Communication-Efficient and Model-Heterogeneous Personalized Federated Learning via Clustered Knowledge Transfer

Yae Jee Cho, Jianyu Wang, Tarun Chirvolu, and Gauri Joshi, *Member, IEEE*

***Abstract*—Personalized federated learning (PFL) aims to train model(s) that can perform well on the individual edge-devices' data where the edge-devices (clients) are usually IoT devices like our mobile phones. The participating clients for cross-device settings, in general, have heterogeneous system capabilities and limited communication bandwidth. Such practical properties of the edge-devices, however, are overlooked by many recent work in PFL, which use the same model architecture across all clients and incur high communication cost by directly communicating the model parameters. In our work, we propose a novel and practical PFL framework named COMET where clients can use heterogeneous models of their own choice and do not directly communicate their model parameters to other parties. Instead, COMET uses *clustered codistillation*, where clients use knowledge distillation to transfer their knowledge to other clients with similar data distributions. This presents a practical PFL framework for the edge-devices to train through IoT networks by lifting the heavy communication burden of communicating large models. We theoretically show the convergence and generalization properties of COMET and empirically show that COMET achieves high test accuracy with several orders of magnitude lower communication cost while allowing client model heterogeneity compared to the other state-of-the-art PFL methods.**

***Index Terms*—Federated learning, communication efficiency, model heterogeneity, knowledge transfer, clustering.**

I. Introduction

FEDERATED learning (FL) [1] has enabled the use of data on thousands of resource-constrained edge-devices (clients) like our mobile phones to train machine learning models without having to transfer the data to the cloud. Many recent work in FL [2], [3], [4] focuses on training a single global model with edge-clients via FL. However, due to the inherently high data heterogeneity across clients [5], [6], a single model that is trained to perform best in expectation for the sum of all participating clients' loss functions may not work well for each client [7], [8], [9]. This underwelming performance of the single global model trained via FL is exacerbated with the commonly used partial client selection in FL [10], [11]. Such limited generalization properties of the conventionally trained global models in FL calls for methods to train personalized models that can perform well on individual clients.

Several work investigated personalized FL (PFL) including applying meta-learning [12], training separate models on each client with weighted aggregation of other clients' models [13], using the global objective as a regularizer for training individual models at each client [14], or using model/data-interpolation with clustering for personalization [15]. However, all of these work impose models of identical architecture, i.e., homogeneous models, to be deployed across clients and require clients to directly communicate their model parameters. Such rigid constraints are limiting for FL with IoT devices in practice where the system capabilities (e.g., CPU, GPU memory, wireless resources) can be limited and heterogeneous across clients while the cost of communicating high-dimensional models with the server can be prohibitively high [16], [17], [18].

In this work, we relax the rigid constraints of having homogeneous models across clients and directly communicating the model parameters by proposing to train personalized models with *clustered codistillation*. Codistillation [22], [23], [24] performs distributed training across clients with reduced communication cost by only exchanging the models' predictions on a common unlabeled dataset instead of the model parameters. In conventional codistillation, a regularizing term is added to the standard cross-entropy loss of each client to penalize the client's prediction from being significantly different from the average of all clients' predictions. In FL, however, such conventional codistillation with the average of all clients' predictions cannot be directly applied since clients can have highly heterogeneous data. Thus, forcing each client to follow the average prediction of all clients can exacerbate its generalization by learning from clients that have significantly different data distributions [25].

Gaining insight from the limitations of the conventional codistillation methods, we propose a novel *clustered codistillation* framework for PFL named COMET that utilizes data correlation across clients. In our proposed COMET, each client uses the average prediction of only the clients that have similar data

Manuscript received 31 May 2022; revised 23 September 2022 and 25 November 2022; accepted 28 November 2022. Date of publication 6 January 2023; date of current version 17 February 2023. This work was supported in part by NSF under Grants CCF-2045694, CCF-2107085, and CNS-2112471. The guest editor coordinating the review of this manuscript and approving it for publication was Prof. Derrick Wing Kwan Ng. *(Corresponding author: Yae Jee Cho.)*

Yae Jee Cho, Jianyu Wang, and Gauri Joshi are with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: yaejeec@andrew.cmu.edu; jianyuw1@andrew.cmu.edu; gaurij@andrew.cmu.edu).

Tarun Chirvolu is with the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: tchiruvo@andrew.cmu.edu).

Digital Object Identifier 10.1109/JSTSP.2022.3231527

TABLE I
COMPARISON OF RELATED SOTA WORK IN PFL WITH OUR PROPOSED COMET

Related SOTA PFL Methods	Client Model Heterogeneity	Public Data	Full Client Participation	Model Parameters Communicated	Data Correlation Considered
PerFedAvg [12]	No	N/A	Not Required	Yes	No
FedFomo [13]	No	N/A	Not Required	Yes	Yes
Ditto [14]	No	N/A	Not Required	Yes	No
HypCluster [15]	No	N/A	Not Required	Yes	Yes
FedMD [19]	Yes	Labeled	Required	No	No
KT-pFL [20]	Yes	Unlabeled	Not Required	Yes	Yes
DS-FL [21]	Yes	Unlabeled	Not Required	No	No
COMET (ours)	**Yes**	**Unlabeled**	**Not Required**	**No**	**Yes**

distributions. COMET prevents clients from learning irrelvant knowledge from the less data correlated clients via clustering. As shown in Table I, to the best of our knowledge, **our work is the first to investigate a novel PFL framework for resource-constrained edge-devices that i) allows model heterogeneity, ii) improves communication-efficiency, and iii) utilizes data correlation.** We show throughout our paper that with theoretical guarantees, COMET is able to outperform most of the state-of-the-art PFL methods in terms of the achieved highest average test accuracies across clients with significantly reduced communication cost. In summary, COMET largely improves on the state-of-the-art PFL methods in the following ways:

1) Allows model heterogeneity across clients where the architecture and size of the model for local training can flexibly vary across the clients depending on their resource capabilities. Such facilitation of model heterogeneity is needed for scenarios in which clients participating in FL have varying and often limited system resources [18], [26].
2) Reduces the communication cost by several orders of magnitude via transferring logits instead of the high-dimensional model parameters between the clients and the server. Such communication-efficiency is proportionaly improved by how much larger the dimension of the model parameters is than that of the logit space.
3) Improves generalization performance for clients using data correlation, preventing clients from learning from less data correlated clients via clustered codistillation when there is data heterogeneity.

We further validate COMET's strength by presenting a theoretical analysis of COMET with its convergence guarantees and generalization properties. Our analysis shows that clustering by data correlation indeed improves the generalization properties of individual clients that are data heterogeneous. It also shows that each client can maximize its personalization performance by different degrees of regularization. Our experiments show that for both model homogeneous and model heterogeneous environments, COMET achieves high test accuracy with several orders of magnitude less communication compared to other SOTA FL methods. We show an overview of our proposed COMET's framework in Fig. 1.

Before presenting our proposed PFL framework COMET, we first review the background and related work in Section II, and then elaborate in detail on our proposed method COMET in Section III. We then provide the theoretical convergence and generalization guarantees of COMET in Section IV, and further demonstrate COMET's strong empirical performance in terms of test accuracy and communication efficiency in Section V. Lastly, we leave concluding remarks and potential future directions in Section VI.

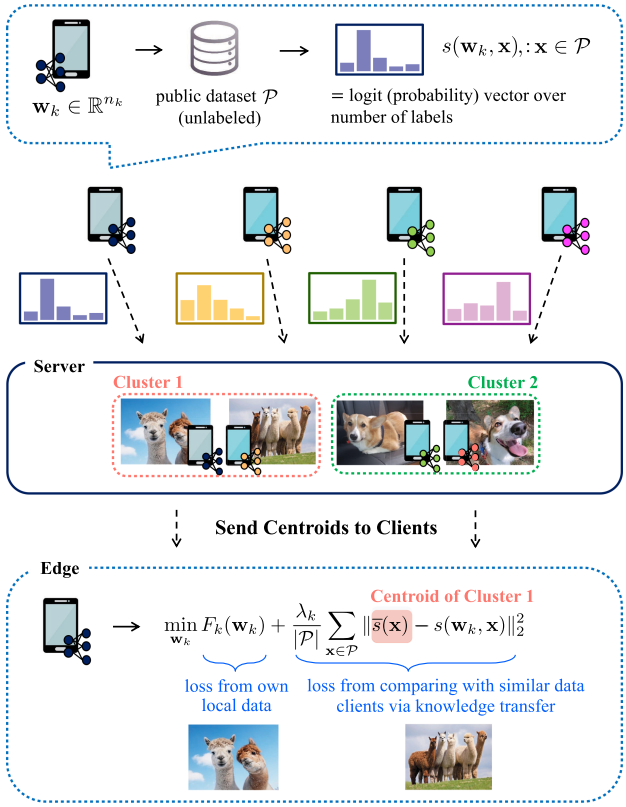


Fig. 1. Overview of Our Proposed PFL Framework COMET.

II. BACKGROUND AND RELATED WORK

A. Personalized Federated Learning (PFL)

In PFL, the goal is to train a single or several model(s) that can generalize well to each client's test dataset. In [12], using meta-learning for training a global model that better represents each client's data was proposed. A similar line of work [27] uses the moreau envelope as a regularizer for client local training.

Work in [13] proposed to find the optimal weighted combination of models from clients so that each client gets a model that better represents its target data distribution. The authors in [15] propose general approaches that can be applied to vanilla FL for personalization, including client clustering and data/model interpolation. These previous literature however, all requires homogeneous models and direct communication of the model parameters across the clients and server which can be highly infeasible for the resource-limited and system heterogeneous clients.

Several recent work [19], [21] have proposed using knowledge distillation (KD) for training personalized models with client model heterogeneity. However, [19] heavily relies on impractical assumptions for cross-device FL such as full client participation or having *large labeled public data*. Moreover, [19], [21] does not consider the data correlation across the clients where each client can improve personalization with the other clients with correlated data instead of those with largely different data distributions. A concurrent work [20] proposes utilizing the clients' data correlation using a coefficient matrix. However, the work imposes communicating high-dimensional models with an additional cost of communicating the coefficient matrix across clients. We show that COMET can overcome these constraints of the previous work with competitive performance outputs.

B. Knowledge Transfer and Codistillation

Conventional transfer learning is used to transfer knowledge from a model that has been trained on a larger-scale dataset to other models for other tasks where the available data may not be sufficient [28], [29]. Commonly used methods for transfer learning includes freezing the layers from the pretrained model and additionally training separate layers with the task of interest [30] or utilizing a separate dataset for distilling the knowledge from the pre-trained model to the other model through minimizing the distance between the models' outputs [31]. In our work, we focus on the latter method of knolwedge transfer well known as knowledge distillation (KD).

KD [31] has been prominently used to transfer knowledge from a pre-trained larger model to a smaller model [32], [33], [34], [35], [36], [37], [38]. Extending from this conventional KD, codistillation transfers knowledge across multiple models that are being trained concurrently. Specifically, each model is trained with the supervised loss with an additional regularizer term that encourages the model to yield similar outputs to the outputs of the other models that are also being trained.

Using codistillation for improved generalization in distributed training has recently been proposed in several works [22], [23], [24]. Authors of [22] have shown empirically that codistillation indeed improves generalization for distributed learning but often results in over-regularization, where the trained model's performance drops due to overfitting to the regularization term. In [23], codistillation was suggested for communication-efficient distributed training, but not in the PFL context where data can be highly heterogeneous across nodes and presented limited experiments on a handful of nodes with homogeneous data distributions across the nodes.

Codistillation in PFL presents a unique challenge in that each client's data distribution can be significantly different from other clients' distributions. Using standard codistillation with all of the clients' logits can lead to each client learning irrelevant information from other clients with different data distributions. We show this is indeed the case for PFL in both our theoretical and empirical results. We show that using clustering to find the clients that have similar data distributions with each other and performing codistillation within these clusters improves the personalized model's performance significantly for clients.

III. Proposed PFL Framework: COMET

A. Preliminaries

Consider a cross-device FL setup where K clients are connected to a central server. We consider a N-class classification task where each client $k \in [K]$ has its local training dataset $\mathcal{B}_k$ with $|\mathcal{B}_k| = m_k$ data samples. We denote $p_k = m_k / \sum_{k=1}^{K} m_k$ as the fraction of data for client k. Each data sample ξ is a pair $(\mathbf{x}, y)$ where $\mathbf{x} \in \mathbb{R}^d$ is the input and $y \in [1, N]$ is the label. The dataset $\mathcal{B}_k$ is drawn from the local data distribution $\mathcal{D}_k$, where we denote the empirical data distribution of $\mathcal{B}_k$ as $\widehat{\mathcal{D}}_k$. In the standard FL [1], clients aim to collaboratively find the model $\mathbf{w} \in \mathbb{R}^n$ that maps the input $\mathbf{x}$ to label y, such that $\mathbf{w}$ minimizes the empirical risk $F(\mathbf{w}) = \sum_{k=1}^{K} p_k F_k(\mathbf{w})$. The function $F_k(\mathbf{w})$ is the local objective of client k, defined as $F_k(\mathbf{w}) = \frac{1}{|\mathcal{B}_k|} \sum_{\xi \in \mathcal{B}_k} f(\mathbf{w}, \xi)$ with $f(\mathbf{w}, \xi)$ being the composite loss function.

With high data heterogeneity across clients, the optimal model parameters $\mathbf{w}^*$ that minimize the global objective $F(\mathbf{w})$ can generalize badly to the clients whose local objective $F_k(\mathbf{w})$ significantly differs from $F(\mathbf{w})$. Such clients may choose to opt out of FL, and instead train their own models of their choice $\mathbf{w}_k \in \mathbb{R}^{n_k}$ by minimizing their local objectives. This solo local training can work better than participating in FL for individual clients with large number of training samples (i.e., large m_k), since $\widehat{\mathcal{D}}_k$ becomes similar to $\mathcal{D}_k$, ensuring good generalization.

If, however, clients have only a few number of training samples [39], the distributions $\mathcal{D}_k$ and $\widehat{\mathcal{D}}_k$ can differ significantly, and therefore a model $\mathbf{w}_k$ trained only using the local dataset $\mathcal{B}_k$ can generalize badly. Indeed clients with small datasets are motivated to participate in FL, but they may not actually benefit from FL due to the bad generalization properties coming from other clients with significantly different data distributions. We show that with our proposed COMET, clients with small or large data samples both can improve generalization by being clustered with clients with similar data distributions.

In general, clients do not need to fully customize their model architecture, but rather make the choice out of practical circumstances such as computational memory and power that the clients' system capacity allows. For instance, for image classification, the clients can simply use commonly used ResNets and depending on their system capacity they can decide whether it will be the smallest ResNet or a larger one. It can be possible that the clients take one step further to fully customize the neural network architecture by methods such as neural architecture search [40], but this is orthogornal to COMET and is at the

TABLE II
LIST OF KEY NOTATIONS

K	Total number of clients
m	Number of selected clients per communication round
T	Total number of communication rounds
$\mathcal{B}_k$	Labeled private dataset of client $k \in [K]$
$\mathcal{P}$	Unlabeled public dataset
N	Number of classes for the classification task
$\mathbf{w}_k$	Classification models of each client $k \in [K]$
$s(\mathbf{w}_k, \mathbf{x})$	Logits of the model $\mathbf{w}_k$ over input data $\mathbf{x}$
$\mathbf{s}_k$	Logits $s(\mathbf{w}_k, \mathbf{x})$ stacked to rows for $\mathbf{x} \in \mathcal{P}$
$\alpha_{k,i}$	Client k's weight over the logits of other clients $i \in [K]$.
$\bar{\mathbf{s}}_k$	Weighted average of other clients' logits for client k.
λ_k	Weight of the Regularization Term for Client $k \in [K]$.

discretion of individual clients. For clarity of the paper we list the notation used for the paper Table II.

B. COMET Objective

In our proposed COMET framework, we use codistillation to train personalized models, where each client codistills with the other clients with similar, correlated model outputs with its own model output. Formally, clients have access to their private dataset $\mathcal{B}_k$ and a public dataset $\mathcal{P}$, consisting of *unlabeled* data. Inspired by the successful usage of knowledge transfer via unlabeled data [18], [31], [41] (accessible through data generators, open-sourced repositories, or data markets), the public dataset $\mathcal{P}$ is used as a reference dataset for codistillation across clients. Note that such unlabeled public datasets used in this work have been used commonly in previous work [17], [18], [20], [42] in FL. The classification models $\mathbf{w}_k$, $k \in [K]$ output soft-decisions (logits) over the pre-defined number of classes N, which is a probability vector over the N classes. We refer to the logits of model $\mathbf{w}_k$ over input data $\mathbf{x}$ in either the private or public dataset as $s(\mathbf{w}_k, \mathbf{x}) : \mathbb{R}^{n_k} \times (\mathcal{B}_k \cup \mathcal{P}) \to \Delta_N$, where Δ_N is the probability simplex over N. For notational simplicity, we define $\mathbf{s}_k \in \mathbb{R}^{|\mathcal{P}| \times N}$ as $s(\mathbf{w}_k, \mathbf{x}) \in \mathbb{R}^{1 \times N}, \mathbf{x} \in \mathcal{P}$ stacked into rows for each $\mathbf{x}$. We similarly define $\bar{\mathbf{s}}_k = \sum_{i=1}^{K} \alpha_{k,i} \mathbf{s}_i$.

The clients are connected via a central aggregating server. Each client seeks to find the model parameter $\mathbf{w}_k$ that minimizes the empirical risk $\Phi_k(\mathbf{w}_k; \bar{\mathbf{s}}_k)$, where $\Phi_k(\mathbf{w}_k; \bar{\mathbf{s}}_k)$ is a sum of the empirical risk of its own local training data $F_k(\mathbf{w}_k)$ and the regularization term as follows:

$$\Phi_k(\mathbf{w}_k; \bar{\mathbf{s}}_k) = F_k(\mathbf{w}_k) + \underbrace{\frac{\lambda_k}{|\mathcal{P}|} \sum_{\mathbf{x} \in \mathcal{P}} \|\bar{\mathbf{s}}_k(\mathbf{x}) - s(\mathbf{w}_k, \mathbf{x})\|_2^2}_{\text{regularization term}} \quad (1)$$

The term $\bar{\mathbf{s}}_k(\mathbf{x}) = \sum_{i=1}^{K} \alpha_{k,i} s(\mathbf{w}_i, \mathbf{x})$ denotes the weighted average of the logits from all clients for an arbitrary set of weights for client k, i.e., $\{\alpha_{k,i}\}_{i \in [K]}$ such that $\sum_{i=1}^{K} \alpha_{k,i} = 1$, $\forall k \in [K]$. The term λ_k modulates the weight of the regularization term. The weight $\alpha_{k,i}$ for each client i, $i \in [K]$ with respect to client k results from clustering the logits by the ℓ_2-norm distance. Thus clients with similar logits will have higher weights for each others' logits. The aggregated logit with weights, $\bar{\mathbf{s}}_k$, is calculated and sent by the server to the clients. Details of how the weights $\alpha_{k,i}, i \in [K]$ for each client k are calculated and how the logits are communicated are elaborated in detail in the subsequent subsections. Before going into details of the algorithm we first give more intuition on the formulation of the COMET objective in the next paragraphs.

Regularization Term: Without the regularization term in (1), minimizing $\Phi_k(\mathbf{w}_k; \bar{\mathbf{s}}_k)$ with regards to $\mathbf{w}_k$ is analogous to locally training in solo for minimizing $F_k(\mathbf{w}_k)$ for client k. If we have $\alpha_{k,i} = 1/K, \forall k, i \in [K]$, codistillation is implemented without clustering, using all of the clients' knowledge. We show in the next toy example and in a generalization bound derived for ensemble models for personalization in Appendix B of the significance of setting the values $\alpha_{k,i}, k, i \in [K]$ via clustering to improve personalization. Previous work in distributed ML [23] has proposed to utilize a similar regularization term as in (1) to improve communication-efficiency by only communicating logits and improving generalization across clients. However it does not take into account of the data heterogeneity across the clients and fixes $\bar{\mathbf{s}}_k(\mathbf{x})$ to be the simple average across all clients (i.e., $\alpha_{k,i} = 1/K, \forall k, i \in [K]$). In our work, we instead propose to utilize a weighted average across the clients to take into account the similarities of the data distribution of the clients for better peronalization of training the model $\mathbf{w}_k, k \in [K]$. We further show the improvement of such weighted averaging of the logits across the clients in the next paragraph with the example of linear regression.

Example with Linear Regression: We consider a toy example with linear regression where we have three clients with true models as in Fig. 2(a) where the true models 0 and 1 are similar to each other but the true model 2 is different from the other models. The global model trained from vanilla FedAvg does not match well with any true models as shown in Fig. 2(a). If we minimize (1) with respect to $\mathbf{w}_k$ without clustering, i.e., $\alpha_{k,i} = 1/K, \forall k, i \in [K]$, the output local model also diverges from the true model for each client, especially for client 0 and 1, due to the heterogeneity across the true models (see Fig. 2(b) (d)). Finally, if we minimize (1) with clustering so that for client k, higher weight $\alpha_{k,i}$ is given to the client i that has a similar true model to client k, and smaller weight is given to the other client that has a different true model, the output local model of client k gets close to its true model. This is further explored theoretically in our paper in the subsequent section in Theorem IV.7. COMET is based on this motivation where we set the weights for codistillation in $\bar{\mathbf{s}}_k(\mathbf{x}) = \sum_{i=1}^{K} \alpha_{k,i} s(\mathbf{w}_i, \mathbf{x})$ so that each client k sets higher $\alpha_{k,i}$, $i \in [K]$ for client i that has smaller difference between $s(\mathbf{w}_i, \mathbf{x})$ and $s(\mathbf{w}_k, \mathbf{x})$. Details of the setup for Fig. 2 are in Appendix B.

C. COMET Solver

We minimize (1) with respect to $\mathbf{w}_k$ for each client k on its own device with only communicating the logits instead of the actual model $\mathbf{w}_k$, with the server. With (t, r) denoting the communication round t and local iteration r, we define $\bar{\mathbf{s}}_k^{(t,0)} = \sum_{i=1}^{K} \alpha_{k,i}^{(t,0)} \mathbf{s}_i^{(t,0)}$ for $t \in [0, T-1]$ and $r \in [0, \tau - 1]$ where $\bar{\mathbf{s}}_k^{(t,0)}$ is fixed for all r and updated only for every t. The term T and τ is the total number of communication rounds

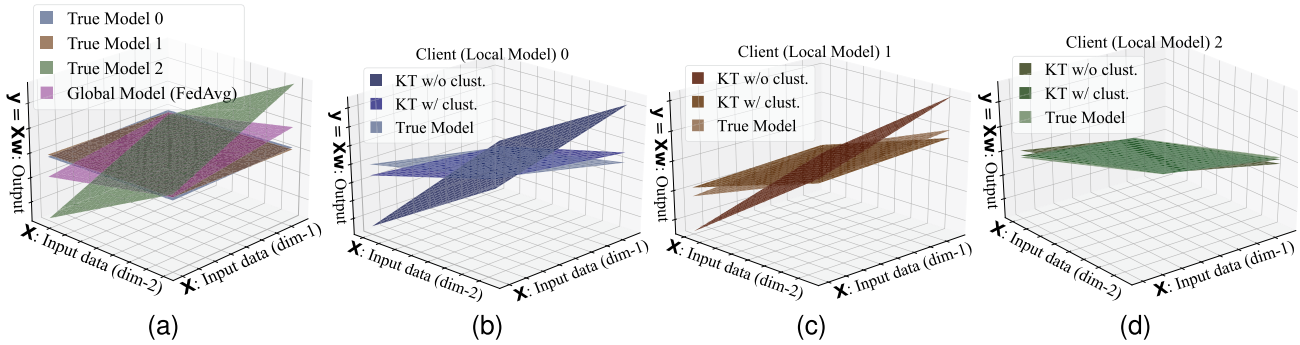


Fig. 2. Linear regression for given input data $\mathbf{X} \in \mathbb{R}^2$ and local (true) models $\mathbf{w} \in \mathbb{R}^{2\times 1}$ for three clients indexed by 0-2; (a): the true model for each client and the global model from FedAvg. The resulting global model does not match well with clients' true model; (b)-(d): the model for each client resulting from minimizing (1) (i.e., KT) with and without clustering (simple average of logits). COMET (KT w/ clust.) yields the model closest to the true model for all clients.

and local iterations respectively. Note that the logit information $\bar{\mathbf{s}}_k^{(t,0)}$ is computed and sent by the server to the clients for every communication round t. Details are in the following paragraphs and Algorithm 1.

Client Side Update: From (1), given $\bar{\mathbf{s}}_k^{(t,0)}$ from the server, each client's local update rule is:

$$\mathbf{w}_k^{(t,r+1)} = \mathbf{w}_k^{(t,r)} - \eta_t \left[\frac{1}{|\xi_k^{(t,r)}|} \sum_{\xi \in |\xi_k^{(t,r)}|} \nabla f(\mathbf{w}_k^{(t,r)}, \xi) \right.$$

$$\left. + \frac{2\lambda_k}{|\mathcal{P}_k^{(t,r)}|} \sum_{\mathbf{x}\in\mathcal{P}_k^{(t,r)}} \nabla s(\mathbf{w}_k^{(t,r)}, \mathbf{x})^T \left(s(\mathbf{w}_k^{(t,r)}, \mathbf{x}) - \bar{\mathbf{s}}_k^{(t,0)}(\mathbf{x}) \right) \right] \tag{2}$$

$$\triangleq \mathbf{w}_k^{(t,r)} - \eta_t \mathbf{g}_k(\mathbf{w}_k^{(t,r)}; \bar{\mathbf{s}}_k^{(t,0)}) \tag{3}$$

The term $\mathbf{w}_k^{(t,r)}$ denotes the local model parameters of client k, η_t is the learning rate, $\xi_k^{(t,r)}$ is the mini-batch randomly sampled from client k's local dataset $\mathcal{B}_k$, and $\mathcal{P}_k^{(t,r)}$ is the mini-batch randomly sampled from the public dataset from client k. We also denote the updated local model of client k after all τ local iterations for round t as $\mathbf{w}_k^{(t+1,0)} = \mathbf{w}_k^{(t,\tau)}$.

COMET uses partial client participation where for every communication round t, m clients are selected with probability p_k without replacement from $k \in [K]$. We denote the set of selected clients as $\mathcal{S}^{(t,0)}$ that is fixed for all local iterations $r \in [0, \tau-1]$. If a client $k \in [K]$ was most recently selected in the previous communication round $t' < t$, and selected again for the current communication round t, we assume that $\mathbf{w}_k^{(t,0)} = \mathbf{w}_k^{(t',\tau)}$. In other words, we retrieve the most recently updated local model for the client that is selected for the next communication round and use that model for local training. Each client $k \in \mathcal{S}^{(t,0)}$ takes $\tau \geq 1$ local updates before sending its logits back to the server where each local update step follows the update in (2).

Server Side Clustered Knowledge Aggregation: After the τ local iterations, each client $k \in \mathcal{S}^{(t,0)}$ sends the logits from its updated local model to the server. The logits are denoted as $\mathbf{s}_k^{(t+1,0)} \in \mathbb{R}^{|\mathcal{P}|\times N}$ which is $s(\mathbf{w}_k^{(t+1,0)}, \mathbf{x}) = s(\mathbf{w}_k^{(t,\tau)}, \mathbf{x}) \in \mathbb{R}^{1\times N}$ stacked in to rows for each $\mathbf{x} \in \mathcal{P}$. The server uses c-means clustering (also known conventionally as the k-means clustering algorithm [43]) to cluster the received m different set of logits, $\mathbf{s}_k^{(t+1,0)}, k \in \mathcal{S}^{(t)}$ to c clusters, where c is an integer such that $1 \leq c \leq m$. The server also gets the next set of selected clients $\mathcal{S}^{(t+1,0)}$ and sends the centroids $\{\mathbf{c}_i^{(t+1,0)}\}_{i\in[c]}$ for each cluster to the clients in $\mathcal{S}^{(t+1,0)}$. Each client $k' \in \mathcal{S}^{(t+1,0)}$ then determines the centroid that is closest to its current model's logit as follows

$$\bar{\mathbf{s}}_{k'}^{(t+1,0)} = \arg\min_{\{\mathbf{c}_i^{(t+1,0)}\}_{i\in[c]}} \|\mathbf{c}_i^{(t+1,0)} - \mathbf{s}_{k'}^{(t+1,0)}\|_2^2 \tag{4}$$

using it for the local update in (2). Since we defined $\bar{\mathbf{s}}_{k'}^{(t+1,0)} = \sum_{i=1}^K \alpha_{k',i}^{(t+1,0)} \mathbf{s}_i^{(t+1,0)}$, (4) gives a natural selection of $\alpha_{k',i}^{(t+1,0)}$

Algorithm 1: Proposed PFL Framework: COMET.

Input: mini-batch size b and b' for each private and public data, number of clusters c
Output: $\{\mathbf{w}_k\}_{k\in[K]}$
Initialize: $\{\mathbf{s}_k^{(0,0)}\}_{k\in\mathcal{S}^{(-1,0)}}$, set of m clients $\mathcal{S}^{(-1,0)}$
For $t = 0, \ldots, T-1$ **communication rounds do**:
  Global server do:
    Cluster $\{\mathbf{s}_k^{(t,0)}\}_{k\in\mathcal{S}^{(t-1,0)}}$ by c-means clustering
    Get centroids $\{\mathbf{c}_i^{(t,0)}\}_{i\in[c]}$ for each cluster
    Select m clients for $\mathcal{S}^{(t,0)}$ without replacement from $[K]$ by the dataset ratio $\{p_k\}_{k\in[K]}$
    Send centroids $\{\mathbf{c}_i^{(t,0)}\}_{i\in[c]}$ to clients $k \in \mathcal{S}^{(t,0)}$
  Clients $k \in \mathcal{S}^{(t,0)}$ **in parallel do:**
    Get $\mathbf{s}_k^{(t,0)}$ for current local model $\mathbf{w}_k^{(t,0)}$, and find $\bar{\mathbf{s}}_k^{(t,0)} = \arg\min_{\{\mathbf{c}_i^{(t,0)}\}_{i\in[c]}} \|\mathbf{c}_i^{(t,0)} - \mathbf{s}_k^{(t,0)}\|_2^2$.
    For $r = 0, \ldots, \tau-1$ **local iterations do:**
      Create mini-batch $\xi_k^{(t,r)}$ and $\mathcal{P}_k^{(t,r)}$ from sampling b and b' samples uniformly at random from $\mathcal{B}_k$ and $\mathcal{P}$ respectively
      Update $\mathbf{w}_k^{(t,r+1)} \leftarrow \mathbf{w}_k^{(t,r)} - \eta \mathbf{g}_k(\mathbf{w}_k^{(t,r)}; \bar{\mathbf{s}}_k^{(t,0)})$
    Send $\mathbf{s}_k^{(t+1,0)} = \mathbf{s}_k^{(t,\tau)}$ for the updated local model $\mathbf{w}_k^{(t,\tau)}$ back to the server

which gives higher weight to the $\mathbf{s}_i^{(t+1,0)}$, $i \in \mathcal{S}^{(t,0)}$ that is closer to client k''s logits, i.e., $\mathbf{s}_{k'}^{(t+1,0)}$. COMET can set $\alpha_{k,i} = 0$ for certain client i if its logit is significantly different from that of client k or if it was not included in the previous set of selected clients. It is also worth noting that COMET can seamlessly be applied to training a personalized model for new incoming clients by the server finding the right cluster for the new client and sending the centroid to the client so that it can use the logits for regularization.

IV. Theoretical Analysis of COMET

In this section, we analyze the convergence and generalization properties of COMET, highlighting the effect of clustered codistillation to the generalization performance.

A. Convergence Analysis

Here, we present the convergence guarantees of COMET with regards to the objective function $\Phi_k(\mathbf{w}_k^{(t,0)}; \bar{\mathbf{s}}_k^{(t,0)})$ as $t \to \infty$ with $\tau = 1$. We use the following assumptions for our analysis:

Assumption IV.1: The composite loss function $f(\mathbf{w}, \xi)$ is Lipschitz-continuous and Lipschitz-smooth for all $\mathbf{w}$, ξ, and therefore $F_1(\mathbf{w}), \ldots, F_k(\mathbf{w})$ are all L_f-continuous and L_p-smooth for all $\mathbf{w}$.

Assumption IV.2: Each $F_1, \ldots, F_k$ is bounded below by a scalar $F_{k,\inf}$ over its domain for $k \in [K]$.

Assumption IV.3: For the mini-batch ξ_k uniformly sampled at random from $\mathcal{B}_k$, the resulting stochastic gradient is unbiased, that is, $\mathbb{E}[\frac{1}{|\xi_k|}\sum_{\xi \in \xi_k} \nabla f(\mathbf{w}_k, \xi)] = \nabla F_k(\mathbf{w}_k)$.

Assumption IV.4: The stochastic gradient's expected squared norm is uniformly bounded, i.e., $\mathbb{E}\|\frac{1}{|\xi_k|}\sum_{\xi \in \xi_k} \nabla f(\mathbf{w}_k, \xi)\|^2 \leq G^2$ for $k = 1, \ldots, K$.

Assumption IV.5: $s(\mathbf{w}, \mathbf{x})$ is L_s-continuous and L_g-smooth for all $\mathbf{w}$ and x.

Now we present the convergence guarantees for COMETin Theorem IV.6 below:

Theorem IV.6: With Assumption IV.1-Assumption IV.5, after running COMET(Algorithm 1) for $t = T$ iterations on client $k \in [K]$ with K total clients participating, with the learning rate satisfying $\sum_{t=0}^{\infty} \eta_t = \infty$, $\sum_{t=0}^{\infty} \eta_t^2 < \infty$, we have that the norm of the gradient of $\Phi_k(\mathbf{w}_k^{(t,0)}; \bar{\mathbf{s}}_k^{(t,0)})$ with respect to $\mathbf{w}_k^{(t,0)}$ given $\bar{\mathbf{s}}_k^{(t,0)}$ goes to zero with probability 1 as $T \to \infty$, i.e., for every client k,

$$\lim_{t \to \infty} \|\nabla_{\mathbf{w}_k^{(t,0)}} \Phi_k(\mathbf{w}_k^{(t,0)}; \bar{\mathbf{s}}_k^{(t,0)})\| = 0 \tag{5}$$

The proof for Theorem IV.6 is deferred to Appendix A. Theorem IV.6 shows that our proposed algorithm COMET converges to a first-order stationary point with respect to $\mathbf{w}_k$ given $\bar{\mathbf{s}}_k$ where the norm of the gradient of our main objective function $\Phi_k(\mathbf{w}_k; \bar{\mathbf{s}}_k)$ with respect to $\mathbf{w}_k$ is 0.

B. Generalization Performance

Now, we show the theoretical grounds for clustered codistillation in regards to the generalization performance for PFL in the problem of linear regression. We also present a generalization bound for ensemble models in the context of personalization in Appendix B. For K clients, we consider a Bayesian framework as in [14] for linear regression where we have θ uniformly distributed on $\mathbb{R}^d$, and each client has its data distributed with parameters $\mathbf{w}_k = \theta + \zeta_k$ where $\zeta_k \sim \mathcal{N}(0, \upsilon_k^2 \mathbf{I}_d)$ and $\mathbf{I}_d$ is the $d \times d$ identity matrix and υ_k is unique to the client's task which is analogous to data heterogeneity in FL. Suppose we have $\mathbf{y}_k = \mathbf{X}_k \mathbf{w}_k + \mathbf{z}$, $k \in [K]$ where $\mathbf{y}_k \in \mathbb{R}^n$, $\mathbf{X}_k \in \mathbb{R}^{n \times d}$, and $\mathbf{z} \in \mathbb{R}^n$ such that $\mathbf{z} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_d)$.

Each client k has its empirical loss function as $F_k(\mathbf{w}_k) = \|\mathbf{X}_k \mathbf{w}_k - \mathbf{y}_k\|_2^2$ with $\widehat{\mathbf{w}}_k = (\mathbf{X}_k^T \mathbf{X}_k)^{-1} \mathbf{X}_k^T \mathbf{y}_k$ being a noisy observation of $\mathbf{w}_k$ with additive covariance $\sigma^2 (\mathbf{X}_k^T \mathbf{X}_k)^{-1}$ since $\widehat{\mathbf{w}}_k \sim \mathcal{N}((\mathbf{X}_k^T \mathbf{X}_k)^{-1} \mathbf{X}_k^T \mathbf{y}_k, \sigma^2 (\mathbf{X}_k^T \mathbf{X}_k)^{-1})$. Then with Lemma 2 [14], with the following definitions:

$$\Sigma_k := \sigma^2 (\mathbf{X}_k^T \mathbf{X}_k)^{-1} + \upsilon_k^2 \mathbf{I}_d \tag{6}$$

$$\overline{\Sigma}_{\setminus k} := \left(\sum_{i \in [K], i \neq k} \Sigma_i^{-1} \right)^{-1} \tag{7}$$

$$\overline{\theta}_{\setminus k} := \overline{\Sigma}_{\setminus k} \sum_{i \in [K], i \neq k} \Sigma_i^{-1} \widehat{\mathbf{w}}_i \tag{8}$$

given $\{\mathbf{X}_i, \mathbf{y}_i\}_{i \in [K], i \neq k}$ we have that

$$\theta = \overline{\theta}_{\setminus k} + \gamma \tag{9}$$

where $\gamma \sim \mathcal{N}(0, \overline{\Sigma}_{\setminus k})$. Further, if we let

$$\widetilde{\Sigma}_k := \overline{\Sigma}_{\setminus k} + \upsilon_k^2 \mathbf{I}_d \tag{10}$$

$$\overline{\Sigma}_k := \left((\widetilde{\Sigma}_k)^{-1} + (\sigma^2 (\mathbf{X}_k^T \mathbf{X}_k)^{-1})^{-1} \right)^{-1} \tag{11}$$

given $\{\mathbf{X}_i, \mathbf{y}_i\}_{i \in [K]}$, we have

$$\mathbf{w}_k = \overline{\Sigma}_k (\sigma^2 (\mathbf{X}_k^T \mathbf{X}_k)^{-1})^{-1} \widehat{\mathbf{w}}_k + \overline{\Sigma}_k (\widetilde{\Sigma}_k)^{-1} \overline{\theta}_{\setminus k} + \vartheta_k \tag{12}$$

where $\vartheta_k \sim \mathcal{N}(0, \overline{\Sigma}_k)$. The term for $\mathbf{w}_k$ in (12) uses the fact that $\widehat{\mathbf{w}}_k$ is a noisy observation of $\mathbf{w}_k$ with additive noise of zero mean and covariance $\sigma^2 (\mathbf{X}_k^T \mathbf{X}_k)^{-1}$, and $\overline{\theta}_{\setminus k}$ is a noisy observation of θ with covariance $\overline{\Sigma}_{\setminus k}$. Given all training samples from the K clients, $\mathbf{w}_k$ in (12) is Bayes optimal.

With COMET, following (1), we solve the objective:

$$\min_{\mathbf{w}_k} \|\mathbf{X}_k \mathbf{w}_k - \mathbf{y}_k\|_2^2 + \lambda_k \|\bar{\mathbf{s}}_k - s(\mathbf{w}_k)\|_2^2 \tag{13}$$

where λ_k is the regularization term as in (1) and $\bar{\mathbf{s}}_k$ and $s(\mathbf{w}_k)$ each is comparative to the $\bar{\mathbf{s}}_k(\mathbf{x})$ and $s(\mathbf{w}_k, \mathbf{x})$ in (1) for a single public data point $\mathbf{x}$. Note that in the setting of linear regression we can set $s(\mathbf{w}_k) = \mathbf{P}\mathbf{w}_k$ where $\mathbf{P} \in \mathbb{R}^{1 \times d}$ is the public data (without loss of generality, we assume single data point for the public data for simplicity). Accordingly, we set $\bar{\mathbf{s}}_k = \sum_{i=1}^K \alpha_{k,i} s(\widehat{\mathbf{w}}_i)$ for an arbitrary set of weights $\alpha_{k,i}, i \in [K]$ for client k. Then we have that the local empirical risk minimizer for (13) is

$$\widetilde{\mathbf{w}}_k = (\mathbf{X}_k^T \mathbf{X}_k + \lambda_k \mathbf{P}^T \mathbf{P})^{-1} + \left(\mathbf{X}_k^T \mathbf{X}_k \widehat{\mathbf{w}}_k \lambda_k \mathbf{P}^T \mathbf{P} \sum_{i=1}^K \alpha_{k,i} \widehat{\mathbf{w}}_i \right) \tag{14}$$

Finally, we present the optimal λ_k^* and $\alpha_{k,i}^*$ for any client $k \in [K]$ given the linear regression problem.

Theorem IV.7: Assuming $\mathbf{X}_k^T\mathbf{X}_k = \beta\mathbf{I}_d$ and $\mathbf{P}^T\mathbf{P} = \nu\mathbf{I}_d$ for some constant β, ν, the λ_k^* and $\alpha_{k,i}^*, i \in [K]$ that minimizes the test performance on client k, $k \in [K]$ i.e.,

$$\lambda_k^*, \alpha_{k,i}^*, i \in [K] = \arg\min_{\lambda_k, \alpha_{k,i}, i\in[K]} \mathbb{E}[F_k(\widetilde{\mathbf{w}}_k)|\widehat{\mathbf{w}}_k, \overline{\theta}_{\backslash k}] \tag{15}$$

we have that

$$\lambda_k^* = \sigma^2/\upsilon_k^2\nu,\ \alpha_{k,i}^* = \frac{B_k}{\sigma^2 + \beta\upsilon_i^2} \tag{16}$$

with $A_k = \left(\sum_{i\in[K], i\neq k} \frac{1}{\sigma^2+\beta\upsilon_i^2}\right)^{-1}$, $B_k = \frac{A_k(\sigma^2+\beta\upsilon_k^2)}{\sigma^2+A_k\beta\upsilon_k^2}$.

The proof for Theorem IV.7 is deferred to Appendix B. Theorem IV.7 shows the optimal weights $\{\alpha_{k,i}^*\}_{i\in[K]}$ and λ_k^* for each client $k \in [K]$ given the objective function (13) and the corresponding minimizer (14), to maximize generalization with COMET. We elaborate on the implications of Theorem IV.7 in the subsequent paragraphs.

Clustering with Data Correlation: With data heterogeneous clients where υ_k, $k \in [K]$ is unique to each client $k \in [K]$, we have that the optimal weights $\alpha_{k,i}^* = \frac{B_k}{\sigma^2+\beta\upsilon_i^2}$ for clients $i \in [K]$ are inversely proportional to υ_i. Intuitively, since larger υ_i leads to a larger divergence from the original θ for client i due to $\mathbf{w}_i = \theta + \zeta_i$, $\zeta_i \sim \mathcal{N}(0, \upsilon_i^2\mathbf{I}_d)$, giving a lower weight $\alpha_{k,i}$ to this client i, improves generalization of the personalzied model for client k. This gives new insight into codistillation for PFL since previous work [19], [23] only consider scenarios where the weights do not consider data correlation, i.e., $\alpha_{i,k} = 1/K$, $\forall i, k \in [K]$ for a non-personalized FL setting. This result corroborates COMET's motivation for clustered codistillation with data correlation for improved generalization performance of the personalized models.

Regularization Weights: The optimal regularization weights are equal to $\lambda_k^* = \sigma^2/(\upsilon_k^2\nu)$, $k \in [K]$, where σ^2 and ν are constant across clients. This shows that clients with large υ_k can improve its generalization performance by having a smaller regularization weight. Since clients with larger υ_k have larger data distribution discrepancies with other clients, having a smaller λ_k can prevent them from assimilating irrelevant knowledge from the other clients. This gives insight into how to set the regularization weight dependent on the client's data discrepancy to other clients. Although in our experiments we use identical λ_k for $k \in [K]$, interesting future directions include varying λ_k of across clients dependent on their data and training progress.

V. Experiments

We demonstrate the efficacy of COMET in terms of the average test accuracy across all clients with the communication cost defined as the total number of parameters communicated across server/clients during the training process including uplink and downlink. All experiments are conducted on 3 different random seeds, and the standard deviations across trials are shown in the parentheses. Further details of the experimental setup are in Appendix B.

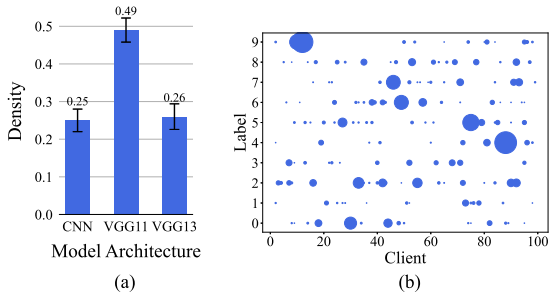


Fig. 3. (a) Proportion of the models architectures deployed across clients for the model heterogeneity scenario; (b) data distribution with $\alpha = 0.01$ for all clients where larger circle indicates larger dataset size for CIFAR10.

A. Experimental Setup

Datasets and models: We evaluate COMET with two different image classification tasks: i) CIFAR10 as the training dataset with CIFAR100 as the public dataset and ii) CIFAR100 [44] as the training data with TinyImagenet [45] as the public dataset. For the training data, we partition data heterogeneously amongst clients using the Dirichlet distribution $\text{Dir}_K(\alpha)$ [46] (smaller α leads to higher data size imbalance and degree of label skew across clients) with $\alpha = 0.01$ to emulate realistic FL with large data heterogeneity (see Fig. 3(b) in Appendix B). For the public dataset, for each task, we only use 2000 of *unlabeled data samples* that are sampled uniformly at random without replacement from the entire public dataset prior to beginning training. We fix the image size to be 32×32 for all tasks. We evaluate COMET with two different scenarios: model homogeneous and heterogeneous. For the model homogeneous scenario, VGG11 [47] is used for all clients. For the model heterogeneous scenario, we sample one of the VGG13/VGG11/CNN model architecture for each client with the probability of a larger model getting assigned to a client is proportional to the client's dataset size (see Fig. 3(a) in Appendix B).

Baselines: We compare COMET with SOTA FL algorithms designed to train either (i) a non-personalized single global model at the server (e.g. FedAvg, FedProx, Scaffold, FedDF) or (ii) personalized model(s) either at the server side as a global model (GM) or client side as a local model (LM) (e.g., Per-FedAvg, Ditto, KT-pFL, DS-FL, FedMD, FedFomo, HypCluster). We do a grid search over the hyperparameters to find the best performing parameters accordingly (see Appendix B). For FedMD which requires training directly on a *labeled* public dataset, we use the available labels in the corresponding public dataset. For fair comparison across different benchmarks, we do not apply any momentum acceleration or weight decay to local training.

B. Experimental Results

Communication-Efficiency of COMET[1]*:* In Table III, we show the performance of COMET compared with the other

[1]Downloading the public dataset is a one-time cost handeled prior to beginning of training. This is a rather small cost (only 2000 unlabeled samples with image size 32×32), compared to the cost of communicating both uplink and downlink the model parameters for every communication round.

TABLE III
AVERAGE TEST ACCURACY OF ALL CLIENTS AND TOTAL COMMUNICATION COST (NUMBER OF PARAMETERS COMMUNICATED IN TOTAL) FOR THE MODEL-HOMOGENEOUS SCENARIO (VGG11) AND TRAINING DATA CIFAR10 WITH PUBLIC DATA CIFAR100

		$C = 0.1$		$C = 0.15$	
Method	Algorithm	Test Acc.	Com.-Cost	Test Acc.	Com.-Cost
Local Training	-	64.02 ($\pm$0.48)	-	64.02 ($\pm$0.48)	-
Non-Personalized	FedAvg	15.53 ($\pm$1.42)	2150×10^7	20.00 ($\pm$2.66)	3120×10^7
	FedProx	13.64 ($\pm$1.23)		16.71 ($\pm$1.79)	
	Scaffold	12.92 ($\pm$1.46)		16.41 ($\pm$1.14)	
	FedDF	15.18 ($\pm$1.18)		17.09 ($\pm$1.54)	
Personalized	Per-FedAvg (GM)	14.47 ($\pm$0.59)		14.61 ($\pm$0.83)	
	Per-FedAvg (LM)	51.74 ($\pm$1.52)		50.25 ($\pm$1.61)	
	Ditto (LM)	67.21 ($\pm$1.86)		68.88 ($\pm$1.95)	
	Ditto (GM)	21.63 ($\pm$2.13)		19.16 ($\pm$2.34)	
	KT-pFL	72.59 ($\pm$0.33)		74.27 ($\pm$0.45)	
	DS-FL	63.27 ($\pm$0.58)	4.4×10^7	65.19 ($\pm$0.32)	6.4×10^7
	FedMD ($C = 1.0$)	64.02 ($\pm$0.48)	6.1×10^7	N/A	N/A
	FedFomo	**74.62** ($\pm$0.42)	$\mathbf{3900 \times 10^7}$	**77.56** ($\pm$0.75)	$\mathbf{5850 \times 10^7}$
	HypCluster	65.77 ($\pm$2.76)	3120×10^7	63.28 ($\pm$1.16)	4100×10^7
	COMET($c = 1$)	70.70 ($\pm$0.46)	4.4×10^7	67.66 ($\pm$0.30)	6.4×10^7
	COMET($c = 2$)	73.33 ($\pm$0.26)	4.8×10^7	70.86 ($\pm$0.73)	6.8×10^7
	COMET($c = 3$)	**74.31** ($\pm$0.40)	$\mathbf{5.2 \times 10^7}$	**76.74** ($\pm$0.71)	$\mathbf{7.2 \times 10^7}$
	COMET($c = 4$)	72.67 ($\pm$0.31)	5.6×10^7	73.52 ($\pm$1.15)	7.6×10^7

We train for 200 communication rounds with total number of clients $K = 100$ and selected clients portion as $C \in \{0.1, 0.15\}$ except for FedMD which only supports $C = 1.0$.

TABLE IV
AVERAGE TEST ACCURACY ACROSS ALL CLIENTS AND TOTAL COMMUNICATION COST (NUMBER OF PARAMETERS COMMUNICATED IN TOTAL) FOR THE MODEL-HETEROGENEOUS (VGG13/VGG11/CNN) SCENARIO FOR 200 COMMUNICATION ROUNDS WITH TOTAL NUMBER OF CLIENTS $K = 100$ AND SELECTED PORTION OF CLIENTS $C = 0.10$ EXCEPT FOR FEDMD WHICH ONLY SUPPORTS $C = 1.0$

		CIFAR10/CIFAR100		CIFAR100/TinyImagenet	
Method	Algorithm	Test Acc.	Com.-Cost	Test Acc.	Com.-Cost
Local Training	-	59.29 ($\pm$1.05)	-	29.11 ($\pm$1.52)	-
Personalized	KT-pFL	71.82 ($\pm$0.38)	1780×10^7	47.68 ($\pm$0.43)	1780×10^7
	DS-FL	60.89 ($\pm$0.82)	4.4×10^7	32.75 ($\pm$0.88)	8.8×10^7
	FedMD ($C = 1.0$)	61.05 ($\pm$0.75)	6.1×10^7	35.32 ($\pm$0.79)	12×10^7
	COMET($c = 1$)	70.94 ($\pm$0.46)	4.4×10^7	45.02 ($\pm$0.35)	8.8×10^7
	COMET($c = 2$)	**72.25** ($\pm$0.17)	$\mathbf{4.8 \times 10^7}$	46.10 ($\pm$0.48)	9.6×10^7
	COMET($c = 3$)	71.84($\pm$0.40)	5.2×10^7	**48.59** ($\pm$0.51)	$\mathbf{10 \times 10^7}$
	COMET($c = 4$)	70.01 ($\pm$0.31)	5.6×10^7	46.35 ($\pm$0.52)	11×10^7

SOTA algorithms in regards to the achieved highest test accuracy and communicated number of parameters between server and client with different fraction of selected clients per round ($C \in \{0.1, 0.15\}$). For $C = 0.1$, COMET achieves high test accuracy of 74.31% at the number of clusters $c = 3$, with small communication cost compared to other algorithms (saving at maximum $\times$750). FedFomo achieves a slightly higher test accuracy performance with 74.62%, but the communication cost spent (3900×10^7 parameters) is significantly larger than COMET (5.2×10^7 parameters). Similarly, KT-pFL is able to achieve a comparable test accuracy of 72.59%, but with significantly larger communication cost than COMET ($\times$ 414). Algorithms that train a non-personalized single global model performs strictly worse than all personalized algorithms showing that the traditional FL framework does not perform well to individual clients in the setting of high data heterogeneity. For $C = 0.15$, COMET also achieves a comparable high test accuracy of 76.74% with only a small communication cost of 7.2×10^7 parameters (saving at maximum $\times$813) where FedFomo achieves a slightly higher accuracy of 77.56% with larger communication cost of 5850×10^7 parameters.

Client Model Heterogeneity: We also demonstrate the performance of COMET when clients have heterogeneous models dependent on their dataset size (see Fig. 3(a) in Appendix B) in Table IV. Note that this is a realistic setting of FL where clients can have smaller or larger models dependent on their dataset size or system capabilities. KT-pFL, DS-FL, and FedMD allows client model heterogeneity amongst the SOTA PFL algorithms which we set as baselines. With client model heterogeneity, COMET achieves the highest test accuracy

of 72.25% and 48.59% for training/public dataset CIFAR10/CIFAR100 and CIFAR100/TinyImagenet respectively, with the communication cost being each 4.8×10^7 and 10×10^7. Specifically, COMET outperforms KT-pFL by around 1% while saving at maximum $\times 371$ on communication cost. The test accuracy of COMET with model heterogeneity is close to that of COMET with model homogeneity, showing that while model heterogeneity increases the feasibility of COMET, it does not largely hurt the local performance of the clients.

Data Correlation Within the Clusters: COMET performs clustering at the server side to cluster the logits received from different clients by a number of clusters $c \geq 1$. For $c > 1$, the logits from similar data distributions are aggregated as the centroid for each cluster. The degree of data correlation imposed within each cluster is in fact modulated by the number of clusters c in COMET, where $c = 1$ equals to leveraging no data correlation. By varying c for COMET, we evaluate how the data correlation within the clusters effects COMET's performance. For both $C = 0.1$ and $C = 0.15$ in Table III, $c = 3$ achieves the best test accuracy performance, and with increased $c > 3$ the test accuracy drops with higher communication cost. This shows that while increasing data correlation within each cluster helps to a certain extent, exceeding a certain threshold hurts generalization since we are decreasing the diversity of information included within each cluster. Similar effect of c on the performance of COMET is observed in Table IV.

a) Determining the Number of Clusters c: Although the number of clusters c can be treated as a hyperparameter that can be tuned on validation datasets, we provide some insight into how to decide the number of clusters. Specifically, in scenarios where we know roughly how the data distribution is heterogeneously distributed across clients, we can set c as the number of clusters where clients can be roughly grouped by their data distributions. In scenarios where we do not now how the data distribution is distributed across clients but know the intensity of the data heterogeneity across clients, we can set c in proportion to such instensity where if the data distribution is homogeneous across clients setting $c = 1$ will intuitively be the most effective while if the data distribution is largely heterogeneous across clients setting $1 << c < CK$ will be effective where C is the portion of clients selected every communication round and K is the total number of clients.

VI. Limitations and Concluding Remarks

Data and system heterogeneity across the resource-limited clients in FL are critical factors to be considered for devising PFL algorithms. Many previous work have imposed clients to have homogeneous models with direct communication of their model parameters which can incur heavy communication cost. Our proposed COMET caters to the clients' data and system heterogeneity with clustered codistillation, allowing heterogeneous models on clients without the direct communication of the model parameters. Accordingly, COMET achieves competitive performance against SOTA FL methods with smaller communication cost. Along with these advantages, COMET entails few key limitations such as requiring a small label space and public data that is relevant to the training task in order for it to be communication-efficient and effective in personalization. For instance, having a too large label space such as the Glink360 K dataset [48] may make COMET no longer communication efficient. Moreover, a public data that is entirely out-of-domain of the task of interest may not improve the generalization performance of the clients. Future directions include tackling such limitations and also understanding the privacy implications of COMET and the optimal degree of data correlation to maximize the performance of the clients' personalized models.

Appendix

A. Proof for Theorem IV.6

In this section we present the proof for Theorem IV.6. We follow the techniques presented by [23] for the proof. For notational simplicity, we notate all super subscript $(t, 0)$ as (t) throughout the proof, dropping the local iteration index. We define the following σ-algebra on the set that contains the history of the model updates for all clients with $\mathbf{w}^{(t)} = [\mathbf{w}_1^{(t)} \dots \mathbf{w}_K^{(t)}]$ and $\overline{\mathbf{s}}^{(t)} = [\overline{\mathbf{s}}_1^{(t)} \dots \overline{\mathbf{s}}_K^{(t)}]$ as

$$\mathcal{H}_t = \sigma(\{\mathbf{w}^{(i)}, \overline{\mathbf{s}}^{(i)}\} \,|\, i \leq t) \tag{17}$$

and recall

$$\mathbf{g}_k(\mathbf{w}_k^{(t)}; \overline{\mathbf{s}}_k^{(t)}) \triangleq \frac{1}{|\xi_k^{(t)}|} \sum_{\xi \in \xi_k^{(t)}} \nabla f(\mathbf{w}_k^{(t)}, \xi) + \frac{2\lambda}{|\mathcal{P}_k^{(t)}|} \sum_{\mathbf{x} \in \mathcal{P}_k^{(t)}} \nabla s(\mathbf{w}_k^{(t)}, \mathbf{x})^T \left(s(\mathbf{w}_k^{(t)}, \mathbf{x}) - \overline{\mathbf{s}}_k^{(t)}(\mathbf{x}) \right) \tag{18}$$

1) Additional Lemmas: We first present Lemmas and their proofs which we use for the intermediate steps in the main proof for Theorem IV.6.

Lemma A.1: The gradient of the second term in $\Phi_k(\mathbf{w}_k^{(t)}; \overline{\mathbf{s}}_k^{(t)})$ with respect to $\mathbf{w}_k^{(t)}$ is Lipschitz continuous, and therefore with Assumption IV.1, $\Phi_k(\mathbf{w}_k^{(t)}; \overline{\mathbf{s}}_k^{(t)})$ is also a Lipschitz-smooth function with factor L_p.

Proof: With dropping the iteration index t for the upper script for simplicity, let's define the second term in $\Phi_k(\mathbf{w}_k; \overline{\mathbf{s}}_k)$ as

$$q(\mathbf{w}_k; \overline{\mathbf{s}}_k) \triangleq \frac{\lambda}{|\mathcal{P}|} \sum_{\mathbf{x} \in \mathcal{P}} \|\overline{\mathbf{s}}(\mathbf{x}) - s(\mathbf{w}_k, \mathbf{x})\|_2^2 \tag{19}$$

Then we have

$$\nabla_{\mathbf{w}_k} q(\mathbf{w}_k; \overline{\mathbf{s}}_k) = \frac{2\lambda}{|\mathcal{P}|} \sum_{\mathbf{x} \in \mathcal{P}} \nabla s(\mathbf{w}_k, \mathbf{x})^T \left(s(\mathbf{w}_k, \mathbf{x}) - \overline{\mathbf{s}}_k(\mathbf{x}) \right) \tag{20}$$

For an arbitrary $\mathbf{e}_k$ in $q(\cdot; \overline{\mathbf{s}}_k)$ for $\mathbf{x} \in \mathcal{P}$, we have

$$\begin{aligned} &\| \nabla s(\mathbf{w}_k, \mathbf{x})^T \\ &\times \left(s(\mathbf{w}_k, \mathbf{x}) - \overline{\mathbf{s}}_k(\mathbf{x}) \right) - \nabla s(\mathbf{e}_k, \mathbf{x})^T \\ &\times \left(s(\mathbf{e}_k, \mathbf{x}) - \overline{\mathbf{s}}_k(\mathbf{x}) \right) \|^2 \leq 3 \| \nabla s(\mathbf{w}_k, \mathbf{x})^T \\ &\times \left(s(\mathbf{w}_k, \mathbf{x}) - \overline{\mathbf{s}}_k(\mathbf{x}) \right) \end{aligned}$$

$$
\begin{aligned}
&- \nabla s(\mathbf{w}_k, \mathbf{x})^T \left(s(\mathbf{e}_k, \mathbf{x}) - \bar{\mathbf{s}}_k(\mathbf{x})\right) \|^2 + 3\| \\
&\times \nabla s(\mathbf{w}_k, \mathbf{x})^T (s(\mathbf{e}_k, \mathbf{x}) \\
&- \bar{\mathbf{s}}_k(\mathbf{x})) - \nabla s(\mathbf{e}_k, \mathbf{x})^T \left(s(\mathbf{e}_k, \mathbf{x}) - \bar{\mathbf{s}}_k(\mathbf{x})\right) \|^2 \qquad (21) \\
&\leq 3\|\nabla s(\mathbf{w}_k, x)\|^2 \|s(\mathbf{w}_k, x) - s(\mathbf{e}_k, x)\|^2 + 3\|\nabla s(\mathbf{w}_k, x) \\
&- \nabla s(\mathbf{e}_k, x)\|^2 \|s(\mathbf{e}_k, x) - \bar{\mathbf{s}}_k(x)\|^2 \qquad (22) \\
&\leq 3L_s^4 \|\mathbf{w}_k - \mathbf{e}_k\|^2 + 6L_g^2 \|\mathbf{w}_k - \mathbf{e}_k\|^2 \qquad (23) \\
&= (3L_s^4 + 6L_g^2)\|\mathbf{w}_k - \mathbf{e}_k\|^2 \qquad (24)
\end{aligned}
$$

where (21) uses Jensen's inequality for the ℓ_2-norm for three terms, (22) uses the submultiplicativity of the norm, and the LHS of (23) uses Assumption IV.5. Therefore we can conclude that $\|\nabla s(\mathbf{w}_k, \mathbf{x})^T (s(\mathbf{w}_k, \mathbf{x}) - \bar{\mathbf{s}}_k(\mathbf{x})) - \nabla s(\mathbf{e}_k, \mathbf{x})^T (s(\mathbf{e}_k, \mathbf{x}) - \bar{\mathbf{s}}_k(\mathbf{x}))\|^2$ for any $\mathbf{x}$ is Lipschitz-continuous, and hence $\nabla_{\mathbf{w}_k} q(\mathbf{w}_k; \bar{\mathbf{s}}_k)$ is also Lipschitz-continuous. □

Lemma A.2: We have that $\mathbb{E}[\mathbf{g}_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})|\mathcal{H}_t] = \nabla_{\mathbf{w}_k^{(t)}} \Phi_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})$ and $\mathbb{E}[\|\mathbf{g}_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})\|_2^2] \leq 2\,G^2 + 16\lambda^2 L_s^2$ and $\|\nabla_{\mathbf{w}_k^{(t)}} \Phi_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})\|$ and $\|\mathbf{g}_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})\|$ is each bounded by constant $M_1 \geq 0$ and $M_2 \geq 0$.

Proof: By definition of the gradient we have

$$
\begin{aligned}
&\mathbb{E}[\mathbf{g}_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})|\mathcal{H}_t] = \mathbb{E}\left[\frac{1}{|\xi_k^{(t)}|} \sum_{\xi \in \xi_k^{(t)}} \nabla f(\mathbf{w}_k^{(t)}, \xi)\right. \\
&\left. + \frac{2\lambda}{|\mathcal{P}_k^{(t)}|} \sum_{\mathbf{x} \in \mathcal{P}_k^{(t)}} \nabla s(\mathbf{w}_k^{(t)}, \mathbf{x})^T (s(\mathbf{w}_k^{(t)}, \mathbf{x}) - \bar{\mathbf{s}}_k^{(t)}(\mathbf{x}))|\mathcal{H}_t\right] \qquad (25) \\
&= \nabla F_k(\mathbf{w}_k^{(t)}) + \frac{2\lambda}{|\mathcal{P}|} \sum_{\mathbf{x} \in \mathcal{P}} \nabla s(\mathbf{w}_k^{(t)}, \mathbf{x})^T (s(\mathbf{w}_k^{(t)}, \mathbf{x}) - \bar{\mathbf{s}}_k^{(t)}(\mathbf{x})) \qquad (26) \\
&= \nabla_{\mathbf{w}_k^{(t)}} \Phi_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)}) \qquad (27)
\end{aligned}
$$

finishing the proof for the first part of Lemma A.2. Next, we prove the second part of Lemma A.2 showing that

$$
\begin{aligned}
&\mathbb{E}[\|\mathbf{g}_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})\|^2] = \mathbb{E}\left[\|\frac{1}{|\xi_k^{(t)}|} \sum_{\xi \in \xi_k^{(t)}} \nabla f(\mathbf{w}_k^{(t)}, \xi)\right. \\
&\left. + \frac{2\lambda}{|\mathcal{P}_k^{(t)}|} \sum_{\mathbf{x} \in \mathcal{P}_k^{(t)}} \nabla s(\mathbf{w}_k^{(t)}, \mathbf{x})^T (s(\mathbf{w}_k^{(t)}, \mathbf{x}) - \bar{\mathbf{s}}_k^{(t)}(\mathbf{x}))\|^2\right] \qquad (28) \\
&\leq 2\mathbb{E}\left[\|\frac{1}{|\xi_k^{(t)}|} \sum_{\xi \in \xi_k^{(t)}} \nabla f(\mathbf{w}_k^{(t)}, \xi)\|^2\right] \\
&+ 2\mathbb{E}\left[\|\frac{2\lambda}{|\mathcal{P}_k^{(t)}|} \sum_{\mathbf{x} \in \mathcal{P}_k^{(t)}} \nabla s(\mathbf{w}_k^{(t)}, \mathbf{x})^T (s(\mathbf{w}_k^{(t)}, \mathbf{x}) - \bar{\mathbf{s}}_k^{(t)}(\mathbf{x}))\|^2\right] \qquad (29) \\
&\leq \mathbb{E}\left[\frac{8\lambda^2}{|\mathcal{P}_k^{(t)}|} \sum_{\mathbf{x} \in \mathcal{P}_k^{(t)}} \|\nabla s(\mathbf{w}_k^{(t)}, \mathbf{x})^T (s(\mathbf{w}_k^{(t)}, \mathbf{x}) - \bar{\mathbf{s}}_k^{(t)}(\mathbf{x}))\|^2\right] \\
&+ 2G^2 \qquad (30) \\
&\leq \frac{8\lambda^2}{|\mathcal{P}|} \sum_{\mathbf{x} \in \mathcal{P}} \mathbb{E}\left[\|\nabla s(\mathbf{w}_k^{(t)}, \mathbf{x})\|^2 \|s(\mathbf{w}_k^{(t)}, \mathbf{x}) - \bar{\mathbf{s}}_k^{(t)}(\mathbf{x})\|^2\right] + 2G^2 \qquad (31) \\
&\leq 2\,G^2 + 16\lambda^2 L_s^2 \qquad (32)
\end{aligned}
$$

where (29) is due to the Cauchy-Schwarz inequality and AM-GM inequality, (30) is due to Assumption IV.4 and Jensen's inequality, (31) is due to the submultiplicativity of the norm, and (32) is due to Assumption IV.5 and that the maximum ℓ_2-norm distance between two probability vectors is $\sqrt{2}$.

Moreover,

$$
\begin{aligned}
&\|\nabla_{\mathbf{w}_k^{(t)}} \Phi_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})\| \qquad (33) \\
&= \|\frac{2\lambda}{|\mathcal{P}|} \sum_{\mathbf{x} \in \mathcal{P}} \nabla s(\mathbf{w}_k^{(t)}, \mathbf{x})^T (s(\mathbf{w}_k^{(t)}, \mathbf{x}) - \bar{\mathbf{s}}_k^{(t)}(\mathbf{x})) \qquad (34) \\
&+ \nabla F_k(\mathbf{w}_k^{(t)})\| \\
&\leq \frac{2\lambda}{|\mathcal{P}|} \|\sum_{\mathbf{x} \in \mathcal{P}} \nabla s(\mathbf{w}_k^{(t)}, \mathbf{x})^T (s(\mathbf{w}_k^{(t)}, \mathbf{x}) - \bar{\mathbf{s}}_k^{(t)}(\mathbf{x}))\| \qquad (35) \\
&+ \|\nabla F_k(\mathbf{w}_k^{(t)})\| \\
&\leq L_f + 2\lambda \sum_{\mathbf{x} \in \mathcal{P}} \|\nabla s(\mathbf{w}_k^{(t)}, \mathbf{x})^T (s(\mathbf{w}_k^{(t)}, \mathbf{x}) - \bar{\mathbf{s}}_k^{(t)}(\mathbf{x}))\| \qquad (36) \\
&\leq L_f + 2\lambda \sum_{\mathbf{x} \in \mathcal{P}} \|\nabla s(\mathbf{w}_k^{(t)}, \mathbf{x})\| \|(s(\mathbf{w}_k^{(t)}, \mathbf{x}) - \bar{\mathbf{s}}_k^{(t)}(\mathbf{x}))\| \qquad (37) \\
&\leq L_f + 2\sqrt{2}\lambda L_s |\mathcal{P}| = M_1 \qquad (38)
\end{aligned}
$$

and therefore $\|\nabla_{\mathbf{w}_k^{(t)}} \Phi_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})\|$ is bounded by $M_1 \geq 0$. With similar steps we can show that $\|\mathbf{g}_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})\| \leq M_2$ for a certain constant $M_2 \geq 0$. □

B. Main Proof for Theorem IV.6

Using Lemma A.1 and Lemma A.2 we have that

$$
\begin{aligned}
&\mathbb{E}[\Phi_k(\mathbf{w}_k^{(t+1)}; \bar{\mathbf{s}}_k^{(t)})|\mathcal{H}_t] \qquad (39) \\
&= \mathbb{E}[\Phi_k(\mathbf{w}_k^{(t)} - \eta_t \mathbf{g}_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})); \bar{\mathbf{s}}_k^{(t)})|\mathcal{H}_t] \qquad (40) \\
&\leq \Phi_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)}) + \frac{\eta_t^2 L_p}{2} \mathbb{E}[\|\mathbf{g}_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})\|^2|\mathcal{H}_t] \qquad (41) \\
&- \eta_t \nabla_{\mathbf{w}_k^{(t)}} \Phi_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})^T \mathbb{E}[\mathbf{g}_k(\mathbf{w}_k^{(t)}; \bar{\mathbf{s}}_k^{(t)})|\mathcal{H}_t]
\end{aligned}
$$

$$\leq \Phi_k(\mathbf{w}_k^{(t)};\bar{\mathbf{s}}_k^{(t)}) - \eta_t\|\nabla_{\mathbf{w}_k^{(t)}}\Phi_k(\mathbf{w}_k^{(t)};\bar{\mathbf{s}}_k^{(t)})\|^2 + \frac{\eta_t^2 L_p}{2}(2\,G^2 + 16\lambda^2 L_s^2) \tag{42}$$

Assuming $\sum_{t=0}^{\infty}\eta_t^2 < \infty$ and $\sum_{t=0}^{\infty}\eta_t = \infty$, and applying Robbins-Siegmund Theorem (Theorem B.1. in [49]) on (42), we have that with probability 1,

$$\sum_{t=1}^{\infty}\eta_t\|\nabla_{\mathbf{w}_k^{(t)}}\Phi_k(\mathbf{w}_k^{(t)};\bar{\mathbf{s}}_k^{(t)})\|^2 < \infty \tag{43}$$

Now we can show

$$\|\nabla\Phi_k(\mathbf{w}_k^{(t+1)};\bar{\mathbf{s}}_k^{(t)})\|^2 - \|\nabla\Phi_k(\mathbf{w}_k^{(t)};\bar{\mathbf{s}}_k^{(t)})\|^2 \tag{44}$$

$$= (\|\nabla\Phi_k(\mathbf{w}_k^{(t+1)};\bar{\mathbf{s}}_k^{(t)})\| + \|\nabla\Phi_k(\mathbf{w}_k^{(t)};\bar{\mathbf{s}}_k^{(t)})\|) \tag{45}$$

$$\times (\|\nabla\Phi_k(\mathbf{w}_k^{(t+1)};\bar{\mathbf{s}}_k^{(t)})\| - \|\nabla\Phi_k(\mathbf{w}_k^{(t)};\bar{\mathbf{s}}_k^{(t)})\|)$$

$$\leq 2M_1(\|\nabla\Phi_k(\mathbf{w}_k^{(t+1)};\bar{\mathbf{s}}_k^{(t)})\| - \|\nabla\Phi_k(\mathbf{w}_k^{(t)};\bar{\mathbf{s}}_k^{(t)})\|) \tag{46}$$

$$\leq 2M_1\|\nabla\Phi_k(\mathbf{w}_k^{(t+1)};\bar{\mathbf{s}}_k^{(t)}) - \nabla\Phi_k(\mathbf{w}_k^{(t)};\bar{\mathbf{s}}_k^{(t)})\| \tag{47}$$

$$\leq 2M_1\|\eta_t\mathbf{g}_k(\mathbf{w}_k^{(t)};\bar{\mathbf{s}}_k^{(t)})\| \leq 2M_1M_2\eta_t \tag{48}$$

Finally, using Proposition 2 in [50] we have that for $t\to\infty$, $\|\nabla_{\mathbf{w}_k^{(t)}}\Phi_k(\mathbf{w}_k^{(t)};\bar{\mathbf{s}}_k^{(t)})\| \to 0$ with probability 1.

Appendix B

A. Proof for Theorem IV.7

We have that (13) is equal to:

$$\widetilde{\mathbf{w}}_k = \frac{1}{1+\lambda\nu/\beta}\widehat{\mathbf{w}}_k + \frac{1}{1+\beta/\lambda\nu}\sum_{i=1}^{K}\alpha_{k,i}\widehat{\mathbf{w}}_i \tag{49}$$

and the Bayes optimal $\mathbf{w}_k$ in (12) becomes

$$\mathbf{w}_k = \frac{1}{1+\sigma^2/\beta\upsilon_k^2}\widehat{\mathbf{w}}_k + \frac{A_k\sigma^2}{\sigma^2+A_k\beta\upsilon_k^2}\sum_{i=1}^{K}\frac{1}{\sigma^2+\beta\upsilon_i^2}\widehat{\mathbf{w}}_i + \varsigma_k \tag{50}$$

where $A_k = (\sum_{i\in[K],i\neq k}\frac{1}{\sigma^2+\beta\upsilon_i^2})^{-1}$ and $\varsigma_k \sim \mathcal{N}(0,(\frac{\beta}{A_k+\beta\upsilon_k^2} + \frac{\beta}{\sigma^2})^{-1})$. If we aim to find the λ_k and $\alpha_{k,i}, i\in[K]$ that minimizes $\mathbb{E}[F_k(\widetilde{\mathbf{w}}_k)]$ given $\widehat{\mathbf{w}}_k$ and $\overline{\theta}_{\setminus k}$, in other words,

$$\lambda_k^*, \alpha_{k,i}^*, i\in[K]$$

$$= \arg\min_{\lambda_k,\alpha_{k,i},i\in[K]}\mathbb{E}[F_k(\widetilde{\mathbf{w}}_k)|\widehat{\mathbf{w}}_k,\overline{\theta}_{\setminus k}] \tag{51}$$

$$= \arg\min_{\lambda_k,\alpha_{k,i},i\in[K]}\mathbb{E}[\|\mathbf{X}_k\widetilde{\mathbf{w}}_k - (\mathbf{X}_k\mathbf{w}_k+\mathbf{z})\|_2^2|\widehat{\mathbf{w}}_k,\overline{\theta}_{\setminus k}] \tag{52}$$

$$= \arg\min_{\lambda_k,\alpha_{k,i},i\in[K]}\mathbb{E}[\|\mathbf{X}_k(\widetilde{\mathbf{w}}_k - \mathbf{w}_k)\|_2^2|\widehat{\mathbf{w}}_k,\overline{\theta}_{\setminus k}] \tag{53}$$

$$= \arg\min_{\lambda_k,\alpha_{k,i},i\in[K]}\mathbb{E}[\|\widetilde{\mathbf{w}}_k - \mathbf{w}_k\|_2^2|\widehat{\mathbf{w}}_k,\overline{\theta}_{\setminus k}] \tag{54}$$

then taking (49) and (50) into (54) we have that

$$\lambda_k^* = \sigma^2/\upsilon_k^2\nu \tag{55}$$

$$\alpha_{k,i}^* = \frac{B_k}{\sigma^2+\beta\upsilon_i^2} \tag{56}$$

where $B_k = \frac{A_k(\sigma^2+\beta\upsilon_k^2)}{\sigma^2+A_k\beta\upsilon_k^2}$.

Appendix C
Generalization Bound for Ensemble Models in Personalization

As defined in Section III-A, we have the true data distribution of client k defined as $\mathcal{D}_k$, and the empirical data distribution associated with the client's training dataset defined as $\widehat{\mathcal{D}}_k$. For a multi-class classification problem with a finite set of classes, we have that the data's domain is defined by the input space $\mathbf{x}\in\mathcal{X}$ and the output space $y\in\mathcal{Y}$. For the generalization bound analysis we consider hypotheses that maps $h:\mathcal{X}\to\mathcal{Y}$, and $\mathcal{H}$ is defined as the hypotheses space such that $h\in\mathcal{H}$. The loss function $l(h(\mathbf{x}),y)$ measures the classification performance of h for a single data point $(\mathbf{x},y)$ and we define the expected loss over all data points that follow distribution $\mathcal{D}$ as $\mathcal{L}_{\mathcal{D}}(h) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[l(h(\mathbf{x}),y)]$. We assume that $\mathcal{L}(\cdot)$ is convex, and is in the range [0,1]. We define the minimizer of the expected loss over the data that follows the distribution $\mathcal{D}_k$ and $\widehat{\mathcal{D}}_k$ as each $h_k = \arg\min_h\mathcal{L}_{\mathcal{D}_k}(h)$ and $\widehat{h}_k = \arg\min\mathcal{L}_{\widehat{\mathcal{D}}_k}(h)$. Note that for sufficiently large training dataset, we will have $h_k \simeq \widehat{h}_k$.

Our goal is to show the generalization bound for client k such that $\mathcal{L}_{\mathcal{D}_k}(\sum_{i=1}^{K}\alpha_{k,i}h_{\widehat{\mathcal{D}}_i})$, where $h_{\widehat{\mathcal{D}}_i}$ represents the hypothesis trained from client i's training dataset and $\alpha_{k,i}$ represents the weight for the hypothesis of client i for client k. For client $i\in[K]$, $h_{\widehat{\mathcal{D}}_i}$ will be the optimal hypothesis with respect to the training dataset for each client participating in FL, and the generalization bound for $\mathcal{L}_{\mathcal{D}_k}(\sum_{i=1}^{K}\alpha_{k,i}h_{\widehat{\mathcal{D}}_i})$ will show how the weighted average of different hypothesis from the other clients with respect to $\alpha_{k,i}$, $i\in[K]$ helps the generalization of an individual client k with respect to its true data distribution. Before presenting the generalization bound, we present several useful lemmas.

Lemma C.1 (Domain adaptation [51]): With two true distributions $\mathcal{D}_A$ and $\mathcal{D}_B$, for $\forall\,\delta\in(0,1)$ and hypothesis $\forall h\in\mathcal{H}$, with probability at least $1-\delta$ over the choice of samples, there exists:

$$\mathcal{L}_{\mathcal{D}_A}(h) \leq \mathcal{L}_{\mathcal{D}_B}(h) + \frac{1}{2}d(\mathcal{D}_A,\mathcal{D}_B) + \nu \tag{57}$$

where $d(\mathcal{D}_A,\mathcal{D}_B)$ measures the distribution discrepancy between two distributions [51] and $\nu = \inf_h\mathcal{L}_{\mathcal{D}_A}(h) + \mathcal{L}_{\mathcal{D}_B}(h)$.

Lemma C.2 (Generalization with limited training samples): For $\forall\,k\in[K]$, with probability at least $1-\delta$ over the choice of samples, there exists:

$$\mathcal{L}_{\mathcal{D}_k}(h_{\widehat{\mathcal{D}}_k}) \leq \mathcal{L}_{\widehat{\mathcal{D}}_k}(h_{\widehat{\mathcal{D}}_k}) + \sqrt{\frac{\log 2/\delta}{2m_k}} \tag{58}$$

where m_k is the number of training samples of client k. This lemma shows that for small number of training samples, i.e., small m_k, the generalization error increases due to the discrepancy between $\mathcal{D}_k$ and $\widehat{\mathcal{D}}_k$.

Proof: We seek to bound the gap between $\mathcal{L}_{\mathcal{D}_k}(h_{\widehat{\mathcal{D}}_k})$ and $\mathcal{L}_{\widehat{\mathcal{D}}_k}(h_{\widehat{\mathcal{D}}_k})$. Observe that $\mathcal{L}_{\mathcal{D}_k}(h_{\widehat{\mathcal{D}}_k}) = \mathbb{E}[\mathcal{L}_{\widehat{\mathcal{D}}_k}(h_{\widehat{\mathcal{D}}_k})]$, where the expectation is taken over the randomness in the sample draw that generates $\widehat{\mathcal{D}}_k$, and that $\mathcal{L}_{\widehat{\mathcal{D}}_k}(h_{\widehat{\mathcal{D}}_k})$ is an empirical mean

over losses $l(h(x), y)$ that lie within $[0, 1]$. Since we are simply bounding the difference between a sample average of bounded random variables and its expected value, we can directly apply Hoeffding's inequality to obtain

$$\mathbb{P}\left[\mathcal{L}_{\hat{\mathcal{D}}_k}(h_{\hat{\mathcal{D}}_k}) - \mathcal{L}_{\mathcal{D}_k}(h_{\hat{\mathcal{D}}_k}) \geq \epsilon\right] \leq 2e^{-2m\epsilon^2}. \tag{59}$$

Setting the right hand side to δ and rearranging gives the desired bound with probability at least $1-\delta$ over the choice of samples:

$$\mathcal{L}_{\mathcal{D}_k}(h_{\hat{\mathcal{D}}_k}) \leq \mathcal{L}_{\hat{\mathcal{D}}_k}(h_{\hat{\mathcal{D}}_k}) + \sqrt{\frac{\log 2/\delta}{2m_k}}.$$

□

We now present the generalization bound for $\mathcal{L}_{\mathcal{D}_k}(\sum_{i=1}^K \alpha_{k,i} h_{\hat{\mathcal{D}}_i})$ as follows:

$$\mathcal{L}_{\mathcal{D}_k}\left(\sum_{i=1}^K \alpha_{k,i} h_{\hat{\mathcal{D}}_i}\right) \underset{(c)}{\leq} \sum_{i=1}^K \alpha_{k,i} \mathcal{L}_{\mathcal{D}_k}(h_{\hat{\mathcal{D}}_i})$$

$$\underset{(d)}{\leq} \sum_{i=1}^K \alpha_{k,i}[\mathcal{L}_{\mathcal{D}_i}(h_{\hat{\mathcal{D}}_i}) + \frac{1}{2} d(\mathcal{D}_i, \mathcal{D}_k) + \nu_i] \tag{60}$$

where $\nu_i = \inf_h \mathcal{L}_{\mathcal{D}_i}(h) + \mathcal{L}_{\mathcal{D}_k}(h)$, (c) is due to the convexity of $\mathcal{L}$, and (d) is due to Lemma C.1. We can further bound (60) using Lemma C.2 as

$$\mathcal{L}_{\mathcal{D}_k}\left(\sum_{i=1}^K \alpha_{k,i} h_{\hat{\mathcal{D}}_i}\right) \leq \sum_{i=1}^K \alpha_{k,i} \mathcal{L}_{\hat{\mathcal{D}}_i}(h_{\hat{\mathcal{D}}_i}) \tag{61}$$

$$+ \sum_{i=1}^K \alpha_{k,i} \sqrt{\frac{\log 2/\delta}{2m_k}} + \frac{1}{2}\sum_{i=1}^K \alpha_{k,i} d(\mathcal{D}_i, \mathcal{D}_k) + \sum_{i=1}^K \alpha_{k,i}\nu_i$$

$$= \sum_{i=1}^K \alpha_{k,i} \mathcal{L}_{\hat{\mathcal{D}}_i}(h_{\hat{\mathcal{D}}_i}) + \sqrt{\log \delta^{-1}} \sum_{i=1}^K \frac{\alpha_{k,i}}{\sqrt{m_k}}$$

$$+ \frac{1}{2}\sum_{i=1}^K \alpha_{k,i} d(\mathcal{D}_i, \mathcal{D}_k) + \sum_{i=1}^K \alpha_{k,i}\nu_i \tag{62}$$

From (62), with $\mathcal{L}_{\hat{\mathcal{D}}_i}(h_{\hat{\mathcal{D}}_i})$ in general being small for $\forall i \in [K]$ as it is the minimum loss, and m_i being similar to other $m_{i'}$, $i' \in [K]$, the only way to minimize the generalization error of $\mathcal{L}_{\mathcal{D}_k}(\sum_{i=1}^K \alpha_{k,i} h_{\hat{\mathcal{D}}_i})$ is to set the weights $\alpha_{k,i}$ so that the third term $\frac{1}{2}\sum_{i=1}^K \alpha_{k,i} d(\mathcal{D}_i, \mathcal{D}_k)$ is minimized. Note that it is difficult to know the value of ν_i, making it impractical to minimize the fourth term in practice. This generalization results strengthens our motivation to use to find the weights $\alpha_{k,i}$, $i \in [K]$ that minimizes $\frac{1}{|\mathcal{P}|}\sum_{\mathbf{x}\in\mathcal{P}} \| \sum_{i=1}^K \alpha_{k,i} s_i(\mathbf{w}_i, \mathbf{x}) - s(\mathbf{w}_k, \mathbf{x})\|_2^2$ in regards to the objective function we have in (1).

Appendix Details of Experimental Setup and Additional Results

Codes for the results in the paper are presented in the supplementary material.

C. Description for Toy Example - Fig. 2

For Fig. 2, we design a linear regression problem where the true local model for each client is generated as $\mathbf{w}_i = \theta + \zeta_i, i \in [3]$ where $\theta \in \mathbb{R}^{2\times 1}$ is a non-informative prior which elements are uniformly distributed $\mathcal{U}(-10, 10)$ and the elements of ζ_i follows the normal distribution $\mathcal{N}(0, \sigma_i)$, $\sigma_1 = 2, \sigma_2 = 5, \sigma_3 = 200$. The discrepancy across the variance denotes the data-heterogeneity across the clients. The range for $\mathbf{x}$ is $[-10, 10]$ for all elements. We assume all clients have identical dataset size. For implementing COMET for the toy example, we set the public data range as identical to the input data range, and set $\lambda = 50$. For KT w/o clustering, the codistillation term uses a simple average of all the logits from the clients for regularizing while for KT with clustering the weights are set so that clients with similar true local models have higher weights for each other. This setting is also consistent with the generalization analysis presented in Section IV-B.

TABLE V
ADDITIONAL RESULTS FOR VARYING c FOR HYPCLUSTER ALGORITHM FOR EXPERIMENT IN TABLE I

	$C = 0.10$		$C = 0.15$	
	Test Acc.	Com.-Cost	Test Acc.	Com.-Cost
$c = 2$	34.11 ($\pm$2.63)	2340×10^7	28.70 ($\pm$3.03)	3320×10^7
$c = 3$	39.17 ($\pm$2.64)	2540×10^7	41.99 ($\pm$2.48)	3510×10^7
$c = 5$	52.51 ($\pm$1.37)	2930×10^7	51.92 ($\pm$1.64)	3900×10^7
$c = 6$	65.77 ($\pm$2.76)	3120×10^7	63.28 ($\pm$1.16)	4100×10^7

D. Description for Image Classification Experiments

a) Data Partitioning: The partitioning of each individual client's data to training/validation/test dataset is done as follows: after partitioning the entire dataset by the Dirichlet distribution $\text{Dir}_K(\alpha)$ with $\alpha = 0.01$ across clients, we partition each client dataset by a $\{0.1, 0.3, 0.4\}/0.1/0.5$ ratio where the ratio for the training dataset is chosen by random from $\{0.1, 0.3, 0.4\}$ for each client. Such partitioning simulates a more realistic FL setting where individual clients may not have sufficient labeled data samples for training that represents the test dataset's distribution.

b) Local Training and Hyperparameters: For the local-training hyperparameters, we do a grid search over the learning rate: $\eta \in \{0.1, 0.05, 0.01, 0.005, 0.001\}$, batchsize: $b \in \{32, 64, 128\}$, and local iterations: $\tau \in \{10, 30, 50\}$ to find the hyper-parameters with the highest test accuracy for each benchmark. For all benchmarks we use the best hyper-parameter for each benchmark after doing a grid search over feasible parameters referring to their source codes that are open-sourced. For the knowledge distillation server-side hyperparameters, we do a grid search over the public batch size: $b' \in \{32, 64, 128\}$, regularization weight $\lambda \in \{0.05, 0.1, 0.5, 1, 2, 4\}$ to find the best working hyperparameters. The best hyperparameters for COMET we use is $\eta = 0.001, b = 64, \tau = 50, b' = 128, \lambda = 2$.

c) Model Setup: For the model configuration, for the CNN we have a self-defined convolutional neural network with 2 convolutional layers with max pooling and 4 hidden fully connected linear layers of units [120,100,84,50]. The input is the flattened convolution output and the output is consisted of 10 or 100 units each of one of the 0-9 or 0-99 labels. For the VGG, we use the

open-sourced VGG net from Pytorch with torchvision ver.0.4.1 presented in Pytorch without pretrained as False and batchnorm as True.

d) Platform: All experiments are conducted with clusters equipped with one NVIDIA TitanX GPU. The number of clusters we use vary by C, the fraction of clients we select. The machines communicate amongst each other through Ethernet to transfer the model parameters and information necessary for client selection. Each machine is regarded as one client in the federated learning setting. The algorithms are implemented by PyTorch.

REFERENCES

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2017, pp. 1273–1282. [Online]. Available: https://arxiv.org/abs/1602.05629

[2] B. Woodworth et al., "Is local SGD better than minibatch SGD," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 10334–10343.

[3] A. Khaled, K. Mishchenko, and P. Richtárik, "Tighter theory for local SGD on identical and heterogeneous data," in *Proc. 23rd Int. Conf. Artif. Intell. Statist.*, 2020, pp. 4519–4529.

[4] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 7611–7623.

[5] S. Reddi et al., "Adaptive federated optimization," in *Proc. Int. Conf. Learn. Representations*, 2021.

[6] S. U. Stich and S. P. Karimireddy, "The error-feedback framework: Better rates for SGD with delayed gradients and compressed communication," *J. Mach. Learn. Res.*, 2020.

[7] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, *arXiv:2006.04088*.

[8] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning," *IEEE Trans. Neural Netw. Learn. Syst.*, 2022, pp. 1–17.

[9] P. P. Liang et al., "Think locally, act globally: Federated learning with local and global representations," in *Proc. Nternational Workshop Feder-Ated Learn. User Privacy Data Confidentiality Inconjunction*, 2019. [Online]. Available: http://arxiv.org/abs/2001.01523

[10] Y. J. Cho, J. Wang, and G. Joshi, "Client selection in federated learning: Convergence analysis and power-of-choice selection strategies," in *Proc. 25rd Int. Conf. Artif. Intell. Statist.*, 2022.

[11] Y. J. Cho, S. Gupta, G. Joshi, and O. Yağan, "Bandit-based communication-efficient client selection strategies for federated learning," in *Proc. IEEE 54th Asilomar Conf. Signals, Systems, Comput.*, 2020, pp. 1066–1069.

[12] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 3557–3568.

[13] M. Zhang, K. Sapra, S. Fidler, S. Yeung, and J. M. Alvarez, "Personalized federated learning with first order model optimization," in *Proc. Int. Conf. Learn. Representations*, 2021.

[14] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 6357–6368.

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

[16] N. Shazeer et al., "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," in *Proc. Int. Conf. Learn. Respresentations*, 2017.

[17] C. He, M. Annavaram, and S. Avestimehr, "Group knowledge transfer: Federated learning of large CNNs at the edge," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 14068–14080.

[18] Y. J. Cho, A. Manoel, G. Joshi, R. Sim, and D. Dimitriadis, "Heterogeneous ensemble knowledge transfer for training large models in federated learning," in *Proc. 13th Int. Joint Conf. Artif. Intell.*, 2022.

[19] D. Li and J. Wang, "Fedmd: Heterogenous federated learning via model distillation," in *Proc. Int. Workshop Feder-ated Learn. User Privacy Data Confidentiality Conjunction*, 2019.

[20] J. Zhang, S. Guo, X. Ma, H. Wang, W. Xu, and F. Wu, "Parameterized knowledge transfer for personalized federated learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 10092–10104.

[21] S. Itahara, T. Nishio, Y. Koda, M. Morikura, and K. Yamamoto, "Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data," *IEEE Trans. Mobile Comput.*, Aug. 2020. [Online]. Available: https://arxiv.org/abs/2008.06180

[22] S. Sodhani, O. Delalleau, M. Assran, K. Sinha, N. Ballas, and M. Rabbat, "A closer look at codistillation for distributed training," *CoRR* 2020, *arXiv:2010.02838*.

[23] I. Bistritz, A. J. Mann, and N. Bambos, "Distributed distillation for on-device learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 22593–22604.

[24] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 2351–2363.

[25] R. Anil, G. Pereyra, A. Passos, R. Ormandi, G. E. Dahl, and G. E. Hinton, "Large scale distributed neural network training through online distillation," in *Proc. Int. Conf. Learn. Representations*, 2018, *arXiv:1804.03235*.

[26] R. Pathak and M. J. Wainwright, "FedSplit: An algorithmic framework for fast federated optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 7057–7066.

[27] C. T. Dinh, N. H. Tran, and T. D. Nguyen, "Personalized federated learning with moreau envelopes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 21394–21405.

[28] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[29] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, " How transferable are features in deep neural networks," in *Proc. Neural Inf. Process. Syst.*, 2014.

[30] Y. Guo, H. Shi, A. Kumar, K. Grauman, T. Rosing, and R. Feris, "Spottune: Transfer learning through adaptive fine-tuning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4800–4809.

[31] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Proc. NIPS Workshop Deep Learn.*, 2014.

[32] J. Ma, R. Yonetani, and Z. Iqbal, "Adaptive distillation for decentralized learning from heterogeneous clients," in *Proc. 25th Int. Conf. Pattern Recognit.*, 2020, pp. 7486–7492.

[33] S. Itahara, T. Nishio, Y. Koda, M. Morikura, and K. Yamamoto, "Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data," *IEEE Trans. Mobile Comput.*, vol. 22, no. 1, pp. 191–205, Aug. 2020. [Online]. Available: https://arxiv.org/abs/2008.06180

[34] L. Sun and L. Lyu, "Federated model distillation with noise-free differential privacy," in *Proc. 13th Int. Joint Conf. Artif. Intell.*, 2021.

[35] Q. Li, B. He, and D. Song, "Practical one-shot federated learning for cross-silo setting," in *Proc. 13th Int. Joint Conf. Artif. Intell.*, 2021.

[36] Y. Zhou, G. Pu, X. Ma, X. Li, and D. Wu, "Distilled one-shot federated learning," Jun. 2021, *arXiv:2009.07999*.

[37] S. Lee, K. Yoo, and N. Kwak, "Edge bias in federated learning and its solution by buffered knowledge distillation," Feb. 2021, *arXiv:2010.10338*.

[38] L. Lyu et al., "Privacy and robustness in federated learning: Attacks and defenses," *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–21, 2022.

[39] X. Lin et al., "Federated learning with positive and unlabeled data," in *Proc. 39th Int. Conf. Mach. Learn.*, in Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., vol. 162, Jul. 2022, pp. 13 344–13 355.

[40] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *J. Mach. Learn. Res.*, vol. 20, no. 1, pp. 1997–2017, 2019.

[41] Z. Allen-Zhu and Y. Li, "Towards understanding ensemble, knowledge distillation and self-distillation in deep learning," Jul. 2021, *arXiv:2012.09816.pdf*.

[42] H. Chang, V. Shejwalkar, R. Shokri, and A. Houmansad, "Cronus: Robust and heterogeneous collaborative learning with black-box knowledge transfer," in *Proc. NeurIPS Workshop New Front. FL*, 2021.

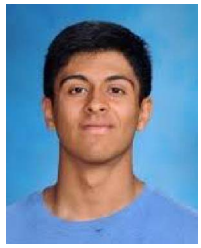
[43] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *J. Roy. Stat. Society. Ser. C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[44] A. Krizhevsky, V. Nair, and G. Hinton, "Learning multiple layers of features from tiny images," CIFAR-10 (Canadian Institute for Advanced Research), 2009. [Online]. Available: http://www.cs.toronto.edu/kriz/cifar.html

[45] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[46] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," in *Proc. Int. Workshop Federated Learn. User Privacy Data Confidentiality Conjunction*, 2019.
[47] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015.
[48] X. An et al., "Partial FC: Training 10 million identities on a single machine," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 1445–1449.
[49] T. Poggio, S. Voinea, and L. Rosasco, "Online learning, stability, and stochastic gradient descent," 2019.
[50] Y. Alber, A. Iusem, and M. Solodovz, "On the projected subgradient method for nonsmooth convex optimization in a Hilbert space," *Math. Program.*, vol. 81, no. 1, pp. 23–25, 1998.
[51] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Mach. Learn.*, vol. 79, no. 1/2, pp. 151–175, 2009.

Yae Jee Cho received the B.S. and M.S. degrees in electrical and computer engineering from Yonsei University, Seoul, South Korea, in 2016 and 2018, respectively. She is currently working toward the Ph.D. degree with Carnegie Mellon University, Pittsburgh, PA, USA, advised by Professor Gauri Joshi. Between 2020 and 2021, she was a Research Intern with Microsoft Research of language intelligence and federated learning. Her research interests include data-aware federated learning, distributed optimization, and large-scale machine learning. She is the Qualcomm Innovation Fellowship (2022) finalist and was recipient of the Korean Government Doctoral Scholarship from Republic of Korea (2019).

Jianyu Wang received the B.E. degree in electronic engineering from Tsinghua University, Beijing, China, in 2017. He is currently working toward the Ph.D. degree with Carnegie Mellon University, Pittsburgh, PA, USA, advised by Professor Gauri Joshi. Between 2020 and 2021, he was a Research Intern with Google Research and in 2019, with Facebook AI Research. His research interests include federated learning, distributed optimization, and systems for large-scale machine learning. His research has been supported by Qualcomm Ph.D. fellowship (2019).



Tarun Chirvolu is currently working toward the undergraduation degree with the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA. Since 2020, he has been assisting Professor Gauri Joshi in research related to personalized federated learning and queueing theory in federated learning. His research interests include federated learning, applied probability, and queueing theory.

Gauri Joshi (Member, IEEE) received the B.Tech. and M.Tech. degrees in electrical engineering from the Indian Institute of Technology Bombay, Mumbai, India, in 2010, and the Ph.D. degree in EECS from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2016. She is currently an Assistant Professor with ECE Department, Carnegie Mellon University, Pittsburgh, PA, USA. From 2016 to 2017, she was a Research Staff Member with IBM T. J. Watson Research Center. Her research interests include federated learning, distributed optimization, and coding theory. Her awards and honors include the NSF CAREER Award in 2021, ACM Sigmetrics Best Paper Award in 2020, and Institute Gold Medal of IIT Bombay in 2010.