inequality, for any $\delta_0 > 0$, with probability at least $1 - \delta_0$,

$$kT_{12}k \le \frac{\nu}{\delta_0\sqrt{n^0|S_1\cap S_1^*|}}. \tag{21}$$

We then analyze $|S_1 \cap S_1^*|$. Similar to the proof of Theorem 1, we can show that $|S_1\cap S_1^*|$ is large enough. From Lemma 3 and using our assumption, we see that the probability of correctly classifying any worker machine i, given by $\mathsf{P}(E_i)$, satisfies $\mathsf{P}(E_i) \ge \frac{1}{2}$. Recall $p = \min\{p_1, p_2, \ldots, p_k\}$, and we obtain $|S_1 \cap S_1^*| \ge \frac{1}{4}p_1 m$ with probability at least $1 - 2\exp(-cpm)$. Let us condition on $|S_1 \cap S_1^*| \ge \frac{1}{4}p_1 m$ and choose $\gamma = 1/L$. Then $\eta \le 1/L$ is satisfied, and on the other hand $\eta \ge \frac{p}{4L}$. Plug this fact in (20), we obtain

$$kT_{11} k \le \Big(1 - \frac{p\lambda}{8L}\Big) k\vartheta_1 - \vartheta_1^* k. \tag{22}$$

We then combine (21) and (22) and have with probability at least $1 - \delta_0 - 2\exp(-cpm)$,

$$kT_1 k \le \Big(1 - \frac{p\lambda}{8L}\Big) k\vartheta_1 - \vartheta_1^* k + \frac{2\nu}{\delta_0 L\sqrt{pmn}}. \tag{23}$$

c) Bound $kT_2 k$: Let us define $T_{2j} := \sum_{i\in S_1\cap S_j^*} \nabla F_i(\vartheta_1)$, $j \ge 2$. We have $T_2 = \frac{\gamma}{m}\sum_{j=2}^{k} T_{2j}$. We condition on S_1 and first analyze T_{2j}. We have

$$T_{2j} = |S_1 \cap S_j^*|\nabla F^j(\vartheta_1) + \sum_{i\in S_1\cap S_j^*} \nabla F_i(\vartheta_1) - \nabla F^j(\vartheta_1). \tag{24}$$

Due to the smoothness of $F^j(\vartheta)$, we know that

$$k\nabla F^j(\vartheta_1) k \le Lk\vartheta_1 - \vartheta_j^* k \le 3L, \tag{25}$$

where we use the fact that $k\vartheta_1 - \vartheta_j^* k \le k\vartheta_j^* k + k\vartheta_1^* k + k\vartheta_1 - \vartheta_1^* k \le 1 + 1 + (\frac{1}{2} - \alpha)\frac{\lambda}{L}\Delta \le 1$. Here, we use the fact that $\Delta \le 2$. In addition, we have

$$\mathsf{E}\Big[\sum_{i\in S_1\cap S_j^*} \nabla F_i(\vartheta_1) - \nabla F^j(\vartheta_1)\Big] = |S_1 \cap S_j^*|\frac{\nu^2}{n},$$

which implies

$$\mathsf{E}\Big[\sum_{i\in S_1\cap S_j^*} \nabla F_i(\vartheta_1) - \nabla F^j(\vartheta_1)\Big] \le \sqrt{|S_1\cap S_j^*|}\frac{\nu}{\sqrt{n}},$$

and then according to Markov's inequality, for any $\delta_1 \in (0, 1)$, with probability at least $1 - \delta_1$,

$$\sum_{i\in S_1\cap S_j^*} \nabla F_i(\vartheta_1) - \nabla F^j(\vartheta_1) \le \sqrt{|S_1\cap S_j^*|}\frac{\nu}{\delta_1\sqrt{n^0}}. \tag{26}$$

Then, by combining (25) and (26), we know that with probability at least $1 - \delta_1$,

$$kT_{2j} k \le c_1 L|S_1 \cap S_j^*| + \sqrt{|S_1\cap S_j^*|}\frac{\nu}{\delta_1\sqrt{n^0}}. \tag{27}$$

By union bound, we know that with probability at least $1-k\delta_1$, (27) applies to all $j = 1$. Then, we have with probability at least $1 - k\delta_1$,

$$\|T_2\| \le \frac{c_1\gamma L}{m}|S_1 \cap \overline{S_1^{\mathbb{I}}}| + \frac{\gamma\nu\sqrt{k}}{\delta_1 m\sqrt{n^0}}\sqrt{|S_1 \cap \overline{S_1^{\mathbb{I}}}|}. \tag{28}$$

According to Lemma 3, we know that

$$\mathbb{E}[|S_1 \cap \overline{S_1^{\mathbb{I}}}|] \le c_1\frac{\eta^2 m}{\alpha^2\lambda^2\Delta^4 n^0}.$$

Then by Markov's inequality, we know that with probability at least $1 - \delta_2$,

$$|S_1 \cap \overline{S_1^{\mathbb{I}}}| \le c_1\frac{\eta^2 m}{\delta_2\alpha^2\lambda^2\Delta^4 n^0}. \tag{29}$$

Now we combine (28) with (29) and obtain with probability at least $1 - k\delta_1 - \delta_2$,

$$\|T_2\| \le c_1\frac{\eta^2}{\delta_2\alpha^2\lambda^2\Delta^4 n^0} + c_2\frac{\nu\eta\sqrt{k}}{\delta_1\sqrt{\delta_2}\alpha\lambda L\Delta^2\sqrt{mn^0}}. \tag{30}$$

Combining (23) and (30), we know that with probability at least $1 - \delta_0 - k\delta_1 - \delta_2 - 2\exp(-cpm)$,

$$\begin{aligned}\|\vartheta_1^+ - \vartheta_1^{\mathbb{I}}\| \le{}& (1 - \frac{p\lambda}{8L})\|\vartheta_1 - \vartheta_1^{\mathbb{I}}\| + \frac{2\nu}{\delta_0 L\sqrt{pmn}} \\ &+ c_1\frac{\eta^2}{\delta_2\alpha^2\lambda^2\Delta^4 n^0} + c_2\frac{\nu\eta\sqrt{k}}{\delta_1\sqrt{\delta_2}\alpha\lambda L\Delta^2\sqrt{mn^0}}.\end{aligned} \tag{31}$$

Let $\delta \ge \delta_0 + k\delta_1 + \delta_2 + 2\exp(-cpm)$ be the failure probability of this iteration, and choose $\delta_0 = \frac{\delta}{4}$, $\delta_1 = \frac{\delta}{4k}$, and $\delta_2 = \frac{\delta}{4}$. Then the failure probability is upper bounded by δ as long as $2\exp(-cpm) \le \frac{\delta}{4}$, which is guaranteed by our assumption that $p \gtrsim \frac{1}{m}\log(mn^0)$. Therefore, we conclude that with probability at least $1 - \delta$, we have

$$\begin{aligned}\|\vartheta_1^+ - \vartheta_1^{\mathbb{I}}\| \le{}& (1 - \frac{p\lambda}{8L})\|\vartheta_1 - \vartheta_1^{\mathbb{I}}\| + \frac{c_0\nu}{\delta L\sqrt{pmn}} \\ &+ \frac{c_1\eta^2}{\delta\alpha^2\lambda^2\Delta^4 n^0} + \frac{c_2\nu\eta k^{3/2}}{\delta^{3/2}\alpha\lambda L\Delta^2\sqrt{mn_0}},\end{aligned}$$

which completes the proof.

A. Proof of Lemma 3

Without loss of generality, we bound the probability of $E_i^{1,j}$ for some $j = 1$. We know that

$$E_i^{1,j} = \left\{F_i(\vartheta_1; \mathcal{Z}_i) \ge F_i(\vartheta_j; \mathcal{Z}_i)\right\},$$

where $\mathcal{Z}_i$ is the set of n^0 data points that we use to estimate the cluster identity in this iteration. In the following, we use the shorthand notation $F_i(\vartheta) := F_i(\vartheta; \mathcal{Z}_i)$. We have

$$\mathbb{P}(E_i^{1,j}) \le \mathbb{P}(F_i(\vartheta_1) > t) + \mathbb{P}(F_i(\vartheta_j) \le t)$$

for all $t \ge 0$. We choose $t = \frac{F^1(\vartheta_1)+F^1(\vartheta_j)}{2}$. With this choice, we obtain

$$\begin{aligned}\mathbb{P}(F_i(\vartheta_1) > t) &= \mathbb{P}\left(F_i(\vartheta_1) > \frac{F^1(\vartheta_1) + F^1(\vartheta_j)}{2}\right) &(32)\\ &= \mathbb{P}\left(F_i(\vartheta_1) - F^1(\vartheta_1) > \frac{F^1(\vartheta_j) - F^1(\vartheta_1)}{2}\right). &(33)\end{aligned}$$

Similarly, for the second term, we have

$$\mathbb{P}(F_i(\vartheta_j) \le t) = \mathbb{P}\left(F_i(\vartheta_j) - F^1(\vartheta_j) \le -\frac{F^1(\vartheta_j) - F^1(\vartheta_1)}{2}\right). \tag{34}$$

Based on our assumption, we know that $\|\vartheta_j - \vartheta_1\| \ge \Delta - (\frac{1}{2} - \alpha)\sqrt{\frac{\lambda}{L}}\Delta \ge (\frac{1}{2} + \alpha)\Delta$. According to the strong convexity of $F^1(\cdot)$,

$$F^1(\vartheta_j) \ge F^1(\vartheta_1^{\mathbb{I}}) + \frac{\lambda}{2}\|\vartheta_j - \vartheta_1^{\mathbb{I}}\|^2 \ge F^1(\vartheta_1^{\mathbb{I}}) + \frac{1}{2}(\frac{1}{2} + \alpha)^2\lambda\Delta^2,$$

and according to the smoothness of $F^1(\cdot)$,

$$\begin{aligned}F^1(\vartheta_1) &\le F^1(\vartheta_1^{\mathbb{I}}) + \frac{L}{2}\|\vartheta_1 - \vartheta_1^{\mathbb{I}}\|^2 \le F^1(\vartheta_1^{\mathbb{I}}) + \frac{L}{2}\frac{\lambda(\frac{1}{2} - \alpha)^2}{L}\Delta^2 \\ &= F^1(\vartheta_1^{\mathbb{I}}) + \frac{1}{2}(\frac{1}{2} - \alpha)^2\lambda\Delta^2.\end{aligned}$$

Therefore, $F^1(\vartheta_j) - F^1(\vartheta_1) \ge \alpha\lambda\Delta^2$. Then, according to Chebyshev's inequality, we obtain that $\mathbb{P}(F_i(\vartheta_1) > t) \le \frac{4\eta^2}{\alpha^2\lambda^2\Delta^4 n^0}$ and that $\mathbb{P}(F_i(\vartheta_j) \le t) \le \frac{4\eta^2}{\alpha^2\lambda^2\Delta^4 n^0}$, which completes the proof.

XII. Proof of Corollary 2

Recall that the error floor at the initial step according to Theorem 2 is

$$\varepsilon_0 \cdot \frac{\nu}{\delta L\sqrt{pmn_0}} + \frac{\eta^2 k}{\delta\alpha_0^2\lambda^2\Delta^4 n^0} + \frac{\nu\eta k^{3/2}}{\delta^{3/2}\alpha_0\lambda L\Delta^2\sqrt{mn_0}}$$

Note that the iterate of IFCA is contractive, i.e., $\|\vartheta_j^{(t+1)} - \vartheta_j^{\mathbb{I}}\| \le \|\vartheta_j^{(t)} - \vartheta_j^{\mathbb{I}}\|$, provided

$$\varepsilon_0 \cdot \frac{p\lambda}{L}(\frac{1}{2} - \alpha_0)\sqrt{\frac{\lambda}{L}}\Delta.$$

This is ensured via Assumption 6, particularly the condition

$$\Delta \gtrsim \max\{\alpha_0^{-2/5}(n^0)^{-1/5}, \alpha_0^{-1/3}m^{-1/6}(n^0)^{-1/3}\}$$

in conjunction with the fact that $0 < \alpha_0 < \frac{1}{2}$. Let $\alpha_t \in \mathbb{R}$ be the real number such that

$$\|\vartheta_j^{(t)} - \vartheta_j^{\mathbb{I}}\| \le (\frac{1}{2} - \alpha_t)\sqrt{\frac{\lambda}{L}}\Delta, \quad \forall j \in [k].$$

From the argument above, the sequence $\{\alpha_0, \alpha_1, \ldots\}$ is non-decreasing.

Now, we show that if IFCA is run for T^0 iterations, we obtain a sufficiently large value of α_{T^0}, i.e., $\alpha_{T^0} \ge \frac{1}{4}$. After T^0 iterations, we have

$$\|\vartheta_j^{(T^0)} - \vartheta_j^{\mathbb{I}}\| \le (1 - \frac{p\lambda}{8L})^{T^0}(\frac{1}{2} - \alpha_0)\Delta + \frac{8L}{p\lambda}\varepsilon_0,$$

and if the right hand side of the above equation is upper bounded by $\frac{1}{4}\sqrt{\frac{\lambda}{L}}\Delta$, we can have $\alpha_{T^0} \ge \frac{1}{4}$. Note that the conditions (a) $(1 - \frac{p\lambda}{8L})^{T^0}(\frac{1}{2} - \alpha_0)\Delta \le \frac{1}{8}\sqrt{\frac{\lambda}{L}}\Delta$ and (b) $\frac{8L}{p\lambda}\varepsilon_0 \le \frac{1}{8}\sqrt{\frac{\lambda}{L}}\Delta$ suffice. Part (b) follows directly from the separation condition of Assumption 6. For part (a), observe that it suffices

to ensure that $(1-\frac{p\lambda}{8L})^{T^0} \le \frac{1}{4}\sqrt{\frac{\lambda}{L}}$. Thus, we know it suffices to have

$$T^0 \ge \frac{\log(4\sqrt{L/\lambda})}{\log(\frac{1}{1-\frac{p\lambda}{8L}})}.$$

Using the fact that $\log(1-x) \le -x$ for any $x \in (0,1)$, we further know that after

$$T^0 \ge \frac{8L}{p\lambda}\log(4\sqrt{L/\lambda}) \tag{35}$$

iterations, we have $\|\vartheta_j^{(T^0)} - \vartheta_j^*\| \le \frac{1}{4}\sqrt{\frac{\lambda}{L}}\Delta$.

Now that after T^0 iterations, we have $\|\vartheta^{(t)}-\vartheta_j^*\| \le \frac{1}{4}\sqrt{\frac{\lambda}{L}}\Delta$, we keep running the algorithm for another T^α iterations. In these iterations, since $\alpha_t \ge 1/4$, the error floor according to Theorem 2 becomes

$$\varepsilon_0 \cdot \frac{\nu}{\delta_0 L\sqrt{pmn^0}} + \frac{\eta^2 k\, n_0}{\delta_0\lambda^2\Delta^4} + \frac{\nu\eta k^{3/2}}{\delta_0^{3/2}\lambda L\Delta^2\sqrt{mn^0}}, \tag{36}$$

since α can be merged into the constant. Note that we also use the new symbol δ_0 as the failure probability of a fixed cluster $j \in [k]$ in a fixed iteration, instead of δ. We will use the definitions of ε_0 and δ_0 in (36) in the following. Let $T := T^0 + T^{00}$. We then have with probability at least $1-kT\delta_0$, for all $j \in [k]$,

$$\|\vartheta_j^{(T)} - \vartheta_j^*\| \le (1-\frac{p\lambda}{8L})^{T^{00}}\|\vartheta_j^{(T^0)} - \vartheta_j^*\| + \frac{8L}{p\lambda}\varepsilon_0.$$

Then, we know that when we choose

$$T^{00} = \frac{8L}{p\lambda}\log\left(\frac{p\Delta}{32\varepsilon_0 L}\sqrt{\frac{\lambda}{L}}^{3/2}\right), \tag{37}$$

we have

$$(1-\frac{p\lambda}{8L})^{T^{00}}\|\vartheta_j^{(T^0)} - \vartheta_j^*\| \le \exp(-\frac{p\lambda}{8L}T^{00})\frac{1}{4}\sqrt{\frac{\lambda}{L}}\Delta \le \frac{8L}{p\lambda}\varepsilon_0,$$

which implies $\|\vartheta_j^{(T)} - \vartheta_j^*\| \le \frac{16L}{p\lambda}\varepsilon_0$. The total number of iterations we need is then at most

$$T = T^0 + T^{00} \le \frac{8L}{p\lambda}\log\left(\frac{p\Delta}{32\varepsilon_0}\frac{\lambda}{L}^{3/2}\cdot 4\sqrt{\frac{L}{\lambda}}\right)$$
$$\le \frac{8L}{p\lambda}\log\left(\frac{p\lambda\Delta}{8\varepsilon_0 L}\right),$$

according to (35) and (37). Finally, we check the failure probability. Let δ be the upper bound of the failure probability of the entire algorithm. Let us choose

$$\delta_0 = \frac{p\lambda\delta}{ckL\log(mn^0)} \tag{38}$$

with some constant $c > 0$. The failure probability is

$$kT\delta^0 \le \frac{8kL}{p\lambda}\log\frac{\Delta}{\varepsilon_0}\cdot\frac{p\lambda\delta}{ckL\log(mn^0)} = \frac{8\delta}{c}\frac{\log(\frac{\Delta}{\varepsilon_0})}{\log(mn^0)}$$
$$\le \delta\frac{\log(\frac{1}{\varepsilon_0})}{\log((mn^0)^{c/8})}.$$

On the other hand, according to (36), we know that

$$\frac{1}{\varepsilon_0} \le \mathcal{O}(\max\{\sqrt{mn^0}, n^0\}),$$

then, as long as c is large enough, we can guarantee that $(mn^0)^{c/8} > \frac{1}{\varepsilon_0}$, which implies that the failure probability is upper bounded by δ. Our final error floor can be obtained by replacing δ_0 in (36) with (38) and redefining

$$\varepsilon := \frac{16L}{p\lambda}\varepsilon_0.$$

ACKNOWLEDGMENT

The authors would like to thank Justin Hong for initial investigation on the one-shot clustering algorithm and also would like to thank Mehrdad Farajtabar for helpful comments.

REFERENCES

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[2] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:1610.02527*.

[3] B. McMahan and D. Ramage. (2017). *Federated Learning: Collaborative Machine Learning Without Centralized Training Data*. [Online]. Available: https://research.googleblog.com/2017/04/federated-learning-collaborative.html

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

[5] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learn-ing: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 8, pp. 3710–3722, Aug. 2020.

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

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

[8] B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering," in *Proc. 5th Int. Conf. Comput. Inf. Technol.*, vol. 1, Dec. 2002, pp. 291–324.

[9] Q. Li and B. Man Kim, "Clustering approach for hybrid recommender system," in *Proc. IEEE/WIC Int. Conf. Web Intell. (WI)*, Oct. 2003, pp. 33–38.

[10] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, no. 1, pp. 41–75, Jul. 1997.

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

[12] W. S. DeSarbo and W. L. Cron, "A maximum likelihood methodol-ogy for clusterwise linear regression," *J. Classification*, vol. 5, no. 2, pp. 249–282, 1988.

[13] Y. Sun, S. Ioannidis, and A. Montanari, "Learning mixtures of linear classifiers," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 721–729.

[14] M. Zinkevich, M. Weimer, L. Li, and A. J. Smola, "Parallelized stochastic gradient descent," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 2595–2603.

[15] B. Recht, C. Re, S. Wright, and F. Niu, "Hogwild: A lock-free approach to parallelizing stochastic gradient descent," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 693–701.

[16] M. Li *et al.*, "Scaling distributed machine learning with the parameter server," in *Proc. 11th USENIX Symp. Operating Syst. Design Implement. (OSDI)*, 2014, pp. 583–598.

[17] A. Hard *et al.*, "Federated learning for mobile keyboard prediction," 2018, *arXiv:1811.03604*.

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

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

[20] L. Li, W. Xu, T. Chen, G. B. Giannakis, and Q. Ling, "RSA: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, no. 1, 2019, pp. 1544–1551.

[21] 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. 2020.

[22] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," in *Proc. Int. Conf. Learn. Representations (ICLR)*, 2020, pp. 1–26.

[23] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 4615–4625.

[24] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning: A meta-learning approach," 2020, *arXiv:2002.07948*.

[25] F. Chen, M. Luo, Z. Dong, Z. Li, and X. He, "Federated meta-learning with fast convergence and efficient communication," 2018, *arXiv:1802.07876*.

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

[27] E. L. Zec, J. Martinsson, O. Mogren, L. R. Sütfeld, and D. Gillblad, "Federated learning using mixture of experts," in *Proc. ICLR Conf.*, 2020, pp. 1–10.

[28] M. Reisser, C. Louizos, E. Gavves, and M. Welling, "Federated mixture of experts," 2021, *arXiv:2107.06724*.

[29] L. Xu and M. I. Jordan, "On convergence properties of the EM algorithm for Gaussian mixtures," *Neural Comput.*, vol. 8, no. 1, pp. 129–151, 1996.

[30] D.-S. Lee, "Effective Gaussian mixture learning for video background subtraction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 5, pp. 827–832, May 2005.

[31] M. Wedel and W. A. Kamakura, "Mixture regression models," in *Market Segmentation* (International Series in Quantitative Marketing). Boston, MA, USA: Springer, 2000, pp. 101–124.

[32] D. Yin, R. Pedarsani, Y. Chen, and K. Ramchandran, "Learning mixtures of sparse linear regressions using sparse graph codes," *IEEE Trans. Inf. Theory*, vol. 65, no. 3, pp. 1430–1451, Mar. 2018.

[33] J. R. Fienup, "Phase retrieval algorithms: A comparison," *Appl. Opt.*, vol. 21, no. 15, pp. 2758–2769, Aug. 1982.

[34] R. P. Millane, "Phase retrieval in crystallography and optics," *J. Opt. Soc. Amer. A, Opt. Image Sci.*, vol. 7, no. 3, pp. 394–411, 1990.

[35] P. Netrapalli, P. Jain, and S. Sanghavi, "Phase retrieval using alternating minimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2796–2804.

[36] C. Daskalakis, C. Tzamos, and M. Zampetakis, "Ten steps of EM suffice for mixtures of two Gaussians," in *Proc. Conf. Learn. Theory*, 2017, pp. 704–710.

[37] X. Yi, C. Caramanis, and S. Sanghavi, "Solving a mixture of many random linear equations by tensor decomposition and alternating minimization," 2016, *arXiv:1608.05749*.

[38] S. Balakrishnan, M. J. Wainwright, and B. Yu, "Statistical guarantees for the EM algorithm: From population to sample-based analysis," *Ann. Statist.*, vol. 45, no. 1, pp. 77–120, 2017.

[39] I. Waldspurger, "Phase retrieval with random Gaussian sensing vectors by alternating projections," *IEEE Trans. Inf. Theory*, vol. 64, no. 5, pp. 3301–3312, May 2018.

[40] A. Ghosh and R. Kannan, "Alternating minimization converges super-linearly for mixed linear regression," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 1093–1103.

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

[42] S. Wang, F. Roosta-Khorasani, P. Xu, and M. W. Mahoney, "GIANT: Globally improved approximate Newton method for distributed optimization," in *Proc. 32nd Conf. Neural Inf. Process. Syst.*, 2018, pp. 1–11.

[43] A. Ghosh, R. K. Maity, and A. Mazumdar, "Distributed Newton can communicate less and resist Byzantine workers," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 18028–18038.

[44] P. Jain, P. Netrapalli, and S. Sanghavi, "Low-rank matrix completion using alternating minimization," in *Proc. 45th Annu. ACM Symp. Theory Comput.*, 2013, pp. 665–674.

[45] X. Yi, C. Caramanis, and S. Sanghavi, "Alternating minimization for mixed linear regression," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 613–621.

[46] B. Yan, M. Yin, and P. Sarkar, "Convergence of gradient EM on multi-component mixture of Gaussians," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6956–6966.

[47] M. J. Wainwright, *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*, vol. 48. Cambridge, U.K.: Cambridge Univ. Press, 2019.

[48] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, "Optimal distributed online prediction using mini-batches," *J. Mach. Learn. Res.*, vol. 13, pp. 165–202, Jan. 2012.

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

[50] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009.

[51] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6467–6476.

[52] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio, "An empirical investigation of catastrophic forgetting in gradient-based neural networks," 2013, *arXiv:1312.6211*.

[53] K. James *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proc. Nat. Acad. Sci. USA*, vol. 114, no. 13, pp. 3521–3526, Mar. 2017.

[54] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 19586–19597.

[55] R. Vershynin, "Introduction to the non-asymptotic analysis of random matrices," 2010, *arXiv:1011.3027*.

[56] M. Rudelson and R. Vershynin, "Hanson-Wright inequality and sub-Gaussian concentration," *Electron. Commun. Probab.*, vol. 18, pp. 1–9, Jan. 2013.

Avishek Ghosh received the bachelor's degree from the Department of Electronics and Telecommunication Engineering, Jadavpur University, the master's degree from the Electrical Communication Engineering Department, Indian Institute of Science (IISc), Bengaluru, and the Ph.D. degree from the Electrical Engineering and Computer Sciences (EECS) Department, UC Berkeley, in 2021, under the supervision of Prof. Kannan Ramchandran and Prof. Aditya Guntuboyina. He is currently a HDSI (Data Science) Post-Doctoral Fellow at the University of California at San Diego, San Diego. His research interests are broadly in theoretical machine learning, including federated learning, multi-agent reinforcement/bandit learning, theoretically understanding challenges in multi-agent systems, and competition/collaboration across agents.

Jichan Chung received the B.S. degree in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2017. He is currently pursuing the Ph.D. degree with the Department of Electrical Engineering and Computer Science, University of California at Berkeley, CA, USA. His research interests include distributed machine learning, federated learning, and memory-efficient learning.

Dong Yin received the B.S. degree from Tsinghua University, China, in 2014, and the Ph.D. degree from the Department of Electrical Engineering and Computer Sciences (EECS), UC Berkeley, in 2019, under the supervision of Prof. Kannan Ramchandran. He is currently a Research Scientist at Deep-Mind, Mountain View, CA, USA. His current research interests are theory and application of reinforcement learning, federated learning, robustness in machine learning, and representation learning.

Kannan Ramchandran (Fellow, IEEE) received the Ph.D. degree from Columbia University in 1993. He has been a Professor in electrical engineering and computer science at UC Berkeley since 1999. Prior to that, he was a Faculty Member of the University of Illinois at Urbana–Champaign from 1993 to 1999. He has published extensively, and holds more than a dozen patents. His current research interests are in distributed machine learning and cloud computing, statistical signal processing, and coding and information theory. He was a recipient of the 2017 IEEE Kobayashi Computers and Communications Award for his pioneering contributions to the theory and practice of distributed storage codes and distributed compression. He has received several awards for his research and teaching, including the IEEE Information Theory Society and Communication Society Joint Best Paper Award for 2012, the IEEE Communication Society Data Storage Best Paper Award in 2010, two best paper awards from the IEEE Signal Processing Society in 1993 and 1999, the Okawa Foundation Award for Outstanding Research at Berkeley in 2001, the Hank Magnusky Scholar Award at Illinois in 1998, and the EECS Departmental Outstanding Teaching Award at Berkeley in 2009.