UNIVERSITY OF CALIFORNIA SAN DIEGO

On Distributed Learning Techniques for Machine Learning

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Electrical Engineering (Signal & Image Processing)

by

Saeed Vahidian

Committee in charge:

Professor Bill Lin, Chair
Professor Philip E. Gill
Professor Yuanyuan Shi
Professor Hao Su
Professor Xiaolong Wang

2023

The Dissertation of Saeed Vahidian is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2023

DEDICATION

To My Beloved Heavenly Parents:

Ali ibn Abi Talib (Imam Amir al-Mu'minin) and *Fatemeh al-Zahra*

&

To My Beloved Earthly Parents:

Malek-Hossein and *Ashraf*.

EPIGRAPH

The most valuable of all treasures is knowledge, and the worst of all losses is ignorance.

Imam Hossein (son of Ali ibn Abi Talib)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACKNOWLEDGEMENTS

I would like to take this opportunity to extend my heartfelt appreciation and gratitude to my beloved family: my parents, Malek-Hossein and Ashraf; my brother and brother-in-laws, Vahid, Mohsen, and Mohammad-Ali; and my sisters and sister-in-law, Farideh, Fatemeh, and Maryam, whose unconditional love, support, and encouragement have been my constant source of strength and motivation throughout my Ph.D. journey. Their unwavering faith in my abilities, their willingness to sacrifice their own needs and desires to support my dreams, and their steadfast presence during the many challenges and setbacks that I faced along the way, have been instrumental in helping me reach this milestone in my academic career. I am deeply indebted to them for their unwavering support, and I dedicate this thesis to them with love and gratitude for all that they have done for me.

I would like to express my profound gratitude to my dearest friends: Mr. Shahram Karimi, Mrs. Akram Golparvar, Mr. Ali Hooshmand, Dr. Mohammad Samavat, and Dr. Seyed Javad Hashemi, whose guidance, care, and encouragement have been the bedrock of my journey toward earning this Ph.D. degree. Their selflessness, kindness, support, and unwavering belief in my abilities have been a source of motivation and inspiration, and I am truly blessed to have such wonderful friends who have been with me every step of the way.

My Ph.D. at the University of California San Diego (UCSD) was enriched by the mentorship of scholars and professors. I am thankful to my Ph.D. committee members: Prof. Philip E. Gill, Dr. Hao Su, Dr. Yuanyuan Shi, and Dr. Xiaolong Wang.

I am humbled to express my deep gratitude towards my Ph.D. advisor Prof. Bill Lin whose unwavering commitment to excellence, boundless enthusiasm, and tireless dedication to fostering my intellectual growth and development has been an inspiration to me throughout my Ph.D. journey. His comments, insight, and expertise have enriched my academic experience, allowing me to grow as a researcher. I am truly honored to have had the privilege of working under his supervision.

Chapter 2, in full, has been submitted for the publication of the material as it may appear in CVPR, 2023. Saeed Vahidian, Sreevatsank Kadaveru, Woonjoon Baek, Weijia Wang, Vyacheslav Kungurtsev, Chen Chen, Mubarak Shah, and Bill Lin. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in full, is a reprint of the material as it appears in AAAI Conference on Artificial Intelligence, 2023. Saeed Vahidian, Mahdi Morafah, Weijia Wang, Vyacheslav Kungurtsev, Chen Chen, Mubarak Shah, and Bill Lin. The dissertation author was the primary investigator and author of this paper.

Chapter 4, in full, is a reprint of the material as it appears in IEEE 41st international conference on distributed computing systems workshops, 2021. Saeed Vahidian, Mahdi Morafah, and Bill Lin. The dissertation author was the primary investigator and author of this paper.

Chapter 5, in part, is a reprint of the material as it appears in NeurIPS Workshop, 2023. Saeed Vahidian, Mahdi Morafah, Chen Chen, Mubarak Shah, and Bill Lin. The dissertation author was the primary investigator and author of this paper. This Chapter has also been submitted with the same authors for publication of the material as it may appear in the journal of IEEE Transactions on Artificial Intelligence.

Chapter 6, in full, is a reprint of the material as it appears in Conference on Uncertainty in Artificial Intelligence (UAI), 2020. Saeed Vahidian, Baharan Mirzasoleiman, and Alexander Cloninger. The dissertation author was the primary investigator and author of this paper.

VITA

2007-2012 B.Sc. in Electrical Engineering, Ferdowsi University of Mashhad, Iran

2012-2014 M.Sc. in Electrical Engineering, Khaje Nasir Toosi University of Technology, Iran

2018-2023 Doctor of Philosophy, University of California San Diego

PUBLICATIONS

CVPR'2022 When Do Curricula Work in Federated Learning?
Saeed Vahidian, S. Kadaveru, W. Baek, W. Wang, V. Kungurtsev, C. Chen, M. Shah, B. Lin , *submitted to The IEEE/CVF Computer Vision and Pattern Recognition Conference (CVPR), 2022.*

CVPR'2022 FLAR: A Communication Efficient Federated Learning Framework for Video Action Recognition

U. Khalid, **Saeed Vahidian**, T. Yang, M. Mendieta, V. Kungurtsev, G. Sun, Y. Zhu, Bill Lin, C. Chen *submitted to The IEEE/CVF Computer Vision and Pattern Recognition Conference (CVPR), 2022*

AAAI'2022 Efficient Distribution Similarity Identification in Federated Learning via Principal Angles Between Client Data Subspaces

Saeed Vahidian, M. Morafah, W. Wang, V. Kungurtsev, B. Lin , *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI, Washington DC., 2022.*

NeurIPS'2022 Rethinking Data Heterogeneity in Federated Learning: Introducing a New Notion and Standard Benchmarks

Saeed Vahidian, M. Morafah, C. Chen, M. Shah, B. Lin, *NeurIPS workshop 2022.*

NeurIPS'2022 FedIS: Dynamic Clustering via Inference Similarity for Non-IID Federated Learning

Saeed Vahidian, M. Moraffah, W. Wang, and B. Lin, *NeurIPS workshop 2022.*

Trans on AI'2022 FedIS (extended version): Dynamic Clustering via Inference Similarity for Non-IID Federated Learning

Saeed Vahidian, M. Moraffah, W. Wang, and B. Lin, *submitted to IEEE Transaction on AI, 2022.*

IEEE Access'2022 Optimality of Spectrum Pursuit for Column Subset Selection Problem: Theoretical Guarantees and Applications in Deep Learning

Saeed Vahidian, M. Joneidi, S. Khodadadeh, A. Esmaeili, B. Lin, *IEEE Access, 2022.*

ICDCS'2021 Personalized Federated Learning by Structured and Unstructured Pruning under Data Heterogeneity

Saeed Vahidian, M. Moraffah, B. Lin, *IEEE International Conference on Distributed Computing Systems, 2021.*

CVPR'2020 Select to Better Learn: Fast and Accurate Deep Learning using Data Selection from Nonlinear Manifolds

M. Joneidi, **Saeed Vahidian,** A. Esmaeili, W. Wong, N. Rahnavard, B. Lin, and M. Shah, *The IEEE/CVF Computer Vision and Pattern Recognition Conference (CVPR), 2020.*

ICLR'2020 Unsupervised Meta-Learning through Latent-Space Interpolation in Generative Models

S. Khodadadeh, S. Zehtabian, **Saeed Vahidian**, W. Wang, B. Lin, and L. Boloni, *International Conference on Learning Representations (ICLR), 2020.*

UAI'2020 Coresets for estimating means and mean square error with limited greedy samples

Saeed Vahidian, A. Cloninger, B. Mirzasoleiman, *Conference on Uncertainty in Artificial Intelligence (UAI), 2020.*

ABSTRACT OF THE DISSERTATION

On Distributed Learning Techniques for Machine Learning

by

Saeed Vahidian

Doctor of Philosophy in Electrical Engineering (Signal & Image Processing)

University of California San Diego, 2023

Professor Bill Lin, Chair

Modern machine and deep learning algorithms require a lot of data and computation and benefit from large and representative datasets (e.g., ImageNet and COCO), as much data and computational resources as possible should be gathered. However, in some of the applications, collecting abundant examples for certain classes in domains such as biology and medicine may be impossible in practice. For instance, in dermatology, there are some rare diseases with a small number of patients. It is, therefore, natural to ask can we train a model using data that is naturally dispersed among different parties in practice (e.g., edge devices and hospitals, etc.) without explicitly sharing their data? `Federated Learning (FL)` is a recently proposed distributed training framework in edge computing environment that enables distributed edge

devices to collaboratively train a global model under the orchestration of a central server without compromising the privacy of their data. While FL has great potential, it faces challenges in practical settings, including statistical data heterogeneity (Non-IID data), personalization, fairness, computation overhead, and communication cost. I designed techniques that could alleviate the mentioned challenges in FL. FL is explicitly designed for Non-IID edge devices. A global model can perform well on personalized predictions if the edge device's context and personal data are nicely featured and embodied in the dataset, which is not the case for most edge devices. Most techniques for personalization either fail to build a model with low generalization error or are not very effective, especially when local distributions are far from the average distribution. In addition, when the edge devices are the edge devices, computational efficiency, and communication cost will be a crucial bottleneck, as edge devices are typically constrained by computational limitations and upload bandwidth of 1 MB/s or less.

On the other side of the spectrum, due to redundancy, datasets' information content is much smaller than their actual volume, despite their steady growth. Existing techniques are not effective in identifying and extracting the non-redundant information content, considering the intrinsic structure of the data. Hence, machine learning models are trained on massive data volumes, which necessitates exceptionally large and expensive computational resources. A crucial challenge of machine learning today is to develop methods that can extract representative information volumes and accurately and robustly learn from the extracted representatives. My methods have immediate application to high-impact problems where massive data prohibits efficient learning and inference, such as GAN, recommender systems, graphs, video, and other large data streams. My research approach to uniquely address this challenge consists of (1) extracting the information volume by summarizing the most representative subsets. 2) At the core of my research lie rigorous and practical techniques that provide strong guarantees for the quality of the extracted representatives and the learned models' accuracy. My proposed methods open up new avenues for learning from representative data extracted from massive datasets.

Chapter 1

Introduction

Federated learning is an active and ongoing area of research. While recent work has begun to address the challenges, communication and computation overhead, protocols, software, infrastructures, edge device-server synchronization, fairness, data privacy, system heterogeneity, and statistical heterogeneity are among the critical open directions yet to be explored in order to make federated learning more practical and powerful. Developing methods to handle those restrictions and challenges has been the direction of my research.

Very recently, with the proliferation of edge computing billions of edge devices are connected to the Internet. Driving by this trend, there is an inspiring motivation to push the AI frontiers to the network edge in order to fully unleash the potential of the edge data. To meet this demand, federated learning, as a paradigm that pushes computing tasks and services from the network core to the network edge, has been widely recognized as a promising solution. While FL is becoming more popular it still suffers from notorious challenges including fairness, optimization of improvement, and statistical data heterogeneity, computation overhead, communication cost. Developing methods to handle those restrictions and challenges has been the direction of my research. Federated learning is particularly useful in leveraging the low data that is distributed among different distributed parties to train a deep neural net model.

On the other hand, the ever-increasing size of modern datasets motivated part of my work to be on Graph (Data) Summarization/ Coreset Construction/Low-Rank Approximation.

Part of my research concerned designing techniques that can gain insights from the underlying data structure by utilizing complex and higher-order interactions between data points. In many domains, such as stock exchange, computational social science, the internet of things (IoT), and health care, rapid streams of data are generated from various sources. For example, a typical self-driving car equipped with radar, cameras, lidar, and ultrasonic sensors produces more than 4TB of data per day. Similarly, various sensors and smart devices in IoT applications generate a huge amount of data at a high velocity. The challenge in utilizing such data is to create algorithms capable of efficiently extracting and fusing representative data from various sources in order to learn and make inferences on the fly. I developed methods to extract representative subsets of data to accelerate the training of the machine and deep learning models.

1.1 Dissertation Organization

In Chapter 2, we addressed the issue of data heterogeneity from a rarely studied dimension of FL, ordered learning [1]. We argued that one pathway to understanding the drastic accuracy drop in federated learning is by scrutinizing the behavior of the edge devices' deep models on data with different levels of "difficulty," which has been left unaddressed. We conduct extensive empirical studies and theoretical analysis over many orderings spanning three kinds of learning: curriculum, anti-curriculum, and random curriculum. We find that curriculum learning largely alleviates non-IIDness. Interestingly, the more discrepant the data distributions across edge devices are, the more they benefit from ordered learning. We provided an analysis explaining this phenomenon, specifically indicating how curriculum training appears to make the objective landscape progressively less convex, suggesting fast converging iterations at the beginning of the training procedure. We derive quantitative results of convergence for both convex and nonconvex objectives by modeling, for the first time, the curriculum training on federated devices as local SGD with locally biased stochastic gradients. Apart from that, inspired by ordered learning, we propose novel yet simple curricula for participant edge devices in FL for the first time, which is

a more sophisticated mechanism for edge device selection. Our proposed approach generalizes edge device selection strategies to the FL setting, which is still a non-trivial and open problem.

In Chapter 3, we studied clustered federated learning. In many FL applications where there may be severe data heterogeneity among edge devices, a single relevant global model may not exist. Personalized FL approaches have been studied as a solution. However, in scenarios where separate groups of edge devices have significant differences in the distributions of their local data, clustered FL could be an optimal solution as it mimics IID training within each cluster. We proposed two new approaches to clustered federated learning named PACFL [2] and FLIS [3] . In particular, FLIS partitions the edge devices into different clusters by leveraging the inference similarity of edge devices' models without having access to their private data and then trains models for every cluster of users. Further, PACFL [2] directly aims to efficiently identify distribution similarities among edge devices by analyzing the principal angles between the edge device data subspaces. The proposed approach offers great advantages, such as the fact that the initialized clusters (models) are not inherently noisy; it has the flexibility to trade off between *personalization* and *globalization*; and it presents a simple and alternative solution to the distribution similarity measures such as MMD and KL in FL setups since the server cannot make use of them due to data privacy.

PACFL improved the performance of the prior global arts on the standard benchmarks like CIFAR-100/10, SVHN, and FMNIST PACFL by 19%, as well as all the personalized competitors by around 8%. Convergence analysis for nonconvex objectives is guaranteed and since the clustering is performed on the *data* and not the parameters, it does not suffer from associated issues of multi-modality (multiple separate local minima).

In Chapter 4, we proposed a method to realize personalization in federated learning which comes by a huge reduction in computation and communication round for free [4]. The excellent performance of modern CNNs often comes at significant inference costs due to their more stacked layers and thus more learnable parameters. The use of these high-capacity networks may be significantly hampered in federated learning scenarios where, in addition to accuracy,

computational efficiency and small network sizes are critical enablers. For example, a ResNet-152 has more than 60 million parameters and entails more than 20G float-point operations (FLOPs) when trained on an image with a resolution of 224×224. This is unlikely to be affordable on resource-constrained platforms such as embedded edge devices in FL. Other than that, the FL scenario must also take care of practical issues, including statistical data heterogeneity, communication costs, and personalization. In this Chapter, we argued that the mentioned three challenges point in the same direction, i.e., we showed that statistical data heterogeneity can be a blessing factor [4] and proposed a new framework for personalized federated learning under data heterogeneity that is communication efficient. To realize this personalization, we leveraged finding a small subnetwork for each edge device by applying hybrid pruning (a combination of structured and unstructured pruning). Through extensive experiments on various DNNs and benchmark datasets, we observed the existence of similar subnetworks for edge devices with similar data (labels).

We remove the commonly shared interfering parameters of each layer and keep the personalized parameters that can represent the features of local data in each edge device by iteratively pruning the parameters and channels. Since the edge devices with similar data (label overlap) share similar personalized parameters, by the proposed Sub-FedAvg [4] method, we would average the models of edge devices at the server based on the intersection of the remaining channels and parameters of the edge devices. This method of aggregation at the server not only does not impact the accuracy performance of each edge device in a Non-IID setting but also improves it by up to 35% on CIFAR-10/100, EMNIST, and MNIST. Sub-FedAvg achieves a significant reduction in the required communication round by $2-10$ times on benchmark datasets, as well as $2.4\times$ FLOP reduction on the benchmark LeNet-5 architecture.

In Chapter 5, we proposed a new notion and framework for Non-IID partitioning in FL setups by analyzing the principal angles between subspaces of different classes of the dataset [5].

Finally, in Chapter 6, we studied data/graph summarization. In many problems in sociology, finance, computer science, and operations research, we have networks of interconnected

entities and pairwise relations between them. Due to the proliferation of real-world network data and the natural representations of pairwise relationships in data, we are typically interested in working with data as represented by a *graph structure*. A problem that arises often in practice is calculating the expected value of a variable in the form of the sum of the values of a smooth function on the nodes of a graph. For example, in semi-supervised learning with Graph Neural Networks (GCNs), the generalization error is specified as the average of the loss functions associated with all the nodes in a graph. In real-world networks containing millions of nodes and billions of edges, it is impractical to evaluate the function at every single node. Therefore, *an important question is how to select a small representative subset (coreset) of nodes from a million-node graph such that the weighted sum of function values sampled at the nodes of the subset can be a good estimate of the sum of function values over the entire graph?*

Bayesian coresets were an easy-to-implement and computationally simple solution to the problem, but their construction may not reflect the guarantees sought by a model-specific based task. Such guarantees are critical for active learning, where collecting functions or labels is expensive and the hope is that an accurate estimate can be found by querying just a few labels. Inference algorithms such as standard MCMC algorithms, variational methods, subsampling, and streaming methods fail to bound the inferential error made in the dataset reduction. These methods entail expensive iterative access to a constant fraction of the data and have no guarantees on the quality of their inferential results.

In this Chapter, we introduced a scalable optimization algorithm [6] with no correction steps (in contrast to FrankWolfe and its variants), a variant of gradient ascent for *coreset selection in graphs*, that greedily selects a weighted subset of vertices that deemed most important to sample.

Chapter 2

Curricula in Federated Learning

2.1 Introduction

Inspired by the learning principle underlying the cognitive process of humans, curriculum learning (CL) generally proposes a training paradigm for machine learning models in which the difficulty of the training task is progressively scaled, going from "easy" to "hard". Prior empirical studies have demonstrated that CL is effective in avoiding bad local minima and in improving the generalization results [7, 8]. Also interestingly, another line of work proposes the exact opposite strategy of prioritizing the harder examples first, such as [9, 10, 11]–these techniques are referred to as “anti-curriculum". It is shown that certain tasks can benefit from anti-curriculum techniques. However, in tasks such as object detection [12, 13], and large-scale text models [14] CL is standard practice.

Although the empirical observations on CL appear to be in conflict, this has not impeded the study of CL in machine learning tasks. Certain scenarios [15] have witnessed the potential benefits of CL. The efficacy of CL has been explored in a considerable breadth of applications, including, but not limited to, supervised learning tasks within computer vision [16], healthcare [17], reinforcement learning tasks [18], natural language processing (NLP) [19] as well as other applications such as graph learning [20], and neural architecture search [21].

Curriculum learning has been studied in considerable depth for the standard centralized training settings. However, to the best of our knowledge, our work is the first attempt at studying the methodologies, applications, and efficacy of CL in a decentralized training setting and in

particular for federated learning (FL). In FL, the training time budget and the communication bandwidth are the key limiting constraints, and as demonstrated in [15] CL is particularly effective in settings with a limited training budget. It is an interesting proposition to apply the CL idea to an FL setting, and that is exactly what we explore in our work (in Section 2.3).

The idea of CL is agnostic to the underlying algorithms used for federation and hence can be very easily applied to any of the state-of-the-art solutions in FL. Our technique does not require a pre-trained expert model and does not impose any additional synchronization overhead on the system. Also, as the CL is applied to the client, it does not add any additional computational overhead to the server.

Further, we propose a novel framework for efficient client selection in an FL setting that builds upon our idea of CL in FL. We show in Section 2.6, CL on clients is able to leverage the real-world discrepancy in the clients to its advantage. Furthermore, when combined with the primary idea of CL in FL, we find that it provides compounding benefits.

Contributions: In this Chapter, we comprehensively assess the efficacy of CL in FL and provide novel insights into the efficacy of CL under the various conditions and methodologies in the FL environment.

We provide a rigorous convergence theory and analysis of FL in non-IID settings, under strongly convex and non-convex assumptions, by considering local training steps as biased SGD, where CL naturally grows the bias over the iterations.

We hope to provide comprehensible answers to the following six important questions:

Q1: *Which of the curriculum learning paradigm is effective in FL? And under what conditions?*

Q2: *Can CL alleviate the statistical data heterogeneity in FL?*

Q3: *Does the efficacy of CL in FL depend on the underlying client data distributions?*

Q4: *Whether the effectiveness of CL is correlated with the size of datasets owned by each client?*

Q5: *Are there any benefits of smart client selection? And can CL be applied to the problem of client selection?*

Q6: *Can we apply the ideas of CL to both the client data and client selection?*

We test our ideas on two widely adopted network architectures on popular datasets in the FL community (CIFAR-10, CIFAR-100, and FMNIST) under a wide range of choices of curricula and compare them with several global state-of-the-art baselines. We have the following findings:

- CL in FL boosts the classification accuracy under both IID and Non-IID data distributions.
- The efficacy of CL is more pronounced when the client data is heterogeneous.
- CL on client selection has a synergic effect that compounds the benefits of CL in FL.
- CL can alleviate data heterogeneity across clients and CL is particularly effective in the initial stages of training as the larger initial steps of the optimization are done on the "easier" examples which are closer together in distribution.
- The efficacy of our technique is observed in both lower and higher data regimes.

2.2 Related Work

Early CL formulated the easy-to-hard training paradigm in the context of deep learning [7]. CL determines a sequence of training instances, which in essence corresponds to a list of samples ranked in ascending order of learning difficulty [8]. Samples are ranked according to per-sample loss [22]. In the early steps of training, samples with smaller loss (higher scores) are selected, and gradually the subset size over time is increased to cover all the training data. [23] proposed to manually sort the samples using human annotators. Self-paced learning (SPL) [8] chooses the curriculum based on hardness (e.g., per-sample loss) during training. [24] proposes using a consistency score (c-score) calculated based on the consistency of a model in correctly predicting a particular example's label trained on i.i.d. draws of the training set. [25] determines

the difficulty of learning an example by the metric of the earliest training iteration, after which the model predicts the ground truth class for that example in all subsequent iterations.

2.3 Curriculum Components

Federated Learning (FL) techniques provide a mechanism to address the growing privacy and security concerns associated with data collection, all-the-while satiating the need for large datasets for training powerful machine learning models. A major appeal of FL is its ability to train a model over a distributed dataset without needing the data to be collated into a central location for training. In the FL framework, we have a server and multiple clients with a distributed dataset. The process of federation is an iterative process that involves multiple rounds of back-and-forth communication between the server and the clients that participate in the process [5]. This back-and-forth communication incurs a significant communication overhead, thereby limiting the number of rounds that are pragmatically possible in real-world applications. Curriculum learning is an idea that particularly shines in these scenarios where the training time is limited [15]. Motivated by this idea, we define a curriculum for training in the federated learning setting. A curriculum consists of three key components:

The scoring function: It is a mapping from an input sample, $x_i \in \mathscr{D} = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, to a numerical value, $s_i(x_i) \in R^+$. We define a range of scoring functions when defining a CL for the FL setting in the subsequent sections. When defining the scoring function of a curriculum in FL, we look for loss-based dynamic measures for the score that update every iteration, unlike the methods proposed in [24] which produce a fixed score for each sample. This is because the instantaneous score of samples changes significantly between iterations, and using a fixed score leads to an inconsistency in the optimization objectives, making training less stable [22]. Also, we avoid techniques like [23] which requires human annotators, as it is not practical in a privacy-preserving framework.

The pacing function: The pacing function $g_\lambda(t)$ determines scaling of the difficulty of the

training data introduced to the model at each of the training steps t and it selects the highest scoring samples for training at each step. The pacing function is parameterized by $\lambda = (a, b)$ where a is the fraction of the training budget needed for the pacing function to begin sampling from the full training set and b represents the fraction of the training set the pacing function exposes to the model at the start of training. The full training set size and the total number of training steps (budget) are denoted by N and T, respectively. Further, we consider five pacing function families, including exponential, step, linear, quadratic, and root (sqrt). The expressions we used for the pacing functions are shown in Table 2.1 and Fig. 2.1. We follow [26] in defining the notion of pacing function and use it to schedule how examples are introduced to the training procedure.

Table 2.1. The five families of pacing functions we employed in this Chapter. The parameter a determines the fraction of training time until all data is used. Parameter b sets the initial fraction of the data used.

Pacing Function	Expression
Exponential	$Nb + \frac{N(1-b)}{e^{10}-1}(e^{\frac{10t}{aT}} - 1)$
Step	$Nb + N\lfloor \frac{t}{aT} \rfloor$
Root (Sqrt)	$Nb + \frac{N-b}{\sqrt{aT}}\sqrt{t}$
Linear	$Nb + \frac{N-b}{aT}t$
Quadratic	$Nb + \frac{N-b}{(aT)^2}t^2$

The order: Curriculum learning orders sample from the highest score (easy ones) to lowest score, anti-curriculum learning orders from lowest score to highest, and finally, random curriculum randomly samples data in each step regardless of their scores.

2.4 Experiment

Experimental setting. To ensure that our observations are robust to the choice of architectures, and datasets, we report empirical results for LeNet-5 [27] architecture on CIFAR-10 [28] and Fashion MNIST (FMNIST) [29], and ResNet-9 [30] architecture for CIFAR-100 [28] datasets. All models were trained using SGD with momentum.

Baselines and Implementation. To provide a comprehensive study on the efficacy of CL on

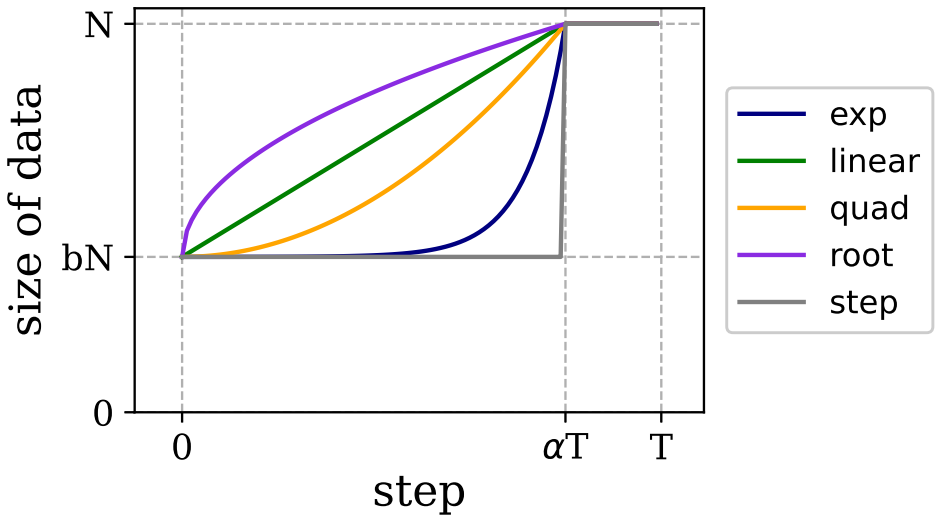


Figure 2.1. Pacing function curves of different families are used throughout the Chapter. As shown, the hyperparameter α specifies the fraction of the training step until all data is used in training. The hyperparameter b determines the initial fraction of the data used in training.

Algorithm 1. The Curriculum FL Framework

Input: M clients indexed by m, sampling rate $R \in (0,1]$, participating-client number K, communication rounds R_C, server model f with θ_g, pacing function $g_\lambda : [T] \to [N]$, scoring function $s : [N] \to \mathbb{R}$, order $o \in \{$"*curriculum*","*anti*","*random*"$\}$,

Server executes:

initialize f with θ

for each round $t = 0, 1, 2, \ldots$ **do**

$\mathbb{S}_t \leftarrow$ (random set of K clients)

for each client $m \in \mathbb{S}_t$ **in parallel do**

broadcast θ_g^t to clients

$\theta_m^{(t)} \leftarrow \text{ClientUpdate}(m, \theta_g^{(t)})$

$\theta_g^{(t+1)} = \sum_{m=1}^{K} \frac{|\mathscr{D}_m|}{\sum_{i=1}^{K} |\mathscr{D}_i|} \theta_m^{(t)}$ $\{\}\mathscr{D}_m$ is the set of the local data on the client with index m.

return θ_g^{t+1}

ClientUpdate (m, θ_g^t)**:**

Obtain the score of each data sample using θ_g^t and/or θ_m^t as described in section 2.4.1

$(\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}) \leftarrow \text{sort}(\{\mathbf{x_2}, \ldots, \mathbf{x_n}\}, \text{s}, \text{o})$

for $t = 1, 2, \ldots, T$ **do**

$\theta_m^t \leftarrow \text{train}(\theta_g^t, \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x}_{g(t)}\})$

FL setups, we consider the predominant approaches to FL that train a global model, including FedAvg [31], FedProx [32], SCAFFOLD [33], and FedNova [34]. In all experiments, we assume 100 clients are available, and 10% of them are sampled randomly at each round. Unless stated otherwise, throughout the experiments, the number of communication rounds is 100, each client performs 10 local epochs with a batch size of 10, and the local optimizer is SGD. To better understand the mutual impact of different data partitioning methods in FL and CL, we consider both federated heterogeneous (Non-IID) and homogeneous (IID) settings. In each dataset other than IID data partitioning settings, we consider two different federated heterogeneity settings as in [2, 35]: Non-IID label skew (20%), and Non-IID Dir(β).

2.4.1 Effect of scoring function in IID and Non-IID FL

In this section, we investigate five scoring functions. As discussed earlier, in standard centralized training, samples are scored using the loss value of an expert pre-trained model. Given a pre-trained model $f_\theta : \mathscr{X} \to \mathscr{Y}$, the score is defined as $s_i(x_i) = \frac{r_i}{\sum_i r_i}$, where $r_i = \frac{1}{\mathscr{L}(y_i, f_\theta(x_i))}$, with $\mathscr{L}$ being the inference loss. In this setup, a higher score corresponds to an easier sample.

In FL [31, 36], a trusted server broadcasts a single initial global model, θ_g, to a random subset of selected clients in each round. The model is optimized in a decentralized fashion by multiple clients who perform some steps of SGD updates ($\theta_k = \theta_g - \eta \nabla \mathscr{L}_k$). The resulting model is then sent back to the server, ultimately converging to a joint representative model. With that in mind, in our setting, the scores can be obtained by clients either via the global model that they receive from the server, which we name as s_G or by their own updated local model, named s_L or the score can be determined based on the average of the local and global model loss, named as s_{LG}[1].

We further consider another family of scoring that is based on ground truth class prediction. In particular, in each round, clients receive the global model from the server and get the prediction using the received global model and the current local model as $\hat{y}_G$ and $\hat{y}_L$,

[1]Since it produces very similar results to s_G, we skipped it.

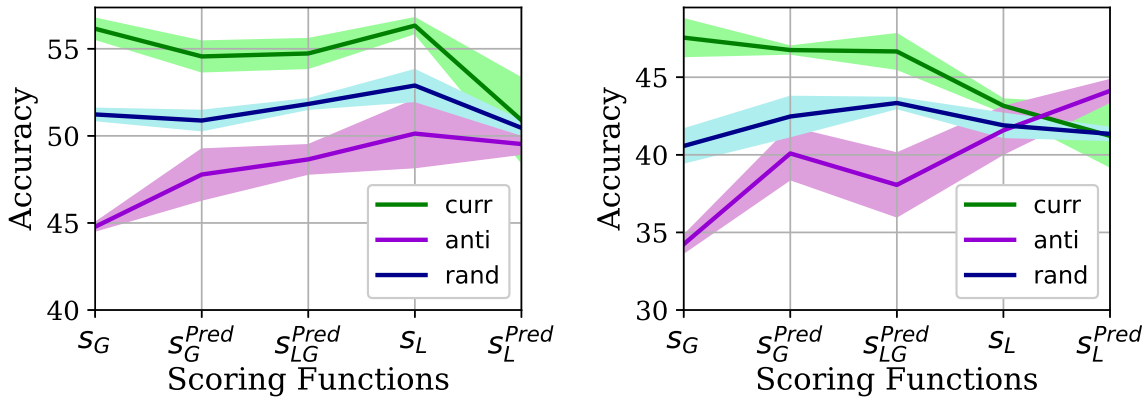


Figure 2.2. Scoring client samples based on the global model (s_G) provides the most accurate scores for all levels of Non-IIDness. Scoring based on the local model (s_L^{pred}) provides the least accurate scores, especially when data are Non-IID, as it provides the worst accuracy. Evaluating the effect of using different scoring methods on accuracy when the clients employ curriculum, anti-curriculum, or random ordering during their local training on CIFAR-10 with IID data (left) and Non-IID (2) (right). All curricula use the linear pacing function with $a = 0.8$ and $b = 0.2$. We run each experiment three times for 100 communication rounds with 10 local epochs and report the mean and standard deviation (std) for global test accuracy. Note that the results for vanilla FedAvg for the left figure, and the right one are 52.30 ± 0.86, and 41.96 ± 1.77, respectively.

respectively. For those samples whose $\hat{y}_L$ and $\hat{y}_G$ do not match, the client tags them as hard samples and otherwise as easy ones. This scoring method is called s_{LG}^{pred}. Further, ground truth class prediction and scoring can be solely done by the global model or the client's local model, which end up with two other different scoring methods, namely s_G^{pred}, and s_L^{pred} respectively. This procedure is described in Algorithm 1.

Fig. 2.2 demonstrates what the impact of using these various scoring methods is on the global accuracy when curriculum, anti-curriculum, or random ordering is exploited by the clients in the order in which their CIFAR-10 examples are learned with FedAvg under IID and Non-IID (2) data partitions. The results are obtained by averaging over three randomly initialized trials and calculating the standard deviation (std).

The results reveal that **first,** *the scoring functions are producing broadly consistent results except for s_L^{pred} for both IID and Non-IID and s_L for Non-IID settings. s_G provides the most accurate scores, thereby improving the accuracy by a noticeable margin compared to its peers.* This is quite expected, as the global model is more mature compared to the client model, **second,** *the curriculum learning improves the accuracy results consistently across different scoring functions,* **third,** *curriculum learning is more effective when the clients underlying data*

distributions are Non-IID. To ensure that the latter does not occur by chance, we will delve into this point in detail in subsection 2.4.3. Due to the superiority of s_G relative to others, we set the scoring function to be s_G henceforth. We will further elaborate on the precision of s_G compared to an expert model in Section 2.6.4.

2.4.2 Effect of pacing function and its parameters in IID and Non-IID FL

In order to study the effect of different families of pacing functions along with the hyperparameters $\lambda = (a,b)$, we test the exponential, step, linear, quadratic, and root function families. We further first fix b to 0.2 and let $a \in \{0.1, 0.5, 0.8\}$[2]. The accuracy results are presented in Fig. 2.3 and Fig. 2.4. As is evident, for all pacing function families, the trends between the curriculum and the other orderings, i.e., (anti, random)-curriculum are markedly opposite in how they improve/degrade by sweeping a from small values to large ones. The pattern for Non-IID which is presented in Fig. 2.4 is almost similar to that of IID. Values of $a \in [0.5, 0.8]$ produce the best results. As can be seen from Fig. 2.3, the best accuracy achieved by curriculum learning outperforms the best accuracy obtained by other orderings by a large margin. For example, in the “linear" pacing function, the best accuracy achieved for curriculum learning when $a = 0.8$ is 56.60 ± 0.91 which improved the vanilla results by 4% while that of random when $a = 0.1$ is 52.73 ± 0.81 and improved vanilla by 0.5%. Henceforth, we set $a = 0.8$ and the pacing function to linear. After selecting the pacing function and a the final step is to fix these two and see the impact of b. Now we let all curricula use the linear pacing functions with a = 0.8 and only sweep $b \in \{0.0025, 0.1, 0.2, 0.5, 0.8\}$ and report the results in Fig. 2.5. Perhaps most striking is that curriculum learning tends to have smaller values of b to improve accuracy, which is in contrast with (random-/anti) orderings. The performance of the anti-curriculum shows a significant dependence on the value of b. Further, curriculum learning provides a robust benefit for different values of b and it beats the vanilla FedAvg by $4-7$% depending upon the

[2]Note that $b \in [0,1]$. Also, $a = 0$ or $b = 1$ is equivalent to no ordered training, i.e., standard training.

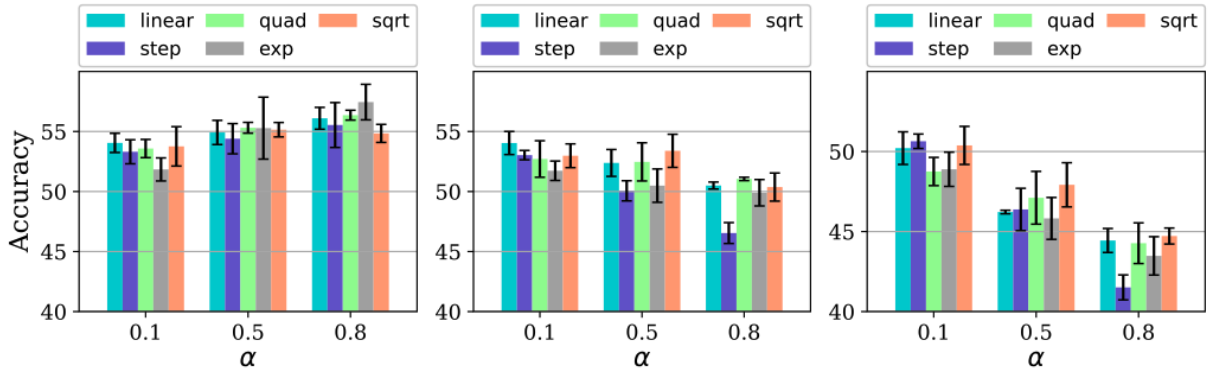


Figure 2.3. Bigger a values provide better accuracy performance for all pacing function families on IID settings for curriculum learning. But a notable contrast can be seen with random-/anti ordering. The effect of using different pacing function families and their hyperparameter a on accuracy when the clients employ curriculum, anti-curriculum or random ordering during their local training on CIFAR-10 with IID data. We run each experiment three times for 100 communication rounds with 10 local epochs and report the mean and std for global test accuracy. The figures from left to right are for curriculum, random, and anti ones.

distribution of the data. Henceforth, we fix b to 0.2.

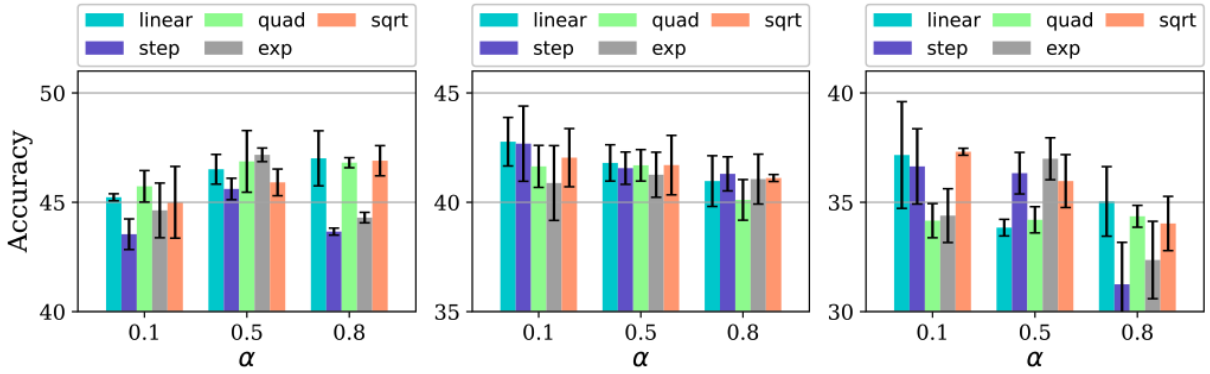


Figure 2.4. Bigger a values provide better accuracy performance for most of pacing function families and on both IID and Non-IID setting for curriculum learning. But a notable contrast can be seen with random-/anti ordering. The effect of using different pacing function families and their hyperparameter a on the accuracy when the clients employ curriculum, anti-curriculum or random ordering during their local training on CIFAR-10 with Non-IID Dir(0.05) data. The figures from left to right are for curriculum, random, and anti ones.

2.4.3 Effect of level of data heterogeneity

Equipped with the ingredients explained in the preceding section, we are now in a position to investigate the significant benefits of employing curriculum learning in FL when the data distribution environment is more heterogeneous. To ensure a reliable finding, we investigate the impact of heterogeneity in four baselines through extensive experiments on benchmark datasets,

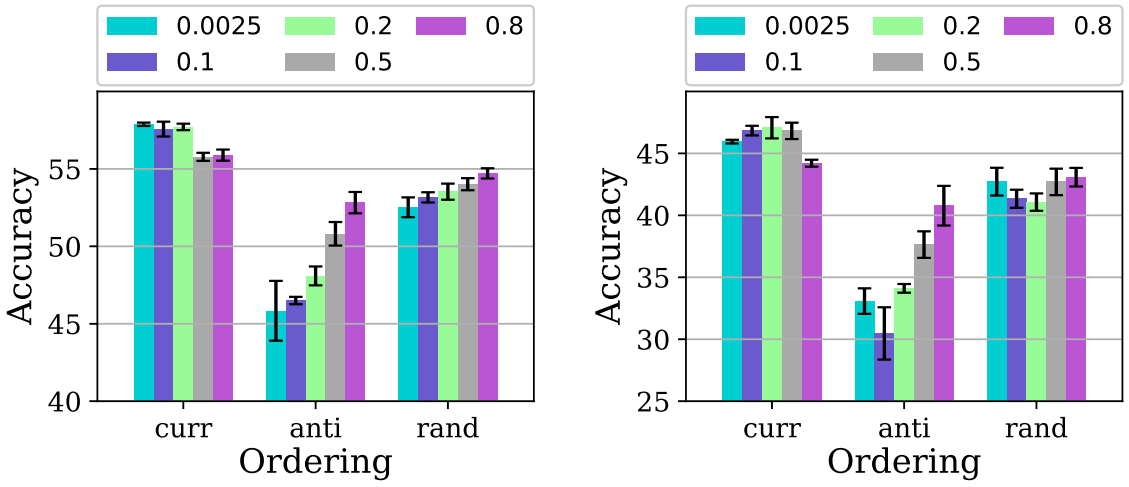


Figure 2.5. Smaller b values provide better accuracy performance for both IID and Non-IID settings as further corroborate the benefit of employing curriculum learning. Evaluating the effect of hyperparameter b on accuracy when the clients employ curriculum, anti-curriculum, or random ordering during their local training on CIFAR-10 with IID for FedAVg (left), and with Dir(0.05) for FedAvg (right). All curricula use the linear pacing functions with a = 0.8. We run each experiment three times for 100 communication rounds with 10 local epochs and report the mean and std for global test accuracy.

i.e., CIFAR-10, CIFAR-100, and FMNIST. In particular, we distribute the data between clients according to Non-IID Dir(β) defined in [37]. In Dir(β), heterogeneity can be controlled by the parameter β of the Dirichlet distribution. Specifically, when $\beta \to \infty$ the clients' data partitions become IID, and when $\beta \to 0$ they become extremely Non-IID.

To understand the relationship between the ordering-based learning and the level of statistical data heterogeneity, we ran all baselines for different Dirichlet distribution β values $\beta \in \{0.05, 0.2, 0.9\}$. The accuracy results of different baselines on CIFAR-10 while employing (anti-) curriculum, or random learning with linear pacing functions $(0.8, 0.2)$ and using s_G are presented in Tables 2.2, 2.3, 2.4, and 2.5. For the comprehensiveness of the study, we will present the results for CIFAR-100 in Fig. 2.6. It shows the same trend as in CIFAR-10.

The results are surprising: **The benefits of ordered learning are more prominent with increased data heterogeneity**. The greater the distribution discrepancy between clients, the greater the benefit to curriculum learning.

If we consider client heterogeneity as distributional skew[38], then this is logical: easier data samples are those with overall lower variance, both unbiased and skew from the mean, and

Table 2.2. Curriculum-learning helps more when training with more severe data heterogeneity across clients. Understanding the benefit of ordered learning with increasing data heterogeneity ($\beta = 0.9 \rightarrow 0.05$) when clients are trained on CIFAR-10 with `FedAvg` method.

Non-IIDness	Curriculum	Anti	Random	Vanilla
Dir($\beta = 0.05$)	46.34 ± 1.55	31.16 ± 3.16	41.91 ± 2.23	39.56 ± 4.91
Dir($\beta = 0.2$)	51.09 ± 0.39	42.34 ± 1.48	46.35 ± 1.44	46.75 ± 0.72
Dir($\beta = 0.9$)	55.36 ± 0.69	46.86 ± 0.35	52.42 ± 0.90	52.19 ± 0.73

Table 2.3. Curriculum-learning helps more when training with more severe data heterogeneity across clients. Understanding the benefit of ordered learning with increasing data heterogeneity ($\beta = 0.9 \rightarrow 0.05$) when clients are trained on CIFAR-10 with `Fedprox` method.

Non-IIDness	Curriculum	Anti	Random	Vanilla
Dir($\beta = 0.05$)	47.94 ± 0.96	36.08 ± 1.52	42.62 ± 0.35	41.48 ± 0.29
Dir($\beta = 0.2$)	50.02 ± 0.15	40.92 ± 0.90	46.41 ± 1.12	46.18 ± 0.90
Dir($\beta = 0.9$)	56.48 ± 0.18	48.37 ± 0.91	51.69 ± 0.40	53.07 ± 1.25

thus the total collection of CL-easier data samples in a dataset is more IID than the alternative. Thus, in the crucial early phases of training, the training behaves closer to FedAvg/FedProx-/SCAFFOLD/FedNova under IID distributions. Therefore, *CL can alleviate the drastic accuracy drop when clients' decentralized data are statistically heterogeneous, which comes from stable training from IID samples to Non-IID ones, fundamentally improving the accuracy.* This is formalized with quantitative convergence rates in the Section 2.7 and Section 2.7.1.

Table 2.4. Curriculum learning helps more when training with more severe data heterogeneity across clients. Understanding the benefit of ordered learning with increasing data heterogeneity ($\beta = 0.9 \rightarrow 0.05$) when clients are trained on CIFAR-10 with `FedNova` method.

Non-IIDness	Curriculum	Anti	Random	Vanilla
Dir($\beta = 0.05$)	43.73 ± 0.09	28.31 ± 1.93	37.81 ± 3.06	31.97 ± 0.90
Dir($\beta = 0.2$)	47.01 ± 1.89	36.55 ± 1.42	44.21 ± 1.00	41.28 ± 0.30
Dir($\beta = 0.9$)	50.74 ± 0.19	41.76 ± 0.90	48.87 ± 0.88	47.230 ± 1.80

Table 2.5. Curriculum-learning helps more when training with more severe data heterogeneity across clients. Understanding the benefit of ordered learning with increasing data heterogeneity ($\beta = 0.9 \rightarrow 0.05$) when clients are trained on CIFAR-10 with `SCAFFOLD` method.

Non-IIDness	Curriculum	Anti	Random	Vanilla
Dir($\beta = 0.05$)	45.91 ± 1.17	21.29 ± 1.82	38.27 ± 2.19	41.33 ± 1.30
Dir($\beta = 0.2$)	49.69 ± 1.81	28.69 ± 0.60	45.29 ± 1.93	46.62 ± 0.58
Dir($\beta = 0.9$)	52.05 ± 1.14	30.75 ± 0.79	49.25 ± 0.76	50.24 ± 0.57

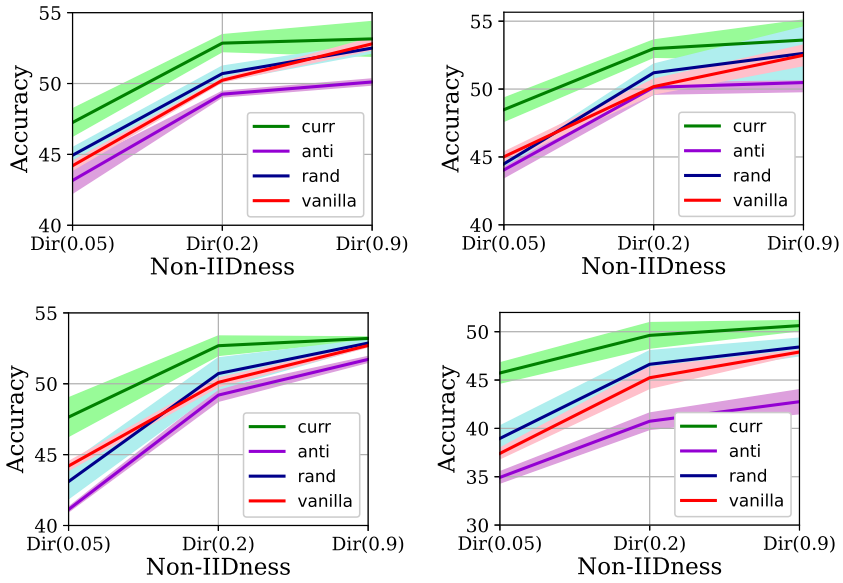


Figure 2.6. Curriculum-learning helps more when training with more severe data heterogeneity across clients on CIFAR-100. Test accuracy of different baselines when sweeping from extremely Non-IID setting, Dir (0.05) to highly IID setting, Dir(0.9). For each baseline, the average of final global test accuracy is reported. We run each baseline 3 times for 100 communication rounds with 10 local epochs. The figures from left to right, are for FedAvg, Fedprox, Scaffold, and FedNova baselines.

2.5 Effect of amount of data on clients end

In this section, we are interested in understanding whether the previous conclusions we made for CIFAR10 generalize to both high and low data regimes on the client's end. In particular, we divide the larger dataset into multiples of the number of clients and randomly assign M of those data partitions to the M clients. The larger the number of partitions, the smaller the amount of data on each of the clients. As can be seen from Fig. 2.7 the amount of data that each client owns has no relationship with the benefit it gains from curriculum learning. In fact, CL ameliorates the classification accuracy performance equally under both lower and higher data regimes on the clients' end.

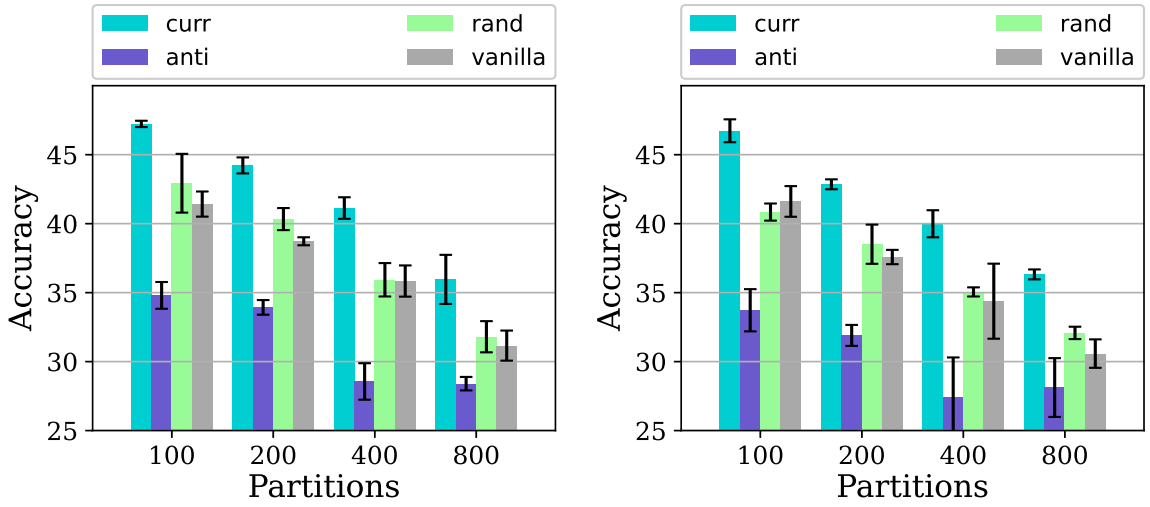


Figure 2.7. There is no correlation between the amount of data on the client's end and the benefit they gain from ordered learning. The accuracy decreases when the amount of data each client owns is reduced, but it gains the same amount of benefit from curriculum learning with more data. Evaluating the impact of the amount of data each client owns on the accuracy when the clients employ curriculum, anti-curriculum or random ordering during their local training on CIFAR-10 with Non-IID (2) for FedAVg (left), and with Dir(0.05) for Fedprox (right). All curricula use the linear pacing functions with a = 0.8 and b = 0.2. Each experiment is repeated three times for a total of 100 communication rounds with ten local epochs, and the mean and standard deviation for global test accuracy are reported.

2.6 Curriculum on Clients

The technique of ordered learning presented in previous sections is designed to exploit the heterogeneity of data at the clients but is not geared to effectively leverage the heterogeneity

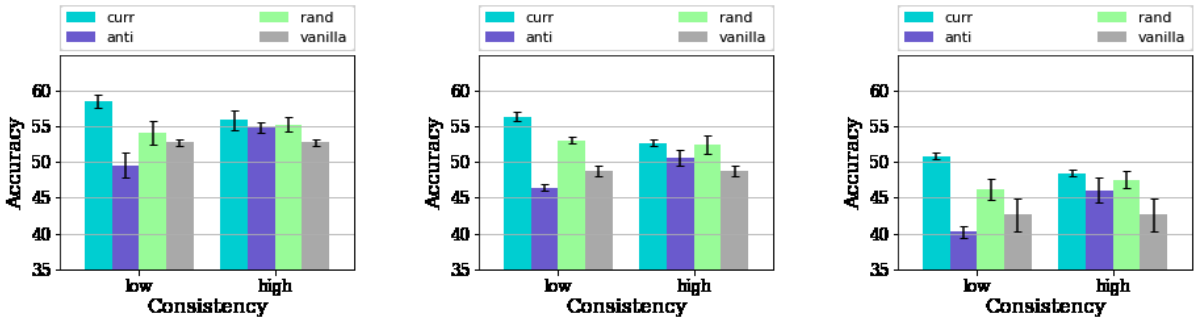
between the clients that, as we discuss further, naturally emerges in the real world.

In the literature, some recent works have dabbled with the idea of smarter client selection, and many selection criteria have been suggested, such as importance sampling, where the probabilities for clients to be selected are proportional to their importance measured by the norm of update [39], test accuracy [40]. The [41] paper proposes client selection based on local loss where clients with higher loss are preferentially selected to participate in more rounds of the federation, which is in stark contrast to [42] in which the clients with a lower loss are preferentially selected. It's clear from the literature that the heterogeneous environment in FL can hamper the overall training and convergence speed [43, 44], but the empirical observations on client selection criteria are either in conflict or their efficacy is minimal. In this section, inspired by curriculum learning, we try to propose a more sophisticated mechanism of client selection that generalizes the above strategies to the FL setting.

2.6.1 Motivation

In the real world, the distributed dataset used in FL is often generated in situ by clients, and the data is measured/generated using a particular sensor belonging to the client. For example, consider the case of a distributed image dataset generated by smartphone users using the onboard camera or a distributed medical imaging dataset generated by hospitals with their in-house imaging systems. In such scenarios, as there is a huge variety in the make and quality of the imaging sensors, the data generated by the clients is uniquely biased by the idiosyncrasies of the sensor, such as the noise, distortions, quality, etc. This introduces variability in the quality of the data at clients, in addition to the Non-IID nature of the data. However, it is interesting to note that these effects apply consistently across all the data at the specific client.

From a curriculum point of view, as the data points are scored and ordered by difficulty, which is just a function of the loss value of that data point, these idiosyncratic distortions uniformly affect the loss/difficulty value of all the data at that particular client. Also, it is possible that the difficulty among the data points at the particular client is fairly consistent as the level



((a)) IID ((b)) Dir(0.2) ((c)) Dir(0.05)

Figure 2.8. Consistency in data difficulty at the client hurts the efficacy of curricula. The effect of consistency in the difficulty distribution at the client nullifies the effect of curricula. The values plotted are for FedAvg on CIFAR-10. The standard deviation values of (Low, High) consistency for the IID are $(0.52, 0.01)$, Dir(0.2) are $(0.51, 0.14)$, and Dir(0.05) are $(0.50, 0.13)$. Note that we use Algorithm 2 to construct partitions with varying difficulty, and as detailed in Section 2.6.3 it is not possible to control the partition difficulty value with arbitrary precision, hence the above minor variations. The Low consistency scenario is generated using $f_{ord} = 0.0$ and the high consistency scenario uses $f_{ord} = 1.0$.

of noise, quality of the sensor, etc. are the same across the data points. This bias in difficulty varies between clients but is often constant within the same client. Naturally, this introduces a heterogeneity in the clients participating in the federation. In general, this can be thought of as some clients being "easy" and some being "difficult". *We can quantify this notion of consistency in difficulty by the standard deviation in the score of the data points at the client.*

When the standard deviation of the intra-client data is low, i.e., when the difficulty of the data within a client is consistent, we find that the curriculum on FL behaves very differently in these kinds of scenarios. We observe the advantage of curriculum diminishes significantly and has similar efficacy as that of random curricula as shown in Fig 2.8. The advantage of curriculum can be defined as $A_o = accuracy(o) - accuracy(vanilla)$, where $o \in \{curr, anti, rand\}$.

2.6.2 Client Curriculum

We propose to extend the ideas of curriculum onto the set of clients, in an attempt to leverage the heterogeneity in the clients. To the best of our knowledge, our work is the first attempt to introduce the idea of `curriculum on clients` in an FL setting. Many curricula ideas can be neatly extended to apply to the set of clients. In order to define a curriculum over the

clients, we need to define a scoring function, a pacing function, and an ordering over the scores. We define the client loss as the mean loss of the local data points at the client (Eq. 2.1), and the client score can be thought of as inversely proportional to the loss. The ordering is defined over the client scores.

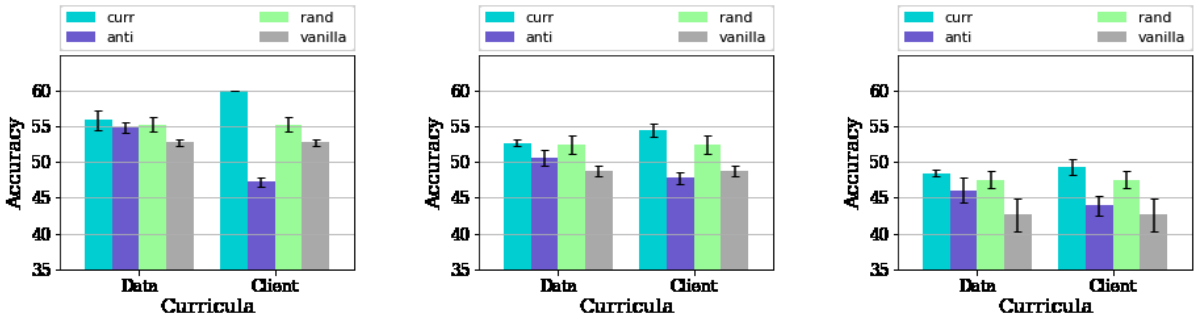
$$\mathscr{L}_k = \frac{1}{\|\mathbb{D}_m\|} \sum_{j}^{\|\mathbb{D}_m\|} l_j \tag{2.1}$$

where m is the index for client, $\mathbb{D}_m$ represents the dataset at client m, l_j is the loss of jth data point in $\mathbb{D}_m$.

The pacing function, as in the case of data curricula, is used to pace the number of clients participating in the federation. Starting from a small value, the pacing function gradually increases the number of clients participating in the federation. The action of the pacing function amounts to scaling the participation rate of the clients.

The clients are scored and rank-ordered based on the choice of ordering, then a subset of size $K^{(t)}$ of the K clients are chosen in a rank-ordered fashion. The value $K^{(t)}$ is prescribed by the pacing function. The $K^{(t)}$ clients are randomly batched into mini-batches of clients. These mini-batches of clients subsequently participate in the federation process. Thereby, we have two sets of curricula, one that acts locally on the client data and the other that acts on the set of clients. Henceforth we will refer to these as the `data curriculum` and the `client curriculum`, respectively. We study the interplay of these curricula in Section 2.6.5.

Fig. 2.9 confirms that the benefits of the algorithm severely depend on the data of the client having a diverse set of difficulties. As is evident from Fig. 2.9, we are able to realize an A_{curr} of $5.67 - 7.19\%$ for the different values of Non-IIDness using our proposed client curriculum algorithm in the scenario with high consistency in the client data where the data curriculum has reduced/no benefits. *This illustrates that the client curriculum is able to effectively overcome the limiting constraint of local variability in data and is able to leverage the heterogeneity in the set of clients to its benefit.*



((**a**)) IID ((**b**)) Dir(0.2) ((**c**)) Dir(0.05)

Figure 2.9. Client curriculum does not suffer from low heterogeneity in the data difficulty and is effective when data curriculum is not. The scenario shown here is the same as the scenario with high local consistency from Fig 2.8. As we observe the client curriculum is able to overcome the limitations of the data curriculum.

2.6.3 Difficulty based partitioning

For the experiments in this section, we require client datasets ($\mathbb{D}_m$) of varying difficulty at the desired level of Non-IIDness. In order to construct partitions of varying difficulty, we need to address two key challenges: one, we need a way to accurately assess the difficulty of each of the data points, and two, we need to be able to work with different levels of Non-IIDness. To address the first challenge, we rank the data points in order of difficulty using an a priori trained expert model $\theta_{\mathscr{E}}$ that was trained on the entire baseline dataset and has the same network topology as the global model. As the expert model has the same topology as the model that we intend to train and as it is trained on the entire dataset, it is an accurate judge of the difficulty of the data points. Interestingly, this idea can be extended to be used as a scoring method for curriculum as well. We call this scoring method the expert scoring s_E. We look at this in greater detail in Section 2.6.4.

To address the second challenge, a possible solution is to first partition the standard dataset into the desired Non-IID partitions using well-known techniques such as the Dirichlet distribution, followed by adding different levels of noise to the samples of the different data partitions. This would partition with varying difficulty; however, doing so would alter the standard baseline dataset, and we would lose the ability to compare the results to known baseline values and between different settings. We would like to be able to compare our performance

results with standard baselines, so we require a method that does not alter the data or resort to data augmentation techniques, and we devise a technique that does just that.

Starting with the baseline dataset, we first divide it into the desired Non-IID partitions the same as before, but then instead of adding noise to the dataset, we attempt to reshuffle the data among partitions in such a way that we create "easy" partitions and "hard" partitions. This can be achieved by ordering the data in increasing order of difficulty and distributing the data among the partitions starting from the "easy" data points, all the while honoring the Non-IID class counts of each of the partitions as determined by the Non-IID distribution. The outline is detailed in Algorithm 2. It is noteworthy that, although we are able to generate partitions of varying difficulty, we do not have direct control over the "difficulty" of each of the partitions and hence cannot generate partitions with an arbitrary distribution of difficulty as can be done by adding noise.

2.6.4 Expert guided and self-guided curricula

The scoring method s_E, as discussed above, can also be used to guide the learning process in a curriculum learning setting. As the expert model used for scoring shares the same network topology as the global model that we intend to train, and as the expert was trained on the entire dataset, the expert-guided curricula can be thought of as a pedagogical mode of learning.

The global model accuracy at different rounds of federation is depicted in Fig. 2.10. We see a clear trend in Fig 2.10 that s_E outperforms s_G in the initial rounds, but s_G converges to s_E over the rounds. Also, s_G accuracy in the initial rounds very closely approximates the random scoring accuracy. The s_G scoring method is a self-guided curriculum, that uses the global model. The global model is just random (noisy) in the initial rounds of federation, and hence the curricula it produces are also random, thereby closely approximating the performance of the random curriculum. As the global model is refined over time, it becomes better at determining the "true" curricula, eventually converging on the s_E curve. The model trained with s_E benefits from curriculum effects from the first round and thus starts strong.

Algorithm 2. Partition Difficulty Distribution

Input: partitions $\{\mathbb{P}_0, \mathbb{P}_1, ..., \mathbb{P}_N\}$ of the input dataset $\mathbb{D}$ of C classes indexed by c, fraction of each partition to replace $f_{ord} \in (0, 1]$, expert model $\theta_{\mathscr{E}}$

Class prior of partitions and dataset:

for each partition $i = 0, 1, 2, ..., N$ **do**

$\mathbb{P}_i \leftarrow$ count(data points of class c in partition i)

Compute loss ($\mathscr{L}$) for each data point in $\mathbb{D}$ using $\theta_{\mathscr{E}}$

$\mathbb{D}_c \leftarrow$ argsort($\mathscr{L}_c$)

Reconstitute partition:

$id_c = cumsum_i(N_{i,c})$

Distribute f_{ord}

for each partition $i = 0, 1, 2, ..., N$ **do**

$\mathbb{P}_i \leftarrow \mathbb{P}_i \cup$ partition($f_{ord} * N_{i,c}$ elements of $\mathbb{D}_c$ beginning at $id_{i,c}$)

$\mathbb{D}'_c \leftarrow$ remaining elements of $\mathbb{D}_c$

Distribute remaining $(1 - f_{ord})$

for each partition $i = 0, 1, 2, ..., N$ **do**

$\mathbb{P}_i \leftarrow \mathbb{P}_i \cup$ random($(1 - f_{ord}) * N_{i,c}$ elements of $\mathbb{D}'_c$)

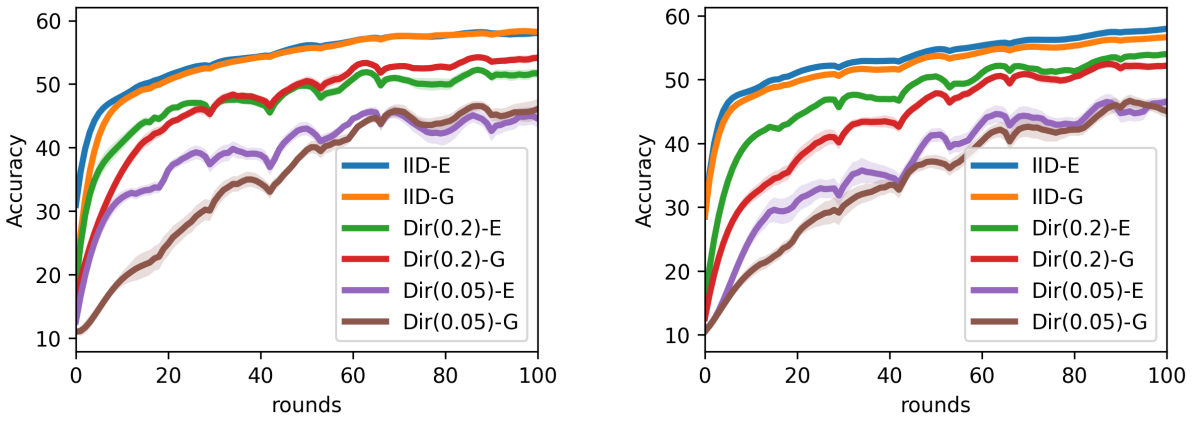
2.6.5 Ablation study

In this section, we show the interplay between the `data curriculum` and the `client curriculum` and measure their contributions towards the global model's accuracy. As reported in Fig 2.11, we observe that the client curriculum and the data curriculum independently outperform the baseline by about $2-3\%$, and we observe a synergic effect of the combination that outperforms both the curricula and the baseline by about 5%.

2.7 Theoretical Analysis and Convergence Guarantees

Now we attempt to analytically motivate the improved performance of CL in general, and for heterogeneous data in particular.

Convergence Rate Advantages of Curriculum Learning Consider a standard loss function of



((a)) Client curriculum

((b)) Data curriculum

Figure 2.10. Effect of expert scoring s_E and s_G on "curr" curriculum. Plotted here is the evolution of the global model's accuracy over the course of the federation for $\beta \in \{0.05, 0.2\}$ and IID with an ordering of 'curr', using FedAvg. s_G and s_E scoring functions have similar behavior on the Client Curricula (Left) and Data Curricula (Right).

the least squares form,

$$\mathscr{L}(\theta, \{x_i, y_i\}) = \frac{1}{2N} \sum_{i=1}^{N} (f(\theta, x_i) - y_i)^2$$

Compute the generic form of the Hessian,

$$\nabla^2_{\theta\theta} \mathscr{L}(\theta, \{x_i, y_i\}) = \frac{1}{N} \sum_{i=1}^{N} \left(\nabla_\theta f(\theta, x_i) \nabla_\theta f(\theta, x_i)^T + \nabla^2_{\theta\theta} f(\theta, x_i)(f(\theta, x_i) - y_i) \right)$$

Note that the Fisher information matrix, or Gauss-Newton term $\nabla_\theta f(\theta, x_i) \nabla_\theta f(\theta, x_i)^T$ is expected to be positive definite and independent of each samples loss value, however, the greater the magnitude of the overall loss $(f(\theta, x_i) - y_i)$ the greater the potential influence of the Hessian of the neural network model $\nabla^2_{\theta\theta} f(\theta, x_i)$ on the overall Hessian of the objective function. Thus, inherently, curriculum training makes the initial objective function more convex than otherwise. This has the clear consequence of enabling faster optimization trajectories at the beginning of the training process.

Distribution Skew and Heterogeneous Data. Formally, in terms of the optimization landscape and criteria, the presence of Non-IID data is often modeled in terms of the quantitative features of the appropriate model in a purely deterministic sense, a different minimizer, etc. Distributionally, however, one can observe, see e.g. [38], that data discrepancy across clients is often manifested

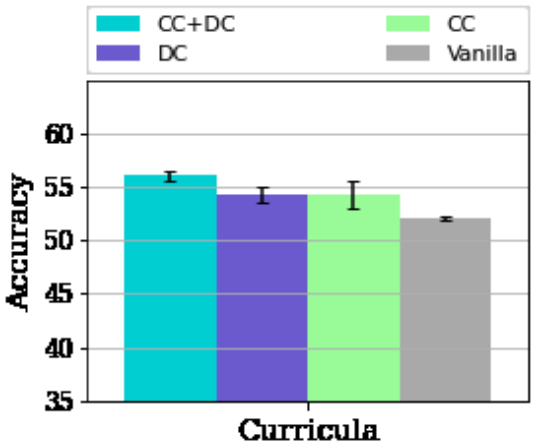


Figure 2.11. Synergic Effect of Client and Data curricula. CC here refers to Client Curriculum and DC refers to Data Curriculum. The figure shows the synergic effects of the curricula.

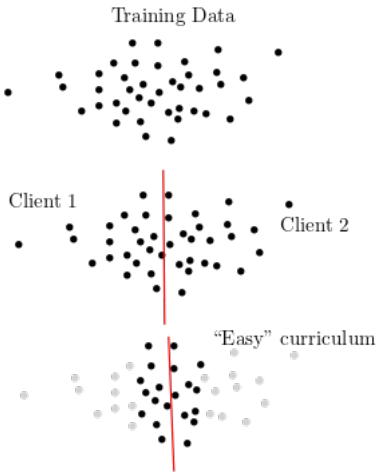


Figure 2.12. Illustration of skew-based heterogeneous data distribution across clients and curriculum learning mitigation thereof.

as *skew*. Skew is the third moment of a random variable that indicates that there is a preferential direction in the uncertainty. As an example, image data is distributed across clients by giving the left division of an image—e.g., the left face of a cat—to one client and the right to another. See Fig. 2.12 for an illustration.

A simple and transparent way to model this is to use the biased SGD framework. Specifically, there is an underlying objective function of interest $f(x)$, however, each client only has access to a biased stochastic gradient of this function. Uniquely in the case, we consider and model, the bias adds up to zero across clients. We shall use the notation of [45] although

for completeness we acknowledge the predecessor [46]. To the best of our knowledge, we present the first analysis of federated averaging with heterogeneous data using the biased SGD framework, despite how naturally it models the training procedure given standard distributional patterns in splitting training data across clients. In the SM, we develop this model formally and provide quantitative convergence results. Informally, the overall findings can be summarized as follows:

1. For strongly convex objectives, the bias introduces error to the asymptotic distance to the optimal solution, the amount of which can be decreased by appropriate annealing stepsizes according to the client or data-based CL.

2. For nonconvex objectives with a bounded gradient assumption, it appears that with a sufficiently annealed stepsize, standard centralized sublinear ergodic rates of convergence to zero approximate stationarity in expectation can be recovered.

2.7.1 Convergence Theory

In this section, we give a review of the literature on Federated Averaging and the associated convergence guarantees, presenting an analysis of how we expect these to be modified by the introduction of curriculum learning.

Curricula, Data Dissimilarity, and Convergence

The score of the data samples is based on the server's parameter vector θ_g. Naturally, this approach creates a significant association between the degree of statistical dissimilarity of the data at each client with the training difficulty score used to rank data samples for curriculum learning. So we can safely purport that CL, for non-iid data, results in a level of dissimilarity that increases with the iteration t.

To understand how this affects the convergence, we review a few standard works and study how increasing heterogeneity with the iteration number affects the convergence guarantees.

To begin with, the state of the art in convergence theory of Federated Averaging (or Local SGD) for convex objectives is given, to the best of our knowledge, in [47].

Here the main result of interest is [47, Theorem 5], for which the objective optimality gap is bounded by,

$$\mathbb{E}[f(\theta_T) - f(\theta^*)] \leq \frac{C_1}{\gamma T} + C_2\gamma\sigma^2 + C_3\gamma^2\sigma^2$$

where the variance σ is proportional to the heterogeneity. It can be seen from the convergence theory that the bound changes to, with σ_t iteration dependent,

$$\mathbb{E}[f(\theta_T) - f(x^*)] \leq \frac{C_1}{\gamma T} + C_2\gamma\sum_{t=0}^{T}\sigma_t^2 + C_3\gamma^2\sum_{t=0}^{T}\sigma_t^2$$

suggesting an overall better convergence quality for any given iteration, since we expect $\sigma_t < \sigma$ up until $t = T$, i.e., early iterations generate better accuracy than otherwise.

In regards to nonconvex objectives, which are of course more faithful to the practice of training neural networks, to the best of our knowledge the state of the art in theoretical convergence guarantees for local SGD is given in [43]. There, a notion of gradient similarity is presented,

$$\Lambda(\theta, q) = \frac{\sum_{m=1}^{M} q_m \|\nabla f_m(\theta)\|^2}{\left\| \sum_{m=1}^{M} q_m \nabla f_m(\theta) \right\|^2}$$

and assuming a bound λ on this term, λ does not appear directly in the convergence bounds in [43, Theorem 4.2 and Theorem 4.4] (respectively for the objective satisfying the PL condition and the general case). However, the number of local steps, which they denote as E, (i.e. the number of SGD steps in `ClientUpdate` in Algorithm 1) depends on $E \propto 1/\lambda$, meaning the greater the dissimilarity and the fewer local iterations are permitted to ensure convergence, a net increase in the total number of communications necessary.

The use of the FedProx objective can also be analyzed through the lens of iterate-varying

dissimilarity. Considering [48, Theorem 4] we have that with,

$$\rho_t = \frac{1}{\mu} - \bar{\rho}(B_t, \gamma, \mu), \bar{\rho}(B_t, \gamma, \mu) = O(B_t)O(\gamma)O(1/\mu)$$

and, with S_t devices chosen at iteration t

$$\mathbb{E}[f(x_{t+1}|S_t] - f(x_t) \leq -\rho_t \|\nabla f(x_t)\|^2$$

and thus with Curriculum training, we see increasing B_t and thus decreasing ρ_t, and thus, again, we shall expect to see initial faster and then gradually slower convergence.

Local SGD Model and Convergence Analysis

Consider the Local SGD framework as presented in Algorithm 3.

Algorithm 3. Local SGD Model of Algorithm 1

Input: M clients indexed by m, participating-client number Q, communication rounds T, local optimization steps J, server model f with parameters θ_g

Server executes:

initialize f with θ_g

for each round $t = 0, 1, 2, ..., T$ **do**

$\mathbb{S}_t \leftarrow$ (random set of Q clients)

for each client $q \in \mathbb{S}_t$ **in parallel do**

broadcast θ_g^t to clients as $\theta_k^{(t,0)}$

for $j = 0, 1, 2, ..., J$

Sample $g_k^{(t,j)} \sim \nabla f(\theta_k^{(t,j)}, \mathscr{D}_k)$

$\theta_k^{(t,j+1)} \leftarrow \theta_k^{(t,j)} - \alpha^{(t,j)} g_k^{(t,j)}$

$\theta_g^{(t+1)} = \sum_{k=1}^{K} \frac{|\mathscr{D}_k|}{\sum_{i=1}^{K} |\mathscr{D}_i|} \theta_k^{(t,J)}$

return θ_g^{t+1}

We formalize the notion of distributional skew by making the following assumption on the bias structure associated with each stochastic gradient computation:

Assumption 1. *It holds that* $g_k^{(t,j)}$ *satisfies,*

$$g_k^{(t,j)} = \nabla f(\theta_k^{(t,j)}) + b_k^{(t,j)}(\theta_k^{(t,j)}) + n_k^{(t,j)}(\theta_k^{(t,j)}, \xi_k^{(t,j)}) \tag{2.2}$$

where $\|b_k^{(t,j)}(\theta_k^{(t,j)})\|^2 \leq B^{(t,j)}$ *for all k, and, for all* θ*,*

$$\sum_{k \in S_t} b_k^{(t,j)}(\theta) = 0 \tag{2.3}$$

and $\xi_k^{(t,j)}$ *is a random variable satisfying,*

$$\mathbb{E}_\xi [n_k^{(t,j)}(\theta_k^{(t,j)}, \xi_k^{(t,j)})] = 0 \tag{2.4}$$

We note that,

$$\begin{aligned}
&B^{(0,0)} = B^{(0,1)} = ... = B^{(0,J)} < B^{(1,0)} = B^{(1,1)} = ... \\
&= B^{(1,J)} < ... < B^{(t,0)} = B^{(t,1)} = ... = B^{(t,J)} < B^{(t+1,0)} \\
&= B^{(t+1,1)} = ... = B^{(t+1,J)} < ... < B^{(T,0)} = B^{(T,1)} \\
&= ... = B^{(T,J)}
\end{aligned}$$

for **client based** curriculum training, and

$$\begin{aligned}
&B^{(0,0)} < B^{(0,1)} < ... = B^{(0,J)} = B^{(1,0)} < B^{(1,1)} < ... \\
&< B^{(1,J)} = B^{(2,0)} < ...B^{(t,0)} < B^{(t,1)} < ... \\
&< B^{(t,J)} = B^{(t+1,0)} < B^{(t+1,1)} < ... \\
&< B^{(t+1,J)} ... B^{(T,K-1)} < B^{(T,J)}
\end{aligned}$$

for **data based** curriculum training.

Now we present two results as depending on the conditions applying to the functions characterizing the optimization. In the first case, we shall consider strongly convex objectives, as

characterizing least squares empirical risk minimization of, e.g., linear models. In this scenario, we permit the variance to grow with the parameter size, i.e., we do not assume bounded gradients.

Theorem 1. *Assume that*

- f *is strongly convex with convexity parameter* $\mu > 0$
- ∇f *is Lipschitz continuous with Lipschitz constant* L
- *the noise variance satisfies,*

$$\mathbb{E}_{\xi}\left[\left\|n_k^{(t,j)}(\theta_k^{(t,j)},\xi_k^{(t,j)})\right\|^2\right] \leq M\left\|\nabla f(\theta_k^{(t,j)}) + b_k^{(t,j)}(\theta_k^{(t,j)})\right\|^2 + \sigma^2$$

- *For all* t, j *we have ,*

$$\alpha^{(t,j)} \leq \frac{1}{4(3+2M)L}$$

Then it holds that the distance to the solution satisfies, after each averaging step,

$$\mathbb{E}\left\|\hat{\theta}^{(T,0)} - \theta^*\right\|^2 \leq \prod_{t=1}^{T}\prod_{j=0}^{J}(1-\alpha^{(t,j)}\mu/2)\|\hat{\theta}^{(0,0)} - \theta^*\|^2 + \sum_{t=1}^{T}\sum_{j=0}^{J}\frac{2(\alpha^{(t,j)})^2[L((3+2M)B^{(t,j)}+3\sigma^3]}{Q} + \sum_{t=1}^{T}\sum_{j=0}^{J}\frac{2\alpha^{(t,j)}L(B^{(t,j)})^2/\mu)}{Q}$$

In studying the form of this result, we note that the overall convergence rate and error resembles the original with an important caveat in regards to the error on account of the bias term. First, the bias term adds an error proportional to the stepsize, thus yielding an asymptotic error bounded from below with the bias. Second, the stepsize can be used to mitigate the error from the bias terms. Indeed, with, e.g., data-based curriculum, if $B^{(t,j)} = O(j^{1/4})$ then $\alpha^{(t,j)} = O(t^{-1}j^{-1/4})$ would mitigate the growing error. It is clear that the standard practice of diminishing stepsizes will result in a lower total error at each iteration for curriculum compared to anti-curriculum. Standard Local SGD guarantees are not preserved regardless, however, with

the asymptotic bias depending on the total degree of data heterogeneity, summed in this weighted manner throughout the optimization procedure.

Nonconvex Objectives Now we consider the general case of nonconvex objectives without any additional conditions regarding the growth properties of the objective function to permit generality encompassing the functional properties of neural networks. Using the biased SGD framework and inspired by the structure of the convergence theory of [49], we study the effect of the associated gradient estimate errors.

Theorem 2. *Assume that*

- $\|\nabla f\|$ *is uniformly bounded by* G
- ∇f *is Lipschitz continuous with Lipschitz constant* L
- *the noise variance satisfies,*

$$\mathbb{E}_{\xi}\left[\left\|n_k^{(t,j)}(\theta_k^{(t,j)},\xi_k^{(t,j)})\right\|^2\right] \leq \sigma^2$$

- f *is lower bounded by* f_*

Then we obtain the ergodic rate,

$$\sum_{t=0}^{T}\sum_{j=0}^{J}\|\nabla f(\theta^{(t,0)}\|^2 \leq Q(f(\theta^0)-f^*)+2\sum_{t=0}^{T}\sum_{j=0}^{J}\alpha^{(t,j)}\left(\alpha^{(t,j)}+\sum_{l=j}^{J}\alpha^{(t,l)}\right)LG^2$$

Compared to standard results, we can see that curricula contribute an error that corresponds to the cross terms of the stepsizes, indicating a benefit to annealing the stepsize along local iterations as well as along averaging steps.

We present the proofs in Appendix B that build up the argument for the main new convergence results we present in Section 2.7.1, specifically Theorem 1 and 4.

2.8 Implementation Details

We begin by splitting the dataset into K partitions, and these partitions are distributed among the N clients in the federation. For most experiments $M = 100$ and the partitions are constructed with an input Non-IID Dirichlet distribution with parameter β and using Algorithm 2 with $f_{ord} = 0$, unless otherwise specified. The merits of the Algorithm 2 are detailed in Section 2.6.3.

At the client, we use an SGD optimizer for training with an exponentially decaying learning $\eta = \eta_0(1 + \alpha * i)^{-b}$, with parameters $\eta_0 = 0.001$, $\alpha = 0.001$, $b = 0.75$ and i is the step index, and a momentum $\rho = 0.9$ and weight decay of $\omega = 5 * 10^{-4}$. The step count i is a parameter local to the clients and is reset at the beginning of each federation round thereby resetting the learning rate back to η_0 for each round of federation. For the ResNet models however, we do not use the exponential decay learning rate and set $b = 0$ with $\eta_0 = 0.01$, and weight decay $\omega = 0$, due to our observation that these values empirically work well.

A small batch size of $bs_{data} = 10$ is used on the server. At each client, we use the local epochs $n_{epoch} = 10$, which, together with the client data partition size, determines the number of local steps at the clients between two global model averaging steps of the federation algorithm. The number of communication rounds of federation is $R = 100$ and the client participation rate is $f = 0.1$, unless otherwise specified. Similarly, when performing client curriculum, we use a client batch size of $bs_{client} = 10$.

Certain federated learning algorithms require additional algorithm specific parameters; these are chosen to match the best values reported by the authors in their respective papers. For reproducibility of the experiments, we seed our random number generator with a seed of 202207 at the beginning of each experiment. Each experiment consists of 3 trials, and we report the mean and variance of the results.

2.9 Conclusion

In this Chapter, we provided a comprehensive study on the benefit of employing CL in FL under both homogeneous and heterogeneous settings. We further ran extensive experiments on a broad range of curricula and pacing functions over three datasets, CIFAR10, CIFAR100, and FMNIST and demonstrated that ordered learning can have noticeable benefits in federated training. Surprisingly, we found empirically that CL can be more beneficial when the clients underlying data distributions are significantly Non-IID. By studying the convergence behavior of FL using a novel biased SGD model based on the observation of data heterogeneity as distributional skew, we were able to theoretically explain this phenomenon. Moreover, we proposed curriculum on clients for the first time. Our results show that the order in which clients are participated in the federation plays an important role in the accuracy performance of the global model. In particular, training the global model in a meaningful order, from the easy clients to the hard ones, using curriculum learning can provide performance improvements over the random sampling approach.

Chapter 3

Clustered Federated Learning

3.1 Introduction

Federated Learning (FL) [50] enables a set of clients to collaboratively learn a shared prediction model without sharing their local data. Some FL approaches aim to train a common global model for all clients [31, 32, 34, 33]. However, in many FL applications where there may be data heterogeneity among clients, a single relevant global model may not exist. Alternatively, personalized FL approaches have been studied. One approach is to first train a global model and then allow each client to fine-tune it via a few rounds of stochastic gradient descent (SGD) [51, 5, 52]. Another approach is for each client to jointly train a global model as well as a local model, and then interpolate them to derive a personalized model [44, 53]. In the former case, the approach often fails to derive a model that generalizes well to the local distributions of each client. In the latter case, when local distributions and the average distribution are far apart, the approach often degenerates to every client learning only on its own local data. Recently, clustered FL [54, 55, 3, 53] has been proposed to allow the grouping of clients into clusters so that clients belonging to the same cluster can share the same optimal model. Clustered FL has been shown to produce significantly better results, especially when separate groups of clients have significant differences in the distributions of their local data. This is possibly due to distinct learning tasks or the mixture of distributions of the local data considered, not necessarily limited to simpler forms of data heterogeneity such as label skews from otherwise the same dataset.

Essentially, what prior clustered FL algorithms are trying to do is group together clients with similar distributions so that clients in the same cluster can leverage each other's data to perform federated learning more effectively. Previous clustered FL algorithms attempt to *learn* these distribution similarities *indirectly* when clients learn the cluster to which they should belong as well as the cluster model during training. For example, the clustered FL approach presented in [55] alternately estimates the cluster identities of clients and optimizes the cluster model parameters via SGD.

In this chapter, we propose two *clustered federated learning* algorithm named as FLIS, and PACFL. We first delve into FLIS and then will present PACFL.

3.2 Background and Related Work

3.2.1 Global Federated Learning with Non-IID Data

The purpose of global federated learning is to train a single global model that minimizes the empirical risk function over the union of the data across all clients. FedAvg [31] simply performs parameter averaging over the client models after several local SGD updates and produces a global model. However, it has been empirically shown that under Non-IID settings, this approach is challenging due to weight divergence (client-drift) and unguaranteed convergence [56, 32, 43]. To alleviate the model drift, [56] proposed to share a small subset of data across all clients with Non-IID data. FedProx [32] incorporates a proximal term to the local training objective to keep the models close to the global model. SCAFFOLD [33] models data heterogeneity as a source of variance among clients and employs a variance reduction technique. It estimates the updated direction of the global model and that of each client. Then, the drift of local training is measured by the difference between these two update directions. Finally, SCAFFOLD corrects the local updates by adding the drift in the local training.

Another successor to FedAvg is FedNova [34], which takes the number of local training epochs of each client in each round into account to obtain an unbiased global model. It suggests

normalizing and scaling the local updates of each party according to their number of local epochs before updating the global model. While these advances may make a global model more robust under local data heterogeneity, they do not directly address local-level data distribution performance relevant to individual clients. Further, they yield poor results in practice due to the presence of Non-IID data where only a subset of all potential features are useful to each client. Therefore, in highly personalized scenarios, the target local model may be fairly different from the global aggregate [57].

3.2.2 Federated Learning from a Client's Perspective

To deal with the aforementioned challenges under Non-IID settings, other approaches aim to build a personalized model for each client by customizing the global model via local fine-tuning [53, 51, 52]. In these schemes, the global model serves as a starting point for learning a personalized model on the local data of every client. However, this global model would not serve as a good initialization if the underlying data distribution of clients is substantially different from each other. Similarly, multi-task learning aims to train personalized models for multiple related tasks in a centralized manner by learning the relationships between the tasks from data [58, 59].

3.2.3 Clustered Federated Learning

Alternatively, clustered FL approaches [54, 55, 53] aim to cluster clients with similar local data distributions to leverage federated learning per cluster more effectively. However, many rounds of the federation may be required until the formation of clusters is stabilized. Specifically, Clustered-FL (CFL) [55] dynamically forms clusters and assigns clients to those clusters based on the cosine similarities of the client updates and the centralized learning. IFCA [54] considers a pre-defined number of clusters. At each round, each active client downloads all the available models from the server and selects the best cluster based on the local test accuracy, which is very demanding in terms of communications cost. In addition, in such a setting where the number of

clusters must be fixed *a priori* regardless of the level of data heterogeneity among clients, IFCA could perform poorly for many clients. This problem can be quite pronounced under settings with highly skewed Non-IID data, which would require more clusters, or under settings with slightly skewed Non-IID data, which would require fewer clusters.

3.3 FLIS

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

- We propose the idea of *inference similarity* as a way for the central server to identify clusters of clients that have similar data distributions without requiring any access to the private data of clients. This way, clients in the same cluster can benefit from each other's training without the corruptive influence of clients with unrelated data distributions.

- Our algorithm can constitute joint and disjoint clusters and does not require the number of clusters to be known *a priori*. Further, it is effective both in Non-IID and IID regimes. In contrast, prior clustered FL approaches [54, 55] consider a pre-defined number of clusters (models) on the server and assign a hard membership ID to the clients. In such settings, these prior approaches could perform poorly under scenarios with highly skewed Non-IID data that would require a greater number of clusters, or scenarios with slightly skewed Non-IID data that would require fewer clusters, since we cannot know how many unique data distributions the client datasets are drawn from.

- Our proposed algorithm can also elegantly handle newcomer clients unseen at training

time by matching them with a cluster model that the client can further personalize with local training.

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

3.3.1 Overview of FLIS Algorithm

In this section, we provide details of our algorithm. We name this algorithm Federated Learning by Inference Similarity (FLIS). FLIS is able to form both joint dynamic clusters with soft membership ID, named FLIS (DC), and disjoint hierarchically formed clusters with hard membership ID, named as FLIS (HC). The overview of FLIS (DC) which forms joint clusters is sketched in Figure 3.1 and presented in Algorithm 4, and 5. The overview of FLIS (HC) which forms disjoint clusters is presented in Algorithm 6.

The first round of the algorithm starts with a random initial model parameters θ_g. In the t-th iteration of FLIS, the central server samples a random subset of clients $\mathscr{S}_t \subseteq [N]$ (N is the total number of clients), and broadcasts the current model parameters $\{\theta^t_{g,j_i}\}_{i=1}^T$ to the clients in $\mathscr{S}_t$. We recall that the local objective L_k is typically defined by the empirical loss over local data. Each client then estimates its cluster identity via finding the model parameter that yields minimum loss on its test data, i.e., $\theta^t_{k,j^*_t} = \text{argmin}_j \; L_k(D^{test}_k; \theta^t_{g,j_t})$. Then the clients perform $\mathscr{T}$ steps of stochastic gradient descent (SGD) updates, get the updated model, and send their model parameters, $\{\theta^{t+1}_k\}_{k=1}^{|\mathscr{S}_t|}$, to the server. After receiving the model parameters from all the participating clients, the server then leverages inference similarity method to form clusters of clients that have similar data distributions. Finally, the server collects all the parameters from clients who are in the same cluster and averages the model parameters of each cluster.

Algorithm 4. The FLIS (DC) framework

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

Init: Initialize the server model with θ_g^0

Def `FLIS_DC`**:**

  for each round $t = 0, 1, 2, \dots$ **do**

    $n \leftarrow \max(R \times N, 1)$ $\mathscr{S}_t \leftarrow \{k_1, \dots, k_n\}$ random set of n clients **for** each client $k \in \mathscr{S}_t$ **in parallel do**

      if $t = 0$ **then**

        download θ_g^0 from the server and start training, i.e. $\theta_{k,j_t^*}^t = \theta_g^0$

      else

        download clusters θ_{g,j_t}^t, $j_t = 1, \dots, T_t$ from the server and select the best cluster according to $\theta_{k,j_t^*}^t = \text{argmin } L_k(D_k^{test}; \theta_{g,j_t}^t)$

      end

      $\theta_{k,j_t^*}^{t+1} \leftarrow \text{ClientUpdate}(C_k; \theta_{k,j_t^*}^t)$ `// SGD training`

    end

    $\{C_{j_{t+1}}\}_{j_{t+1}=1}^{T_{t+1}} =$ `ISC` $(D^{Server}, \{\theta_{k,j_t^*}^{t+1}\}_{k=1,\dots,n})$ `// dynamically clustering clients via inference similarity`

    $\theta_{g,j_{t+1}}^{t+1} = \sum_{k \in C_{j_{t+1}}} |D_k| \theta_{k,j_t^*}^{t+1} / \sum_{k \in C_{j_{t+1}}} |D_k|$

  end

return

Algorithm 5. Inference Similarity Clustering (ISC)

Require: Data on the server D^{Server}, β

Return: The formed clusters $\{C_j\}$

Function `ISC` $(D^{Server}, \{\theta_{k,j_t^*}^{t+1}\}_{k=1,\dots,n})$**:**

  $B_k = F_k(D^{Server}; \theta_{k,j_t^*}^{t+1})$ `//` F_k `is client model`

  $A_{i,j} = \frac{||B_i \odot B_j||_F}{||B_i||_F ||B_j||_F}; i, j = 1, \dots, n$ `// Server constructs the adjacency matrix`

  $\tilde{A}_{i,j} = \Gamma(A_{i,j}) = \text{Sign}(A_{i,j} - \beta)$ `// Server applies hard thresholding and does joint clustering`

  Return $\{C_{j_{t+1}}\}_{j_{t+1}=1}^{T_{t+1}}$

return

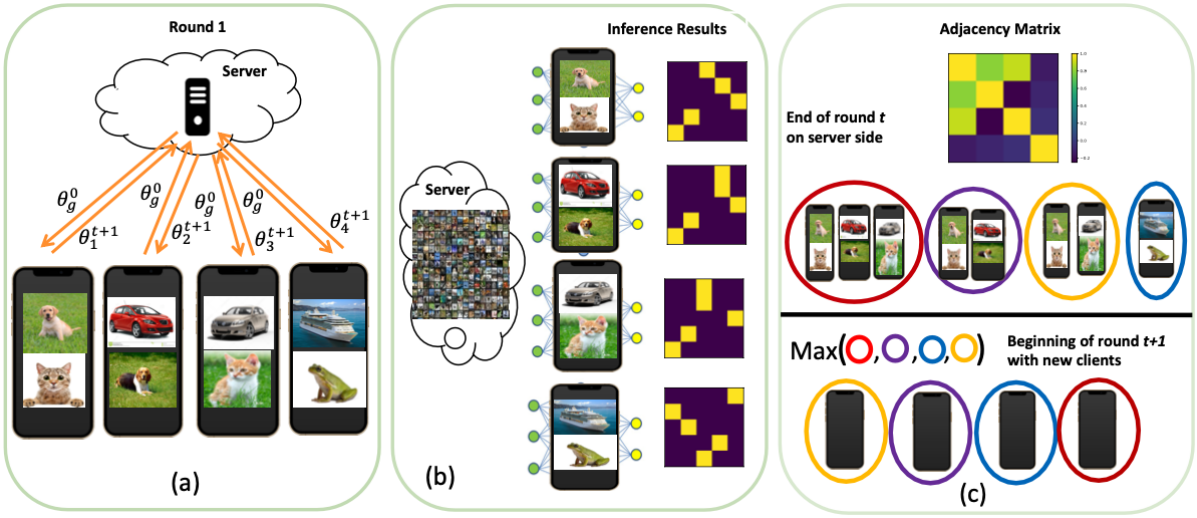


Figure 3.1. A toy example showing the overview of FLIS algorithm. (a) The server sends the initial global model to the clients at Round 1. The clients update the received model using their local data and send back their updated models to the server. (b) The server captures the inference results on its own small dataset. Then according to the similarity of the inference results, the clients are clustered. In this example, clients {1, 2, 3} are yielding more similar inference results compared to Client 4. (c) The server uses inference similarity results to constitute the adjacency matrix and identify their cluster IDs via hard thresholding or hierarchical clustering and does model averaging within each cluster. In the next round, each new client selects the best cluster out of the ones that has been formed in the previous round.

3.3.2 Clustering Clients by Inference Similarity

Herein, we are aiming to find clients with similar data distributions without requiring any prior knowledge about the data distributions. In doing so, we assume that the server has some auxiliary real or synthetic data[1] on its own[2]. The server then performs inference on each client model and obtains a $\tilde{M} \times \tilde{N}$ matrix, $B_k = F_k(D^{server}; \theta^t_{k,j^*_t})$, $k = 1, ..., |\mathscr{S}_t|$, where $\tilde{N}$ and $\tilde{M}$ are the numbers of final neurons of the last fully connected layer (classification layer), and the number of auxiliary samples at the server, respectively. Note that, the columns of B_k can be one-hot or soft labels. Using the inference results of each client, B_k, the server constructs an adjacency matrix as $A_{i,j} = \frac{||B_i \odot B_j||_F}{||B_i||_F ||B_j||_F}$, where $i, j = 1, ..., |\mathscr{S}_t|$, and $\odot$ stands for Hadamard product. Having the adjacency matrix $A_{i,j}$, as mentioned earlier, depending on whether forming joint clusters are

[1]Further, we find it noteworthy to mention that along the years this assumption for the FL scenarios has been adopted in the literature [60, 61, 56, 62, 63, 64, 65, 66, 67].

[2]The number of auxiliary samples used for forming the clusters at the server is 2500.

Algorithm 6. The FLIS (HC) framework

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

Init: Initialize the server model with θ_g^0

Def `FLIS_HC`**:**

  for each round $t = 0,1,2,\ldots$ **do**

    if $t = 0$ **then**

      All clients receive the initial server model θ_g^0, perform local update and send back the updated models to the server. $\mathbf{A} \leftarrow$ server forms $\mathbf{A}$ based on $A_{i,j}$ defined in Subsection 3.3.2.

      $\{C_1,...,C_j\} = \text{HC}(\mathbf{A},\beta)$ `// performing hierarchical clustering to obtain the clusters`

      $\theta_{g,j}^0 \leftarrow \theta_g^0$ `// initializing all clusters with` θ_g^0

    else

      $n \leftarrow \max(R \times N, 1)$ $\mathscr{S}_t \leftarrow \{k_1,\ldots,k_n\}$ random set of n clients

    end

    for each client $k \in \mathscr{S}_t$ **in parallel do**

      Each client k receives its cluster model from the server θ_{g,j_k}^t, $j = 1,\ldots,T$

      $\theta_{k,j_k}^{t+1} \leftarrow \text{ClientUpdate}(C_k;\theta_{k,j_k}^t)$ `// SGD training`

    end

    $\theta_{g,j}^{t+1} = \sum_{k\in C_j} |D_k| \theta_{k,j_k}^{t+1} / \sum_{k\in C_j} |D_k|$

  end

return

of interest or disjoint ones, we propose two different clustering approaches. For FLIS (DC) that constructing joint clusters on the server is of interest, we define a hard thresholding operator Γ which is applied on $A_{i,j}$ and yields $\tilde{A}_{i,j} = \Gamma(A_{i,j}) = \text{Sign}(A_{i,j} - \beta)$, with β being a threshold value. Now, making use of $\tilde{A}_{i,j}$, the server can form joint clusters of interest by putting indices of the positive entries in each row of $\tilde{A}_{i,j}$ in the same cluster as is shown in the toy example in Fig 3.1. In FLIS (DC), in each round, 10 clusters are formed, which is equal to the number of participant clients in each round. For FLIS (HC), having $\tilde{A}_{i,j}$ in hand, the server can group the clients by employing hierarchical clustering (HC) [68] as presented in Algorithm 6. It is noteworthy that in FLIS (HC) the number of formed clusters are fixed and depend upon the clustering threshold (β) of HC which is a hyperparameter.

3.3.3 Experiments

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

Experimental Settings

Datasets and Models. We conduct experiments on CIFAR-10, CIFAR-100, SVHN, and Fashion MNIST (FMNIST) datasets. For each dataset we considered three different federated heterogeneity settings as in [37]: Non-IID label skew (20%), Non-IID label skew (30%), and Non-IID Dir(α). For Non-IID label skew (20%) and (30%) we first randomly assigned 20% and 30% of the total classes to each client, respectively. Then, each class is randomly and equally partitioned and distributed amongst the clients who own that particular class. For Non-IID Dir(α), we get random samples for class c from Dirichlet distribution according to $p_c \sim \text{Dir}(\alpha)$ and give each client j random samples of class c according to $p_{c,j}$ proportion. In this setup, heterogeneity can be controlled by the parameter α of Dirichlet distribution. Specifically, when $\alpha \to \infty$ the clients' data partitions become IID, and when $\alpha \to 0$ they become extremely Non-IID. We used Lenet-5 architecture for CIFAR-10, SVHN, and FMNIST datasets, and ResNet-9 architecture for CIFAR-100 dataset.

Baselines. To show the effectiveness of the proposed method, we compare the results of our algorithm against SOTA personalized FL methods as well as methods targeting learning a single global model. Among the personalized SOTA ones, LG-FedAvg [69], learns local representations and a global head. Per-FedAvg [51] uses meta-learning technique to learn a single model that can adapt to the local dataset by a few steps of SGD. IFCA [54] groups the users into a predefined number of clusters and alternately optimizes the clusters and model parameters via gradient descent. Clustered-FL [55] dynamically groups the clients into clusters based on the cosine

Table 3.1. Test accuracy comparison across different datasets for Non-IID label skew (20%), and (30%).

Algorithm	FMNIST	CIFAR-10	CIFAR-100	SVHN
Non-IID label skew (20%)				
SOLO	95.92 ± 0.57	79.22 ± 1.67	32.28 ± 0.23	79.72 ± 1.37
FedAvg	77.3 ± 4.9	49.8 ± 3.3	53.73 ± 0.50	80.2 ± 0.8
FedProx	74.9 ± 2.6	50.7 ± 1.7	54.35 ± 0.84	79.3 ± 0.9
FedNova	70.4 ± 5.1	46.5 ± 3.5	53.61 ± 0.42	75.4 ± 4.8
Scafold	42.8 ± 28.7	49.1 ± 1.7	54.15 ± 0.42	62.7 ± 11.6
LG	96.80 ± 0.51	86.31 ± 0.82	45.98 ± 0.34	92.61 ± 0.45
PerFedAvg	95.95 ± 1.15	85.46 ± 0.56	60.19 ± 0.15	93.32 ± 2.05
IFCA	97.15 ± 0.01	87.99 ± 0.15	71.84 ± 0.23	95.42 ± 0.06
CFL	77.93 ± 2.19	51.11 ± 1.01	40.29 ± 2.23	73.62 ± 1.76
FLIS (DC)	$\mathbf{97.64 \pm 0.38}$	$\mathbf{89.47 \pm 0.92}$	$\mathbf{73.91 \pm 0.29}$	$\mathbf{95.65 \pm 0.17}$
FLIS (HC)	$\mathbf{97.45 \pm 0.08}$	$\mathbf{89.35 \pm 0.46}$	$\mathbf{73.20 \pm 0.31}$	$\mathbf{95.48 \pm 0.21}$
Non-IID label skew (30%)				
SOLO	93.93 ± 0.10	65 ± 0.65	22.95 ± 0.81	68.70 ± 3.13
FedAvg	80.7 ± 1.9	58.3 ± 1.2	54.73 ± 0.41	82.0 ± 0.7
FedProx	82.5 ± 1.9	57.1 ± 1.2	53.31 ± 0.48	82.1 ± 1.0
FedNova	78.9 ± 3.0	54.4 ± 1.1	54.62 ± 0.91	80.5 ± 1.2
Scafold	77.7 ± 3.8	57.8 ± 1.4	54.90 ± 0.42	77.2 ± 2.0
LG	94.21 ± 0.40	76.58 ± 0.16	35.91 ± 0.20	87.69 ± 0.77
PerFedAvg	92.87 ± 2.67	77.67 ± 0.19	56.42 ± 0.41	91.25 ± 1.47
IFCA	95.22 ± 0.03	80.95 ± 0.29	67.39 ± 0.27	93.02 ± 0.15
CFL	78.44 ± 0.23	52.57 ± 3.09	35.23 ± 2.72	73.97 ± 4.77
FLIS (DC)	$\mathbf{95.95 \pm 0.51}$	$\mathbf{82.25 \pm 1.12}$	$\mathbf{68.36 \pm 0.12}$	$\mathbf{93.08 \pm 0.22}$
FLIS (HC)	$\mathbf{95.35 \pm 0.16}$	$\mathbf{82.17 \pm 0.22}$	$\mathbf{67.51 \pm 0.23}$	$\mathbf{93.10 \pm 0.20}$

similarities of their trainable parameters and centralized learning is performed on a per-class basis. For global FL methods, we compare versus SOTA benchmarks including FedAvg [31], FedProx [32], FedNova [34], and SCAFFOLD [33]. We also compare our results with another baseline named SOLO, where each client trains a model on its own local data without taking part in FL.

Performance Comparison. Table 3.1, and 3.2, show the average final top-1 test accuracy of all clients for all the SOTA algorithms under Non-IID label skew (20%, 30%), and Non-IID Dir(0.1) settings, respectively. In these tables, we report the results of the two proposed clustering approaches i.e., FLIS (DC) (presented in Algorithm 4) as well as FLIS (HC) (presented in Algorithm 6). Under Non-IID settings, SOLO with zero communications cost demonstrates

much better accuracy than all the global FL baselines including FedAvg, Fedprox, FedNova, and SCAFFOLD. On the other hand, each client itself may not have enough data and thus we need to better exploit the similarity among the users by clustering. This further explains the benefits of personalization and clustering in Non-IID settings. Comparing different FL approaches, we can see that FLIS (DC) consistently yields the best accuracy results among all tasks. It can outperform FedAvg by up to $\sim 40\%$.

The accuracy of FedProx is very close to FedAvg. This shows that the regularizer term in FedProx is not that beneficial in the federated training for small values of the regularizer's coefficient. For large values of the regularizer's coefficient, the convergence of FedProx is very slow and the accuracy results are poor. This is consistent with the observations in [70]. Unlike the paper introducing LG-FedAvg, in which the initialization of the algorithm is the solution of many rounds of FedAvg, for the sake of fair comparison, we initialized the model randomly and reported the average final local test accuracy over the entire federated training process rather than the average of the maximum local test accuracies for each client.

It is apparent from table 3.2 for Non-IID Dir(0.1) that LG-FedAvg and Per-FedAvg perform even worse than FedAvg. The performance of the CFL benchmark is close to that of FedAvg in most cases and even worse. IFCA (with two clusters, C=2) obtained the closest results to FLIS, but FLIS consistently beats IFCA especially in Non-IID Dir(0.1) by a large margin. FLIS shows superior learning performance over the SOTA on more challenging tasks. For instance, FLIS is noticeably better than IFCA for CIFAR-10 which is a harder task compared to FMNIST and SVHN by up to $\sim 10\%$ in Non-IID Dir(0.1). As a final note, we also studied the impact of constructing disjoint clusters. HC by extracting disjoint clusters seems to be slightly deteriorating the performance of FLIS, even though it still beats all SOTA baselines.

Communication Efficiency

What is the Required Communication Cost/Round to Reach a Target Test Accuracy? We additionally compare the SOTA baselines in terms of the number of communication rounds/cost

Table 3.2. Test accuracy comparison for Non-IID Dir(0.1).

Algorithm	FMNIST	CIFAR-10	CIFAR-100
SOLO	69.71 ± 0.99	41.68 ± 2.84	16.83 ± 0.51
FedAvg	82.91 ± 0.83	38.22 ± 3.28	44.52 ± 0.42
FedProx	84.04 ± 0.53	42.29 ± 0.95	45.52 ± 0.72
FedNova	84.50 ± 0.66	40.25 ± 1.46	46.52 ± 1.34
Scafold	10.0 ± 0.0	10.0 ± 0.0	43.73 ± 0.89
LG	74.96 ± 1.41	49.65 ± 0.37	23.59 ± 0.26
PerFedAvg	80.29 ± 2.00	53.58 ± 1.57	33.94 ± 0.41
IFCA	85.01 ± 0.30	51.16 ± 0.49	47.67 ± 0.28
CFL	74.13 ± 0.94	42.30 ± 0.25	31.42 ± 1.50
FLIS (DC)	$\mathbf{86.5 \pm 0.76}$	$\mathbf{60.33 \pm 2.30}$	$\mathbf{53.85 \pm 0.56}$
FLIS (HC)	$\mathbf{85.21 \pm 0.18}$	$\mathbf{51.18 \pm 0.21}$	$\mathbf{49.10 \pm 0.19}$

that is required to reach a specific target accuracy. Table 3.3 report the required number of communication rounds to reach the designated target test accuracies for Non-IID label skew (20%). Further, Table 3.4, and Table 3.5 report the required communication cost to reach the designated target test accuracies for Non-IID label skew (30%), and Non-IID Dir (0.1), respectively. As is observed from Table 3.3, in all scenarios, FLIS has the minimum number of communication rounds. For instance, 37 number of communication rounds is sufficient for FLIS to achieve the target accuracy of 50% for Non-IID label skew (20%) in CIFAR-100, whereas some other baselines, e.g. Per-FedAvg requires $\sim 3\times$ more communication rounds and global model FL baselines are the most expensive ones in general. In addition, as is evident from Table 3.4, and Table 3.5, FLIS outperforms all SOTA algorithms in terms of communication cost by a noticeable margin except for LG.

We attribute this to the fact that by grouping the clients with similar data distributions in the same clusters, the clusters tend to mimic the IID setting, which means faster convergence in fewer communication rounds. Note that "$--$" means the baseline was not able to reach the target accuracy. These characteristics of FLIS (HC) are desirable in practice as it helps to reduce the communication overhead in FL systems in two ways: first, it converges fast, and second, rather than communicating all clusters (models) with the clients, the server will receive

Table 3.3. Comparing different FL approaches for Non-IID (20%) in terms of the required number of communication rounds to reach the target top-1 average local test accuracy. We evaluate on FMNIST, CIFAR-10, CIFAR-100, and SVHN datasets.

Algorithm	FMNIST	CIFAR-10	CIFAR-100	SVHN
Target	75%	80%	50%	75%
FedAvg	200	−−	130	150
FedProx	200	−−	115	200
FedNova	−−	−−	120	150
Scafold	−−	−−	82	−−
LG	13	33	−−	16
PerFedAvg	19	60	110	39
IFCA	14	25	40	17
CFL	47	−−	−−	−−
FLIS (HC)	**12**	**24**	**37**	**15**

Table 3.4. Comparing different FL approaches for Non-IID label skew (30%) in terms of the required communication cost in **Mb** to reach the target top-1 average local test accuracy. We evaluate on FMNIST, CIFAR-10, CIFAR-100, and SVHN.

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

the cluster ID from each client and then only send the corresponding cluster to each client.

Learning with Limited Communication

We further consider circumstances that frequently happen in practice, where a limited budget of communication rounds is allowed for federation under a heterogeneous setting. Herein, we compare the performance of FLIS with the rest of SOTA. We allocate a limited communication rounds budget of 80 for all personalized baselines and report the average final test accuracy over all clients versus the number of communication rounds for Non-IID label skew (20%), and (30%)

Table 3.5. Comparing different FL approaches for Non-IID Dir (0.1) in terms of the required communication cost in **Mb** to reach target top-1 average local test accuracy. We evaluate on FMNIST, CIFAR-10, and CIFAR-100.

Algorithm	FMNIST	CIFAR-10	CIFAR-100
Target	75%	45%	40%
FedAvg	9.92	−−	3389.47
FedProx	9.92	−−	2965.52
FedNova	9.92	−−	3178.03
Scafold	−−	−−	5402.44
LG	14.09	**2.46**	−−
PerFedAvg	31.74	18.25	−−
IFCA	13.09	13.09	4575.89
CFL	79.36	−−	−−
FLIS (HC)	**8.73**	8.73	**2372.84**

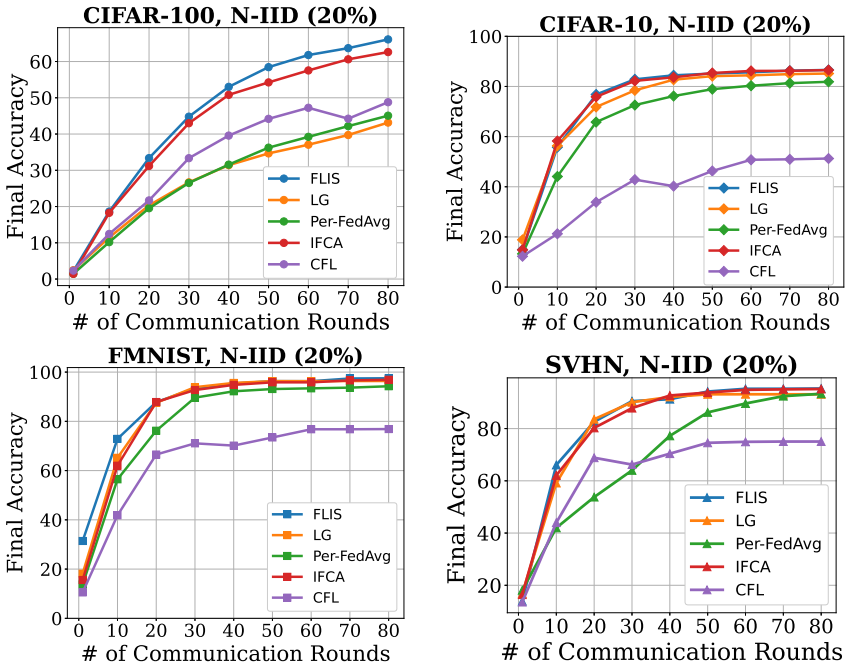


Figure 3.2. Test accuracy versus the number of communication rounds for Non-IID (20%). FLIS converges fast to the desired accuracy and consistently outperforms strong competitors.

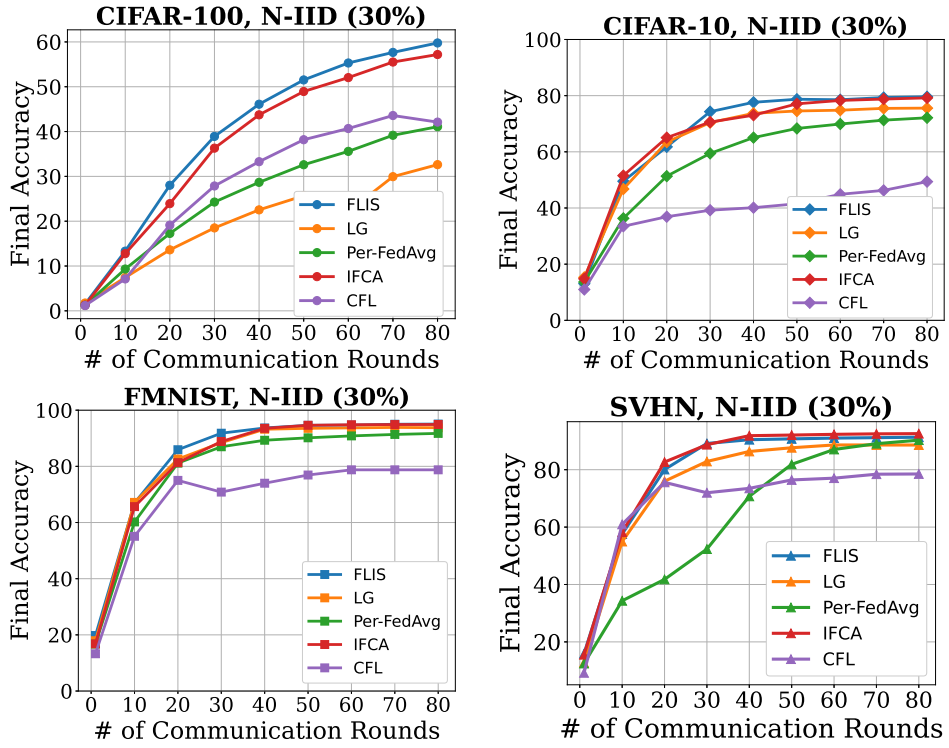


Figure 3.3. Test accuracy versus the number of communication rounds for Non-IID (30%). FLIS converges fast to the desired accuracy and consistently outperforms strong competitors, except in SVHN.

in Fig. 3.2, and Fig. 3.3, respectively. We can see that our proposed method requires only 30 communication rounds to converge in CIFAR-10, SVHN, and FMNIST datasets. CFL yields the worst performance on all benchmarks across all datasets, except for CIFAR-100. Per-Fedavg seems to benefit more from higher communication rounds. IFCA and LG are the closest lines to ours for CIFAR-10, SVHN and FMNIST. FLIS consistently outperforms the SOTA in different communication rounds.

Generalization to unseen clients

As mentioned earlier, FLIS allows new clients arriving after the distributed training to learn their personalized models. In our framework, the unseen client simply provides its model which is trained for some epochs on its own data to the server. The server then obtains the inference results of the unseen client model and extends the adjacency matrix and applies HC. Therefore, the server via inference similarity analysis identifies which existing cluster model

would be most suitable, or the server can inform the client that it should train on its own local data to form a new cluster if the client's data distribution is not sufficiently similar to the distributions of the existing clusters. It is not clear how the other personalized FL algorithms should be extended to handle unseen clients during federation. In order to evaluate the performance of new clients' personalized models, we run an experiment where only 80% of the clients participate to the training. The remaining 20% join the network at the end of the federation and receive the model from the server and personalize it for only 5 epochs. The average local test accuracy of the new clients is reported in Table 3.6. Table 3.6 demonstrates that FLIS allows unseen clients during the training to learn their personalized model with high test accuracy.

Table 3.6. Average local test accuracy across unseen clients on different datasets for Non-IID label skew (20%).

Algorithm	FMNIST	CIFAR-10	CIFAR-100	SVHN
SOLO	95.13 ± 0.42	82.30 ± 1.00	27.26 ± 0.98	91.5 ± 0.64
FedAvg	77.61 ± 3.78	31.01 ± 1.83	32.19 ± 0.32	71.78 ± 3.43
FedProx	74.30 ± 4.70	27.56 ± 3.24	32.41 ± 1.17	74.30 ± 4.70
FedNova	74.66 ± 2.81	31.48 ± 1.49	33.18 ± 0.80	73.04 ± 3.65
Scafold	73.97 ± 1.68	37.22 ± 1.34	23.90 ± 2.61	64.96 ± 4.74
LG	94.58 ± 0.33	77.98 ± 1.61	10.63 ± 0.21	89.48 ± 0.65
PerFedAvg	89.88 ± 0.38	73.79 ± 0.51	30.09 ± 0.35	67.48 ± 2.88
IFCA	96.29 ± 0.04	84.98 ± 0.41	55.66 ± 0.20	94.83 ± 0.14
FLIS (DC)	$\mathbf{97.51 \pm 1.30}$	84.45 ± 1.76	$\mathbf{59.38 \pm 1.46}$	$\mathbf{94.87 \pm 0.34}$
FLIS (HC)	$\mathbf{96.30 \pm 0.20}$	$\mathbf{85.34 \pm 0.11}$	$\mathbf{59.11 \pm 0.52}$	$\mathbf{95.11 \pm 0.18}$

Impact of important Hyper-parameters

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

The influence of the clustering threshold β. We investigate the effect of the clustering threshold β on the final test accuracy. Fig. 3.4 and Fig. 3.5 visualize the accuracy performance behavior of FLIS under different values of β, as well as the local epochs for several datasets for Non-IID (20%) and Non-IID (30%), respectively. We vary β from 0 to 1. The parameter β controls the similarity of the data distribution of clients within a cluster. *Therefore, β achieves a trade-off*

between a purely local and global model and provides a trade-off between generalization and distribution heterogeneity. To delineate, when $\beta = 0$, FLIS groups all the clients into 1 cluster and the scenario reduces to FedAvg baseline. This is the reason for the significant accuracy drop at $\beta = 0$ as it is also evident from these figures, by increasing β, FLIS becomes more strict in grouping the clients. It means FLIS only groups the clients with more amount of label/feature overlap into a cluster leading to a more personalized FL. The optimal performance on CIFAR-10, SVHN, and FMNIST for Non-IID (20%) are achieved at $\beta = 0.3, 0.3, 0.5$, respectively. Finally, when β is 1, the scenario degenerates to the SOLO baseline where each client receives the model from the server and trains it on its own local data without averaging with anyone. According to our experiments, on all datasets, all clients benefit from some level of globalization. In general, *finding the optimal trade-off between globalization and distribution heterogeneity depends on the level of heterogeneity of tasks, the intra-class distance of the dataset, as well as the data partitioning across the clients*. This is precisely what our novel FLIS (HC) is designed to do, to easily find this trade-off via the proximity matrix at the server.

Benefit of more local updates. The benefits of FLIS can be further pronounced by increasing the number of local epochs. The results are shown in Fig. 3.4, and Fig. 3.5. When the number of the local epoch is 1, the clients' local updates are very small. Therefore, the training will be slow and the accuracy becomes lower compared to the bigger number of local epochs given a fixed number of communication rounds. Also, when the clients have not been trained enough, their inference results on the server side would be erroneous which further causes less accurate clustering. For Non-IID (30%) as shown in Fig. 3.5 the results when the number of the local epochs are 10, and 20 are comparable but still are better that that of 1 local epoch. These figures show the performance of FLIS is coupled with local training epochs, especially on more challenging tasks. In contrast, it was shown in [37] when the number of local epochs is too large, the accuracy of all non-personalized models drops which is due to severe-side averaged models drift from the clients' local models [34].

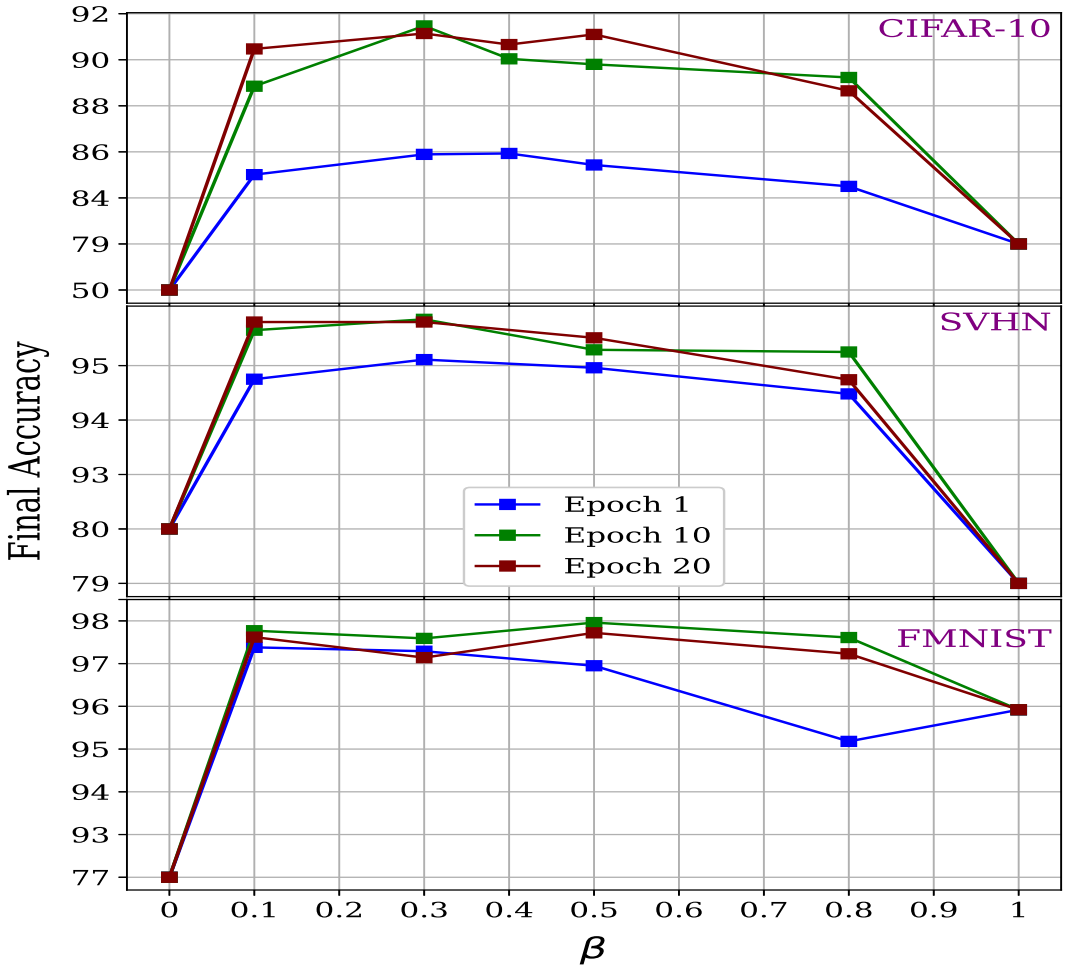


Figure 3.4. Evaluating FLIS (DC)'s accuracy performance versus the clustering threshold β, and the number of local epochs for Non-IID label skew (20%) on CIFAR-10, FMNIST, and SVHN datasets. FLIS (DC) benefits from larger numbers of local training epochs. The optimal values of β are different for different datasets.

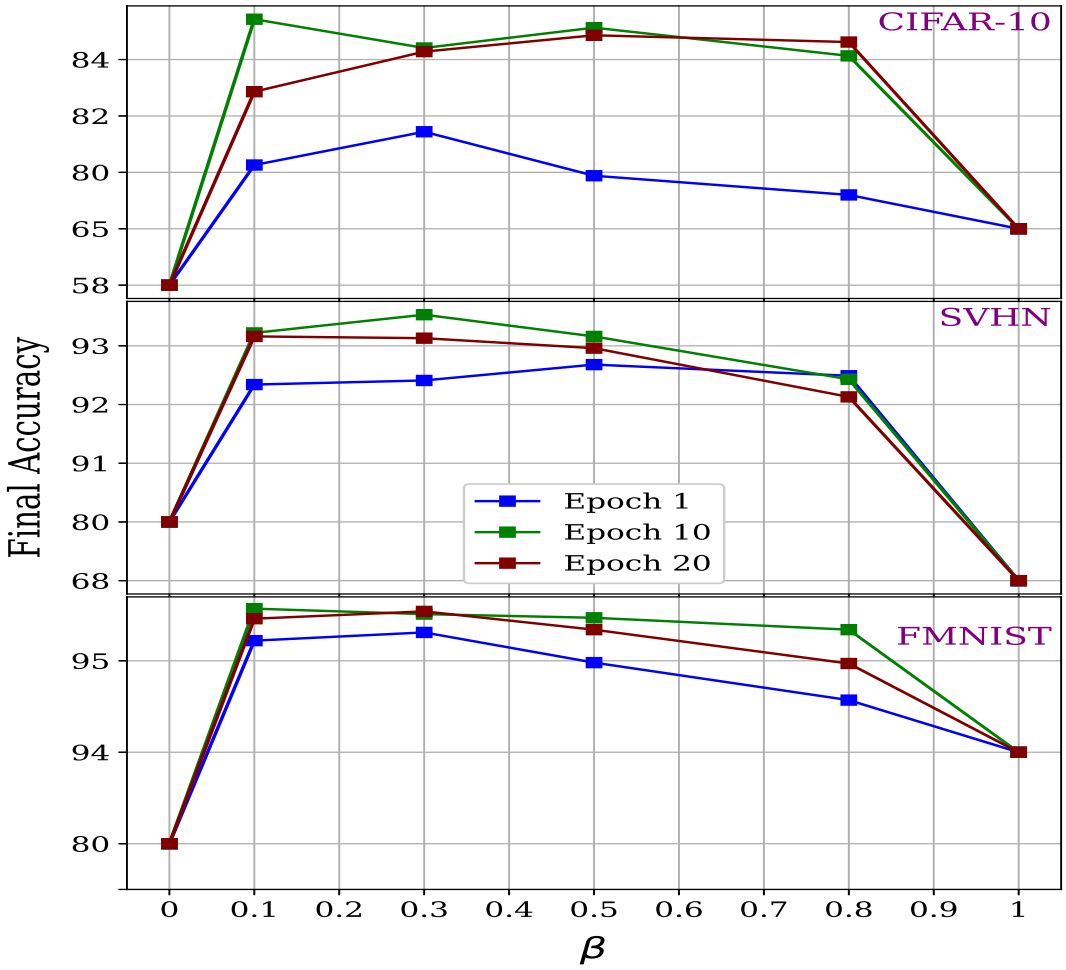


Figure 3.5. Evaluating FLIS (DC)'s accuracy performance versus the clustering threshold β, and the number of local epochs for Non-IID label skew (30%) on CIFAR-10, FMNIST, and SVHN datasets. FLIS (DC) benefits from larger numbers of local training epochs. The optimal values of β are different for different datasets.

The influence of β on clustering error

We investigate the effect of the clustering threshold β on the clustering error. Fig. 3.6, and Fig. 3.7 visualize the clustering error behavior of FLIS versus clustering threshold β and the number of local epochs for Non-IID (20%), and Non-IID (30%) on CIFAR-10, SVHN, and FMNIST datasets, respectively. We define clustering error as the summation of false positives (FP) and false negatives (FN) w.r.t. the ground truth. Depending on the dataset, at some optimal β we should expect minimum clustering error. For Non-IID (20%) on CIFAR-10, SVHN,

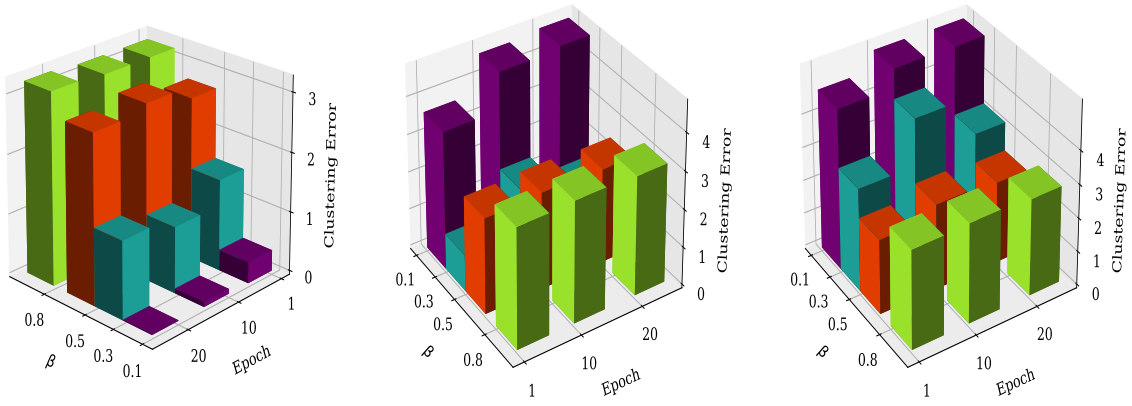


Figure 3.6. Evaluating the clustering error behavior of FLIS (DC) for Non-IID (20%) versus the clustering threshold β, and number of local epoch on **Left:** CIFAR-10, **Middle:** SVHN, and **Right:** FMNIST datasets.

and FMNIST the minimum clustering error occurs at $\beta = 0.1$, $\beta = 0.3$, and $\beta = 0.5$ which is reflected in a shorter error bar. When β is less than the optimal one, a large FP causes a bigger error and when β is bigger than the optimal value, a large FN is a reason for a bigger error. Another noticeable observation is that the accuracy peak of SVHN and FMNIST in Fig. 3.4 and their corresponding minimum clustering error in Fig. 3.6 occur at the same β. While this is not the case for CIFAR-10. Indeed, it is not required that these β values match. This can be explained by the fact that FLIS groups the clients based on the inference similarity/response of neural networks. This means FLIS selects a subset of the most similar clients out of the set of similar clients. This way, FLIS sacrifices some clustering errors by accepting more FN in order to improve accuracy. Similar discussions can be made for Fig. 3.7.

The impact of various sets of auxiliary data on the inference similarity measure

To better understand the inference behavior of clients which is leveraged by the server to group the clients with similar distributions in a Non-IID setting, we did some ablation studies. It is worth noting that, according to our experiments we do not necessarily need an auxiliary dataset on the server side that is drawn from the same dataset used to train or test the clients, nor do we need an auxiliary dataset on the server side that is representative of all the clients involved. These

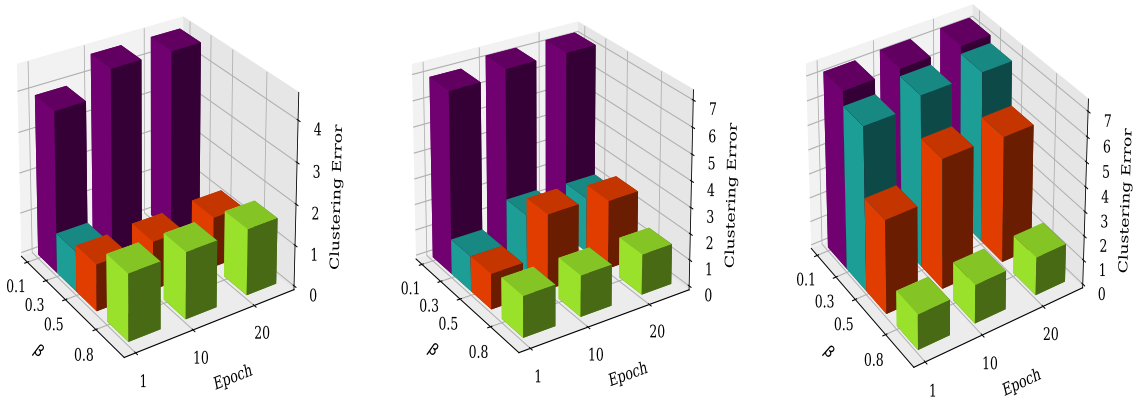
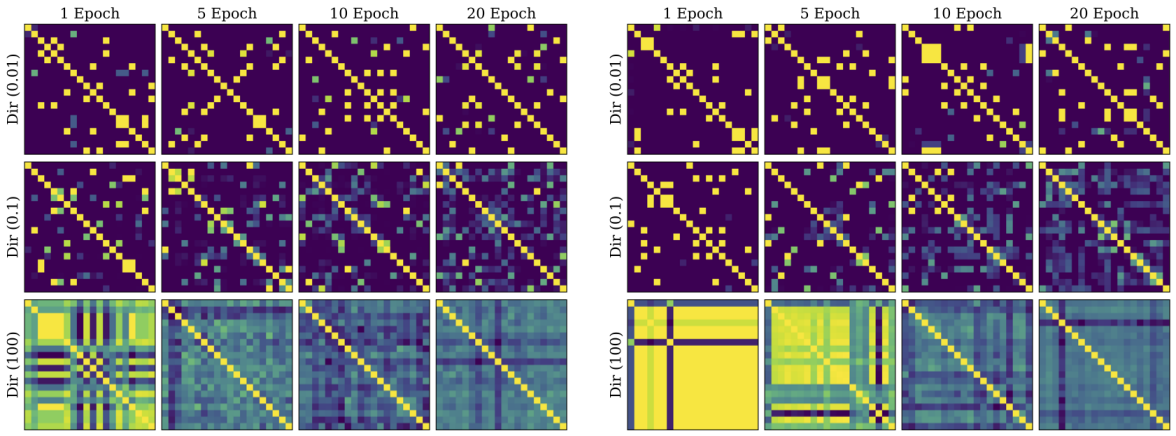


Figure 3.7. Evaluating the clustering error behavior of FLIS (DC) for Non-IID (30%) versus the clustering threshold β, and the number of local epoch on **Left:** CIFAR-10, **Middle:** SVHN, and **Right:** FMNIST datasets.

observations, while not intuitive at first, are not surprising given the well-established results in the pseudo-labeling and unsupervised domain adaptation literature [71, 72, 73, 74, 75, 76] that has already shown the effective use of labeled data from another source domain to evaluate neural nets intended for a different target domain. For example, in our experiments with CIFAR-10 as the target domain, we can just use samples drawn from another source domain like SVHN as the auxiliary dataset to perform inference similarity analysis on the server side, and the results are just as effective as using samples drawn from the same CIFAR-10 dataset. We further conducted experiments just using randomly generated images as the auxiliary dataset, and the results are also just as effective. We observe the same phenomenon when training for SVHN, but using another domain like CIFAR-10 or randomly generated images as the auxiliary dataset, where the results achieved are similar.

Fig. 3.8, 3.9, and 3.10 sketch the ablation study of inference results of clients with similar distribution (label overlap) and dissimilar distribution (without label overlap) for different numbers of local epochs, and α (the Dirichlet distribution parameter). Herein, every client data is drawn independently with class labels following a categorical distribution over P classes parameterized by a vector $\mathbf{v}(\mathbf{v}_i \leq 0; i \in [1,P]; ||\mathbf{v}||_1 = 1)$. We use α to control the Non-IID

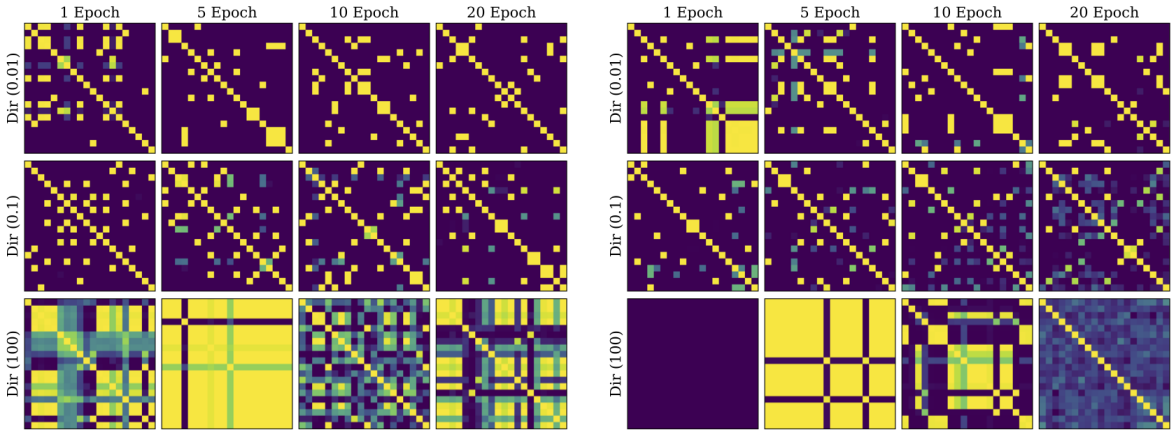
degree of the local data. The smaller α, the larger heterogeneity between local data distributions of clients. Big values of α, e.g., $\alpha = 100$ mimic identical local data distributions (IID). We visualize the inference similarity result of the clients on CIFAR-10, SVHN, and FMNIST, as well as on some synthetic randomly generated data. We sketch the adjacency matrix for the extreme cases of IID-ness and Non-IID-ness with $\alpha = \{100, 0.1, 0.01\}$. In the figures, we move from extremely Non-IID with $\alpha = 0.01$ to extremely IID with $\alpha = 100$. In all cases, the entries on the main diagonal are all 1. In the extremely Non-IID case, there should be more difference between diagonal and off-diagonal entries which is reflected in the figures. As expected, as we move to an extremely IID case, the difference between the diagonal and off-diagonal should decrease. That is because the clients have similar distributions and should produce similar inference results. Therefore, inference similarity can be a metric to measure the IID-ness and Non-IID-ness of the distributions across clients by looking at the difference between diagonal and off-diagonal entries of the adjacency matrix.



Clients trained on CIFAR-10 Clients trained on SVHN

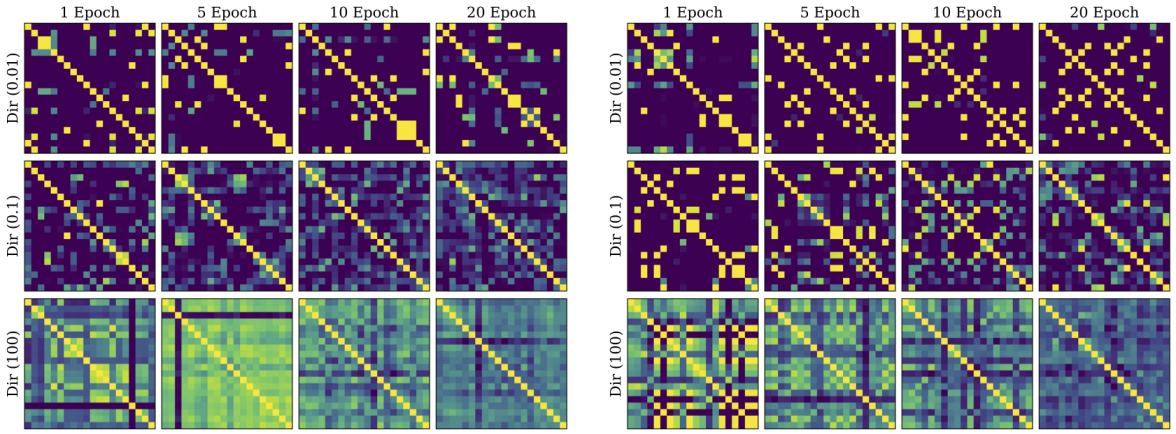
Figure 3.8. Understanding the impact of similarity and heterogeneity of the clients' data on their inference result at the server side for different degrees of Non-IIDness controlled by α, and the number of local epochs per communication round on CIFAR-10 (Left) and on SVHN (Right). 20 clients out of 100 are randomly selected and trained for a certain number of epochs (1, 5,10, and 20). The figures visualize the adjacency matrices obtained based on the inference results of some auxiliary data sampled from CIFAR-10 (Left) and that of some auxiliary data sampled from SVHN (Right) as outlined in line 3 of Algorithm 2.

For producing these figures, 20 clients out of 100 are randomly selected and trained for a



Clients trained on CIFAR-10 Clients trained on SVHN

Figure 3.9. Understanding the impact of similarity and heterogeneity of the clients' data on their inference result at the server side for different degrees of Non-IIDness controlled by α, and the number of local epochs per communication round on CIFAR-10 (Left) and on SVHN (Right). 20 clients out of 100 are randomly selected and trained for a certain number of epochs (1, 5,10, and 20). The figure visualizes the adjacency matrices obtained based on the inference results of some auxiliary data sampled from synthetic randomly generated data (vectors)
as outlined in line 3 of Algorithm 2.



Clients trained on FMNIST Clients trained on FMNIST

Figure 3.10. Understanding the impact of similarity and heterogeneity of the clients' data on their inference result at the server side for different degrees of Non-IIDness controlled by α, and the number of local epochs per communication round on FMNIST (Left) and on FMNIST (Right). 20 clients out of 100 are randomly selected and trained for a certain number of epochs (1, 5,10, and 20). The figure visualizes the adjacency matrix that is obtained based on the inference results of some auxiliary data sampled from FMNIST (Left) and that of some synthetic randomly generated data (vectors)
(Right) as outlined in line 3 of Algorithm 2.

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

3.4 PACFL

Unfortunately, the prior clustered FL approaches have the following *challenges* which in turn limits their applicability in real-world problems. 1) Since previous clustered FL training algorithms start with randomly initialized cluster models that are inherently noisy, the overall training process can be quite time-consuming as many rounds of federated learning may be required until the formation of clusters is stabilized. 2) Approaches like IFCA [54] assume a pre-defined number of clusters, but require the number of clusters to be fixed *a priori*, regardless of the differences in the actual data distributions or learning tasks among the clients, could lead to poor performance for many clients. 3) In each iteration, all cluster models have to be downloaded by the active clients in that round, which can be very costly in communications. 4) Both of the approaches i.e., those that train a common global model for all clients and personalized approaches including IFCA lack the flexibility to trade off between *personalization* and *globalization*. The above-mentioned drawbacks of the prior works, naturally lead to the following important question. *How a server can realize clustered FL efficiently by grouping the clients into clusters in a one-shot manner without requiring the number of clusters to be known apriori, but with substantially less communication cost?* In this work, we propose a novel algorithm, Principal Angles analysis for Clustered Federated Learning (PACFL), to address the above-mentioned challenges of clustered FL.

Contributions. We propose a new algorithm, PACFL, for federated learning that *directly* aims

to efficiently identify distribution similarities among clients by analyzing the *principal angles between the client data subspaces*. Each client wishing to join the federation applies a truncated SVD step on its local data in a *one-shot* manner to derive a *small set of principal vectors*, which form the principal bases of the underlying data. These principal bases provide a *signature* that succinctly captures the main characteristics of the underlying distribution. The client then provides this small set of principal vectors to the server so that the server can directly identify distribution similarities among the clients to form clusters. The privacy of data is preserved since no client data is ever sent to the server but a few $(2\text{-}5)$ principal vectors out of ≈ 500. Thus, the clients data cannot be reconstructed from those $(2\text{-}5)$ number of left singular vectors. However, in privacy-sensitive setups to provide extra protection and prevent any information leakage from clients to server, encryption mechanisms or differential privacy methods that achieve this end can be employed.

On the server side, it efficiently identifies distribution similarities among clients by comparing the principal angles between the client data subspaces spanned by the provided principal vectors – the greater the difference in data heterogeneity between two clients, the more orthogonal their subspaces. Unlike prior clustered FL approaches, which require time-consuming iterative learning of the clusters and substantial communication costs, our approach provides a simple yet effective clustered FL framework that addresses a broad range of data heterogeneity issues beyond simpler forms of Non-IIDness like label skews. Clients can immediately collaborate with other clients in the same cluster from the get-go.

Our novel PACFL approach has the flexibility to trade off between *personalization* and *globalization*. PACFL can naturally span the spectrum of identifying IID data distribution scenarios in which all clients should share training within only 1 cluster, to the other end of the spectrum where clients have extremely Non-IID data distributions in which each client would be best trained on just their own local data (i.e., each client becomes its cluster).

Our framework also naturally provides an elegant approach to handle newcomer clients unseen at training time by matching them with a cluster model that the client can further

personalized with local training. Realistically, new clients may arrive at the federation after the distributed training procedure. In our framework, the newcomer client simply provides its principal vectors to the server, and the server identifies via angle similarity analysis which existing cluster model would be most suitable, or the server can inform the client that it should train on its own local data to form a new cluster if the client's data distribution is not sufficiently similar to the distributions of the existing clusters. On the other hand, it is generally unclear how prior personalized or clustered FL algorithms can be extended to provide newcomer clients with similar capabilities.

Finally, we provide a convergence analysis of PACFL.

3.4.1 Clustered Federated Learning

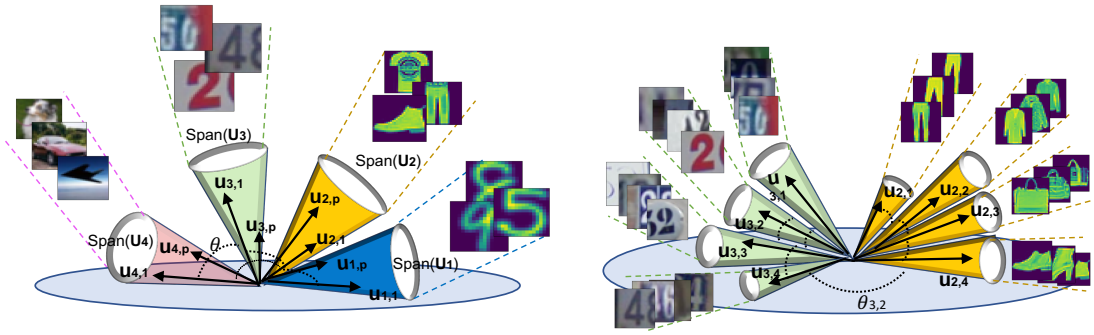


Figure 3.11. There must be a translation protocol enabling the server to understand similarity and dissimilarity in the distribution of data across clients without sharing data. These 2D figures intuitively demonstrate how the principal angle between the client data subspaces captures the statistical heterogeneity. **(Left)** Shows the subspaces spanned by the $\mathbf{U}_p$s of four different datasets (left to right: CIFAR-10, SVHN, FMNIST, and USPS). As can be seen, the principal angle between the subspaces of CIFAR-10 and SVHN is smaller than that of CIFAR-10 and USPS. Table 5.1 shows the exact principal angles between every pair of these subspaces. **(Right)** Shows the angle between the subspaces of different classes of SVHN and that of FMNIST. Indeed, the smallest principal angle between each pair of classes is different as well. For instance, on FMNIST, by setting p of $\mathbf{U}_p$ to 3, the smallest principal angle between Trouser and Dress is 22.47° while that of Trouser and Bag is 51.7° which stems from more similarity between the distribution of (Trouser, Dress) compared to that of (Trouser, Bag).

How Principal Angles Can Capture the Similarity Between Data/Features

The *cosine similarity* is a distance metric between vectors that is known to be more tractable and interpretable than the alternatives while exhibiting high precision clustering

Table 3.7. An example showing how distribution similarities among clients can be perfectly estimated by the principal angles between the client data subspaces. This table shows the proximity matrix of four datasets, where the UMAP visualization has been shown in Fig. 3.12 (c). The entries of this matrix are presented as $x(y)$, where x is the smallest principal angle between two datasets obtained from Eq. 3.1, and y is the summation over the principal angles between two datasets obtained from Eq. 3.2. We let of p in $\mathbf{U}_p$ be 2.

Dataset	CIFAR-10	SVHN	FMNIST	USPS
CIFAR-10	0 (0)	6.13 (12.3)	45.79 (91.6)	66.26 (132.5)
SVHN	6.13 (12.3)	0 (0)	43.42 (86.8)	64.86 (129.7)
FMNIST	45.79 (91.6)	43.42 (86.8)	0 (0)	43.36 (86.7)
USPS	66.26 (132.5)	64.86 (129.7)	43.36 (86.7)	0 (0)

properties, see, e.g. [77]. The idea is to note that for any two vectors x and y, by the dot product calculation $x \cdot y = \|x\| \|y\| \cos\theta$, we see that inverting the operation to solve for θ yields the angle between two vectors as emanating from the origin, which is a scale-invariant indication of their alignment. It presents a natural geometric understanding of the proportional volume of the embedded space that lies between the two vectors. Finally, by choosing to cluster using the data rather than the model, the variance of each SGD sample declines resulting in smoother training. By contrast, clustering by model parameters has the effect of increasing the bias of the clients' models to be closer to each other.

In order to obtain a computationally tractable small set of vectors to represent the data features, we propose to apply truncated SVD on each dataset. We take a small set of principal vectors, which form the principal bases of the underlying data distribution. Truncated SVD is known to yield a good quality balance between computational expense and representative quality of representative subspace methods [78]. Assume there are K number of datasets. We propose to apply truncated SVD on these data matrices, $\mathbf{D}_k, k = 1, ..., K$, whose columns are the input features of each dataset. Further, let $\mathbf{U}_p^k = [\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_p]$, $(p \ll \text{rank}(\mathbf{D}_\text{k}))$ be the p most significant left singular vectors for dataset k. We constitute the proximity matrix $\mathbf{A}$ as in Eq. 3.1 whose entries are the smallest principle angle between the pairs of $\mathbf{U}_p^k$ or as in Eq. 3.2 whose entries are the summation over the angle in between of the corresponding $\mathbf{u}$ vectors (in identical order) in each pair within $\mathbf{U}_p^k$, where $\mathbf{tr}(.)$ is the trace operator, and

$$\mathbf{A}_{i,j} = \Theta_1 \left(\mathbf{U}_p^i, \mathbf{U}_p^j\right), \;\; i,j = 1,...,K \tag{3.1}$$

$$\mathbf{A}_{i,j} = \mathbf{tr}\left(\arccos\left(\mathbf{U}_p^{i\,T} * \mathbf{U}_p^j\right)\right), \;\; i,j = 1,...,K \tag{3.2}$$

The smaller the entry of $\mathbf{A}_{i,j}$ is, the more similar datasets i and j are [3]. Before we proceed further, through some experiments on benchmark datasets, we highlight how the proposed method perfectly distinguishes different datasets based on their hidden data distribution by inspecting the angle between their data subspaces spanned by their first p left singular vectors. For a visual illustration of the result, we refer to Fig. 3.11. As can be seen, the principal angle between the subspaces of CIFAR-10 and SVHN is smaller than that of CIFAR-10 and USPS. Table 1 shows the exact principal angles between every pair of these datasets' subspaces. The entries of this table are presented as $x(y)$, where x is the smallest principal angle between two datasets obtained from Eq. 3.1, and y is the summation over the principal angles between two datasets obtained from Eq. 3.2. Table 3.7 reveals that the similarity and dissimilarity of the four different datasets have been accurately captured by the proposed method. We will provide more examples later on and will show that the similarity/dissimilarity being captured by the proposed method is consistent with well-known distance measures between two distributions including Bhattacharyya Distance (BD), Maximum Mean Discrepancy (MMD) [79], and Kullback–Leibler (KL) distance [80].

[3] It is noteworthy that in practice both of these equations work accurately. However, theoretically and rigorously speaking, when the number of the principal vectors, p, is bigger than 1, it can happen that one of the principal vectors of client k yields a small angle with its corresponding one for client k' while the other principal vectors of client k yield big angle with their corresponding ones for client k'. With that in mind, Eq. 3.2 is a more rigorous measure and it always truly captures the similarity between the client data subspaces.

Algorithm 7. PACFL

Require: Number of available clients N, sampling rate $R \in (0,1]$, clustering threshold β **Server:** Initialize the server model with θ_g^0.

for *each round t = 1, 2,...* **do**
  $m \leftarrow \max(R \cdot N, 1)$
  $\mathscr{S}_t \leftarrow \{k_1, ..., k_n\}$; set of n available clients
  for *each client* $k \in \mathscr{S}_t$ **in parallel do**
    if $t = 1$ *; It is done in one-shot* **then**
      client k sends $\mathbf{U}_p^k$ to the server $\mathbf{U} = [\mathbf{U}_p^{k_1}, ..., \mathbf{U}_p^{k_n}]$ $\mathbf{A} \leftarrow$ server forms $\mathbf{A}$ based on Eq. 3.1 or Eq. 3.2 $\{C_1, ..., C_Z\} = \text{HC}(\mathbf{A}, \beta)$ $\theta_{g,z}^0 \leftarrow \theta_g^0$; initializing all clusters with θ_g^0
    elseif *k is a new arriving client* **then**
      client k sends $\mathbf{U}_p^k$ to the server
      $\mathbf{A}$, $\mathbf{U}$ = PME($\mathbf{A}$, $\mathbf{U}$, $\mathbf{U}_p^k$) ; Alg. 8
      $\{C_1, ..., C_Z\} \leftarrow \text{HC}(\mathbf{A}, \beta)$; Update the clusters, determine the cluster ID of new client k
      client k receives the corresponding cluster model $\theta_{g,z}^t$ from the server
    else
      client k sends its cluster ID to the server and receives the corresponding cluster model $\theta_{g,z}^t$ from the server
    end
    $\theta_{k,z}^{t+1} \leftarrow$ ClientUpdate$(k; \theta_{g,z}^t)$: by SGD training
  end
  for $z = 1 : Z$ *; Z is the number of formed clusters* **do**
    $\theta_{g,z}^{t+1} = \sum_{k \in C_z} |D_k| \theta_{k,z}^{t+1} / \sum_{k \in C_z} |D_k|$;model averaging for each cluster
  end
end

Algorithm 8. Proximity Matrix Extension (PME)

Input: $\mathbf{A}_{old}$ { $M \times M$ proximity matrix formed by M number of seen clients}

Input: $\mathbf{U}_{old} = [U_p^1, ..., U_p^M]$ { Set of first p significant singular vectors of the M seen clients}

Input: $\mathbf{U}_{new} = [U_p^1, ..., U_p^B]$ {The set of first p significant singular vectors of the B new clients}

Output: $\mathbf{A}_{extended}$ {The extended $(M+B) \times (M+B)$ proximity matrix}

Output: $\mathbf{U}_{extended}$

PME($\mathbf{A}_{old}$, $\mathbf{U}_{old}$, $\mathbf{U}_{new}$)

$\mathbf{A}_{extended} \leftarrow [\mathbf{0}]_{(M+B)\times(M+B)}$

$U_{extended} \leftarrow [\mathbf{U}_{old}, \mathbf{U}_{new}]$

$\mathbf{A}_{extended}[1:M, 1:M] = \mathbf{A}_{old}[:,:]$

$\mathbf{A}_{extended}[M:M+B, M:M+B]$ {can be calculated based on Eq. 3.1 or Eq. 3.2}

Return $\mathbf{A}_{extended}$, $\mathbf{U}_{extended}$

Overview of PACFL

In this section, we begin by presenting our PACFL framework. The proposed approach, PACFL, is described in Algorithm 7. We first turn our attention to clustering clients data in a federated network. The proposed method is one-shot clustering and can be used as a simple pre-processing stage to characterize personalized federated learning to achieve superior performance relative to the recent iterative approach for clustered FL proposed in [54]. Before federation, each available client, k, performs truncated SVD on its own data matrix, $\mathbf{D}_k$ [4], and sends the p most significant left singular vectors $\mathbf{U}_p$, as their data *signature* to the central server. Next, the server obtains the proximity matrix $\mathbf{A}$ as in Eq. 3.1 or Eq. 3.2 where $K = |\mathscr{S}_t|$, and $\mathscr{S}_t$ is the set of available clients. When the number of clusters is unknown, for forming disjoint clusters, the server can employ agglomerative hierarchical clustering (HC) [68] on the proximity matrix $\mathbf{A}$. Hence, the cluster ID of clients is determined.

For training, the algorithm starts with a single initial model parameter θ_g^0. In the first iteration of PACFL a random subset of available clients $\mathscr{S}_t \subseteq [N]$, $|\mathscr{S}_t| = n$ is selected by the server and the server broadcasts θ_g^0 to all clients. The clients start training on their local data and

[4] Considering a client owns M data samples, each including N features, we assumed that the M data samples are organized as the columns of a matrix $D_k \in R^{N \times M}$.

perform some steps of stochastic gradient descent (SGD) updates, and get the updated model. The clients will only need to send their cluster membership ID and model parameters back to the central server. After receiving the model and cluster ID memberships from all the participating clients, the server then collects all the parameter updates from clients whose cluster ID are the same and conducts model averaging within each cluster. It is noteworthy that in Algorithm 7, β stands for the Euclidean distance between two clusters and is a parameter in HC.

Desirable properties of PACFL. Unlike prior work on clustered federated learning [54, 55], PACFL has much greater flexibility in the following sense. First, from a *practical* perspective, one of the desirable properties of PACFL is that it can handle the partial participation of clients. In addition, PACFL does not require to know in advance whether certain clients are available for participation in the federation. Clients can join and leave the network abruptly. In our proposed approach, the new clients that join the federation just need to send their data signature to the server and the server can easily determine the cluster IDs of the new clients by constituting a new proximity matrix without altering the cluster IDs of the other clients. In PACFL, prior information about the availability of certain clients is not required.

Second, PACFL can form the best fitting number of clusters, if a fixed number of clusters is not specified. However, in IFCA [54], the number of clusters has to be known apriori. Third, one-shot client clustering can be placed by PACFL for the available clients before the federation and prior information about the availability and the number of certain clients is not required. In contrast, IFCA constructs the clusters iteratively by alternating between cluster identification estimation and loss function minimization which is costly in communication.

Fourth, PACFL does not add significant additional computational overhead to the FedAvg baseline algorithm as it only requires running one-shot HC clustering before training. With that in mind, the computational complexity of the PACFL algorithm is the same as that of FedAvg plus the computational complexity of HC in one-shot ($(O(N^2))$ where N is the total number of clients).

Fifth, in case of either a certain and a fixed number of clients are not available at the initial stage or clients join and leave the network abruptly, clustering by PACFL can easily be applied in a few stages as outlined in Algorithm 8. Each new clients that become available for the federation, sends the signature of its data to the server and the server aggregates the signature of existing clients and the new ones as in $U_{extended}$ (Algorithm 8). Next, the server obtains the proximity matrix $\mathbf{A}$ as in Eq. 3 or Eq. 4 where all the new clients are included. *By keeping the same distance threshold* as before, the cluster ID of the new clients are determined without changing the cluster ID of the old clients.

Sixth, we should note that we tried some other clustering methods including graph clustering methods [81, 82] for PACFL and we noticed that the clustering algorithm does not play a crucial role in PACFL. As long as the clustering algorithm itself does not require the number of clusters to be known in advance, it can be applied in PACFL.

Seventh, the server cannot make use of the well-known distribution similarity measures such as BD, MMD [79], and KL [80] to group the clients into clusters due to the privacy constraints as they require accessing the data or important moments of the data distributions. As shown in Fig. 1, and Table 1 and also as will be shown in the Experiments Section, the proposed approach presents a simple and alternative solution to the above-mentioned measures in FL setups.

We also provide a convergence analysis of the method. The framework we use is from [43], which considers nonconvex learning with Non-IID data. Indeed unlike other works, we can obtain guarantees for nonconvex objectives, as appropriate for deep learning, because the clustering is performed on the *data* and not the parameters, thus no longer suffering from associated issues of multi-modality (multiple separate local minima). We shall see that it recovers the state-of-the-art (SOTA) convergence rate and performance guarantees for Non-IID data.

3.4.2 Experiments

Datasets and Models. We use an image classification task and 4 popular datasets, i.e., FMNIST [29], SVHN [83], CIFAR-10 [28], CIFAR-100 [28], to evaluate our method. For all experiments, we consider LeNet-5 [27] architecture for FMNIST, SVHN, and CIFAR-10 datasets and ResNet-9 [84] architecture for CIFAR-100 dataset.

Baselines and Implementation. To assess the performance of the proposed method against the SOTA, we compare PACFL against the following set of baselines. For baselines that train a single global model across all clients, we compare them with FedAvg [31], FedProx [32] FedNova [34], and SCAFFOLD [33]. For SOTA personalized FL methods, the baselines include LG-FedAvg [69], Per-FedAvg [51], Clustered-FL (CFL) [55], and IFCA [54]. In all experiments, we assume 100 clients are available and 10% of them are sampled randomly at each round. Unless stated otherwise, throughout the experiments, the number of communication rounds is 200 and each client performs 10 local epochs with a batch size of 10 and the local optimizer is SGD. We let p in $\mathbf{U}_p$ be 3-5.

Overall Performance

We compare PACFL with all the mentioned SOTA baselines for two different widely used Non-IID settings, i.e. Non-IID label skew, and Non-IID Dirichlet label skew. We report the mean and standard deviation for the average of final local test accuracy across all clients over 3 runs.

Non-IID Label Skew. In this setting, we first randomly assign $\rho\%$ of the total available labels of a dataset to each client and then randomly distribute the samples of each label amongst clients own those labels as in [37]. In our experiments we use Non-IID label skew 20%, and 30%, i.e. $\rho = \{20, 30\}\%$ respectively.

We further distribute the training data between the clients based on the Dirichlet distribu-

Table 3.8. Test accuracy comparison across different datasets for Non-IID label skew (20%). For each baseline, the average of final local test accuracy over all clients is reported. We run each baseline 3 times for 200 communication rounds with local epoch of 10.

Algorithm	FMNIST	CIFAR-10	CIFAR-100	SVHN
SOLO	95.92 ± 0.57	79.22 ± 1.67	32.28 ± 0.23	79.72 ± 1.37
FedAvg	77.3 ± 4.9	49.8 ± 3.3	53.73 ± 0.50	80.2 ± 0.8
FedProx	74.9 ± 2.6	50.7 ± 1.7	54.35 ± 0.84	79.3 ± 0.9
FedNova	70.4 ± 5.1	46.5 ± 3.5	53.61 ± 0.42	75.4 ± 4.8
Scafold	42.8 ± 28.7	49.1 ± 1.7	54.15 ± 0.42	62.7 ± 11.6
LG	96.80 ± 0.51	86.31 ± 0.82	45.98 ± 0.34	92.61 ± 0.45
PerFedAvg	95.95 ± 1.15	85.46 ± 0.56	60.19 ± 0.15	93.32 ± 2.05
IFCA	97.15 ± 0.01	87.99 ± 0.15	71.84 ± 0.23	95.42 ± 0.06
CFL	77.93 ± 2.19	51.11 ± 1.01	40.29 ± 2.23	73.62 ± 1.76
PACFL	$\mathbf{97.54 \pm 0.08}$	$\mathbf{89.30 \pm 0.41}$	$\mathbf{73.10 \pm 0.21}$	$\mathbf{95.77 \pm 0.18}$

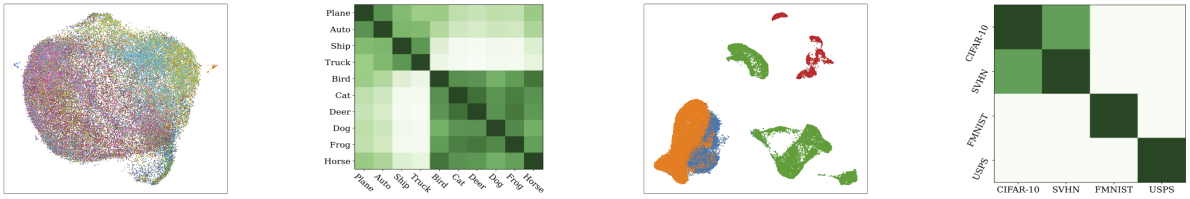
tion similar to [37]. In particular, we simulated the Non-IID partition into N clients by sampling $\mathbf{p}_i \sim Dir_N(\alpha)$ [5] and allocating the $\mathbf{p}_{i,j}$ proportion of the training data of class i to client j as in [37]. We set the concentration parameter of Dirichlet distribution, α to 0.1 throughout the experiments.

Table 3.8, and 3.9 show the results for Non-IID label skew 20% and 30% respectively. As can be seen, global FL baselines, i.e. FedAvg, FedProx, FedNova, and SCAFFOLD perform very poorly. That's due to weight divergence and model drift issues under heterogeneous settings [56]. We can observe from Table 3.8 that PACFL consistently outperforms all SOTA on all datasets. In particular, focusing on CIFAR-100 for Non-IID label skew 20%, PACFL outperforms all SOTA methods (by $+19\%, +18\%, +19\%, +18\%$ for FedAvg, FedProx, FedNova, SCAFFOLD) as well as all the personalized competitors (by $+27\%, +13\%, +1.5\%, +33\%$ for LG, PerFedAvg, IFCA, CFL). We tuned the hyperparameters in each baseline to obtain the best results. IFCA achieved the best performance with 2 clusters which is consistent with the results in [54]. Further, Table 3.10 demonstrates the results for Non-IID Dir (0.1). PACFL has achieved SOTA results over all the reported datasets. The best performance of IFCA with 2 clusters has been used to report the results in Table 3.10.

[5] The value of α controls the degree of Non-IID-ness. A big value of α e.g., $\alpha = 100$ mimics identical label distribution (IID), while $\alpha = 0.1$ results in a split, where the vast majority of data on every client are Non-IID.

Table 3.9. Test accuracy comparison across different datasets for Non-IID label skew (30%). For each baseline, the average accuracy over all clients is reported. We run each baseline 3 times for 200 rounds with 10 local epochs and a local batch size of 10.

Algorithm	FMNIST	CIFAR-10	CIFAR-100	SVHN
SOLO	93.93 ± 0.10	65 ± 0.65	22.95 ± 0.81	68.70 ± 3.13
FedAvg	80.7 ± 1.9	58.3 ± 1.2	54.73 ± 0.41	82.0 ± 0.7
FedProx	82.5 ± 1.9	57.1 ± 1.2	53.31 ± 0.48	82.1 ± 1.0
FedNova	78.9 ± 3.0	54.4 ± 1.1	54.62 ± 0.91	80.5 ± 1.2
Scafold	77.7 ± 3.8	57.8 ± 1.4	54.90 ± 0.42	77.2 ± 2.0
LG	94.21 ± 0.40	76.58 ± 0.16	35.91 ± 0.20	87.69 ± 0.77
PerFedAvg	92.87 ± 2.67	77.67 ± 0.19	56.42 ± 0.41	91.25 ± 1.47
IFCA	95.22 ± 0.03	80.95 ± 0.29	67.39 ± 0.27	93.02 ± 0.15
CFL	78.44 ± 0.23	52.57 ± 3.09	35.23 ± 2.72	73.97 ± 4.77
PACFL	$\mathbf{95.46 \pm 0.06}$	$\mathbf{82.77 \pm 0.18}$	$\mathbf{67.71 \pm 0.21}$	$\mathbf{93.13 \pm 0.27}$



((a)) **((b))** **((c))** **((d))**

Figure 3.12. The main message of this visualization is to understand the cluster structure of different datasets as well as the distribution similarity of different heterogeneous tasks/datasets. (a) depicts the UMAP visualization of CIFAR-10 classes. As can be seen, CIFAR-10 naturally has two super clusters, namely animals (cat, dog, bird, deer, horse, frog) and vehicles (car, plane, ship, truck), which are shown in the purple and green regions, respectively. This means that within each super cluster, the distance between the distribution of the classes is small. While the distance between the distributions of the two super clusters is quite huge. Since the union of clients data is CIFAR-10, two cluster is enough to handle the Non-IIDness across clients. This is the reason that the best accuracy performance on CIFAR-10 is obtained when the number of clusters is 2. (b) We obtained the proximity matrix $\mathbf{A}$ as in Eq. 3.1 and sketched it. The entries of $\mathbf{A}$ are the smallest principle angle between all pairs of classes of CIFAR-10. This concurs with (a) showing the cluster structure of CIFAR-10. (c) The data of MIX-4 is naturally clustered into four clusters. The structure of the 4 clusters is also accurately suggested by PACFL for this task. (d) We did the same thing as in (b) for MIX-4 as well and sketched the matrix.

Table 3.10. Test accuracy comparison across different datasets for Non-IID Dir (0.1). For each baseline, the average of final local test accuracy over all clients is reported. We run each baseline 3 times for 200 communication rounds with 10 local epochs and a local batch size of 10.

Algorithm	FMNIST	CIFAR-10	CIFAR-100
SOLO	69.71 ± 0.99	41.68 ± 2.84	16.83 ± 0.51
FedAvg	82.91 ± 0.83	38.22 ± 3.28	44.52 ± 0.42
FedProx	84.04 ± 0.53	42.29 ± 0.95	45.52 ± 0.72
FedNova	84.50 ± 0.66	40.25 ± 1.46	46.52 ± 1.34
Scafold	10.0 ± 0.0	10.0 ± 0.0	43.73 ± 0.89
LG	74.96 ± 1.41	49.65 ± 0.37	23.59 ± 0.26
PerFedAvg	80.29 ± 2.00	53.58 ± 1.57	33.94 ± 0.41
IFCA	85.01 ± 0.30	51.16 ± 0.49	47.67 ± 0.28
CFL	74.13 ± 0.94	42.30 ± 0.25	31.42 ± 1.50
PACFL	$\mathbf{85.28 \pm 0.12}$	$\mathbf{51.22 \pm 0.25}$	$\mathbf{49.80 \pm 0.09}$

Globalization and Personalization Trade-off

To cope with the statistical heterogeneity, previous works incorporated a proximal term in local optimization or modified the model aggregation scheme at the server side to take the advantage of a certain level of personalization [32, 36, 44]. Though effective, they lack the flexibility to trade off between *personalization* and *globalization*. Our proposed PACFL approach can naturally provide this globalization and personalization trade-off. Fig. 3.13, and Fig. 3.14 visualize the accuracy performance behavior of PACFL versus different values of β which is the L_2 (Euclidean) distance between two clusters when the proximity matrix obtained as in Eq. 3.1, or Eq. 3.2. In other words, β is a threshold controlling the number of clusters as well as the similarity of the data distribution of clients within a cluster under Non-IID label skew. The blue curve and the red bars demonstrate the accuracy, and the number of clusters respectively for each β. Varying β in a range that depends upon the dataset, PACFL can sweep from training a fully global model (with only 1 cluster) to training fully personalized models for each client.

As is evident from Fig. 3.13, the behavior of PACFL on each dataset is similar. In particular, increasing β, decreases the number of clusters (by grouping more number of clients within each cluster and sharing more training) which realizes more globalization. When β is

big enough, PACFL will group all clients into 1 cluster and the scenario reduces to the FedAvg baseline (pure globalization). On the contrary, decreasing β, increases the number of clusters, which leads to more personalization. When β is small enough, individual clusters would be formed for each client and the scenario degenerates to the SOLO baseline (pure personalization). As demonstrated, on all datasets, all clients benefit from some level of globalization. This is the reason why decreasing the number of clusters can improve the accuracy performance in comparison to SOLO. In general, *finding the optimal trade-off between globalization and personalization depends on the level of heterogeneity of tasks, the intra-class distance of the dataset, as well as the data partitioning across the clients*. This is precisely what PACFL is designed to do, to find this optimal trade-off before initiating the federation via the proximity matrix at the server. IFCA lacks this trade-off capability as it must define a fixed number of clusters $(C > 1)$ or with $C = 1$ it would degenerate to FedAvg.

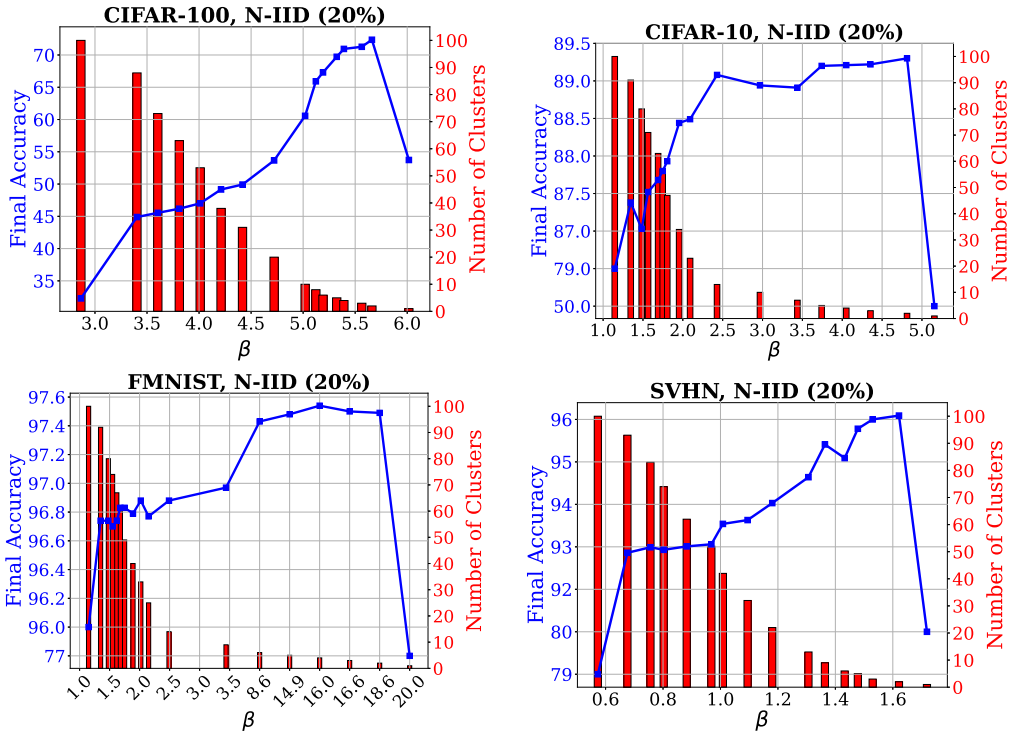


Figure 3.13. Test accuracy performance of PACFL versus the clustering threshold β (when the proximity matrix obtained as in Eq. 3.1), and the number of fitting clusters for Non-IID label skew (20%) on CIFAR-10/100, FMNIST, and SVHN datasets. Each point in the plots are obtained by 200 communication rounds with local epoch of 10, local batch size of 10 and SGD local optimizer.

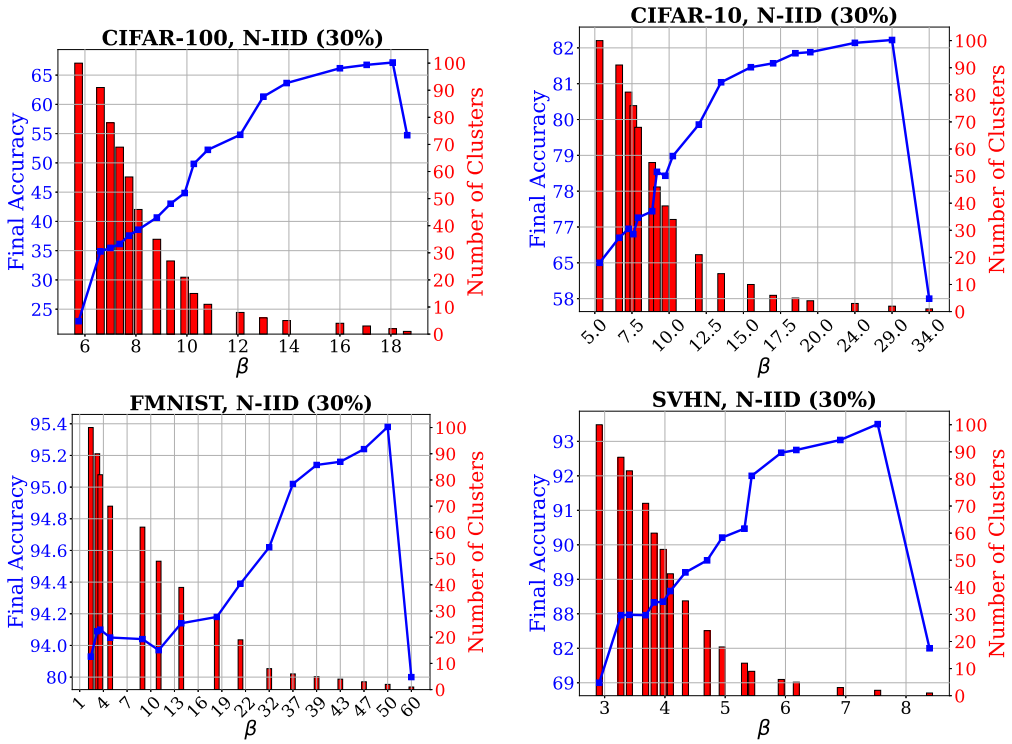


Figure 3.14. Test accuracy performance of PACFL versus the clustering threshold β (when the proximity matrix obtained as in Eq. 3.2), and the number of fitting clusters for Non-IID label skew (30%) on CIFAR-10/100, FMNIST, and SVHN datasets. Each point in the plots is obtained by 200 communication rounds with the local epoch of 10, local batch size of 10, and SGD local optimizer.

Mixture of 4 Datasets

Existing studies have been evaluated on simple partitioning strategies, i.e., Non-IID label skew (20%) and (30%) [37]. While these partitioning strategies synthetically simulate Non-IID data distributions in FL by partitioning a dataset into multiple smaller Non-IID subsets, they cannot design real and challenging Non-IID data distributions. According to the prior sections, due to the small intra-class distance (similarity between distribution of the classes) in the used benchmark datasets, all baselines benefited highly from globalization. This is the reason that PACFL and IFCA could achieve high performance with only 2 clusters.

In order to better assess the potential of the SOTA baselines under a real-world and challenging Non-IID task where the local data of clients have strong statistical heterogeneity, we design the following experiment naming it as MIX-4. We assume that each client owns data samples from one of the four datasets, i.e., USPS [85], CIFAR-10, SVHN, and FMNIST. In particular, we distribute CIFAR-10, SVHN, FMNIST, USPS among 31, 25, 27, 14 clients respectively (100 total clients) where each client receives 500 samples from all classes of only one of these datasets. This is a hard Non-IID task. We compare our approach with the SOTA baselines in the classification of these four different datasets, and we present the average of the clients' final local test accuracy in Table 5.6. As can be seen, IFCA is unable to effectively handle this difficult scenario with tremendous data heterogeneity with just two clusters, as suggested in [54] as the best fitting number of clusters. IFCA (2) with 2 clusters performs almost as poorly as the global baselines while PACFL can find the optimal number of clusters in this task (four clusters) and outperforms all SOTA by a large margin. The results of IFCA (4) with 4 clusters is 76.79 ± 0.43. As observed, PACFL surpasses all the global competitors (by $+14\%, +15\%, +16\%, +8\%$ for FedAvg, FedProx, FedNova, SCAFFOLD) as well as all the personalized competitors (by $+19\%, +35\%, +7\%, +16\%$ for LG, PerFedAvg, IFCA, CFL, respectively). Further, the visualization in Fig. 3.12(c) and 3.12(d) also show how PACFL determines the optimal number of clusters on MIX-4.

Table 3.11. The benefits of PACFL are particularly pronounced when the tasks are extremely Non-IID. This table evaluates different FL approaches in the challenging scenario of MIX-4 in terms of the top-1 test accuracy performance. While all competing approaches have substantial difficulties in handling this scenario with tremendous data heterogeneity, the results clearly show that PACFL is very robust even under such difficult data heterogeneity scenarios.

Algorithm	MIX-4	Algorithm	MIX-4
SOLO	55.08 ± 0.29	LG	58.49 ± 0.46
FedAvg	63.68 ± 1.64	PerFedAvg	42.60 ± 0.60
FedProx	61.86 ± 3.73	IFCA (2)	70.32 ± 3.57
FedNova	60.92 ± 3.60	CFL	61.18 ± 2.63
Scaffold	69.26 ± 0.84	**PACFL**	$\mathbf{77.83 \pm 0.33}$

How Many Clusters Are Needed?

As we emphasized in prior sections, one of the significant contributions of PACFL is that the server can easily determine the best fitting number of clusters just by analyzing the proximity matrix without running the whole federation. For instance, for IID scenarios, we expect the best fitting number of clusters to be one. The reason behind this is that under IID setting, since all clients have similar distributions, they can share in the training of the averaged global model to benefit all. On the other hand, in the case of MIX-4, we expect the best fitting number of clusters to be four. More generally, we expect the best fitting number of clusters to be dependent on the similarities/dissimilarities in distributions among the clients. We empirically show in Fig. 3.13 that the best accuracy results on CIFAR-100, CIFAR-10, SVHN, and FMNIST for Non-IID label skew (20%) are obtained when the number of clusters is 2, 2, 2, and 4, respectively.

The UMAP [86] visualization in Fig. 3.12(a) also confirms that two clusters are the best case for training the local models on partitions of CIFAR-10 dataset. Broadly speaking, CIFAR-10 has 2 big classes, i.e., a class of animals (cat, dog, deer, frog, horse, bird) and a class of vehicles (airplane, automobile, ship and truck). Fig. 3.12(b) also depicts the proximity matrix of CIFAR-10 dataset, whose entries are the principal angle between the subspace of every pairs of 10 classes (labels). This further confirms that our proposed method perfectly captures the level of heterogeneity, thereby finding the best fitting number of clusters in a privacy preserving manner. In particular, our experiments demonstrate that the clients that have the sub-classes of

these two big classes have common features and can improve the performance of other clients that own sub-classes of the same big class if they are assigned to the same cluster. A similar observation can be seen for other datasets.

In PACFL, the server only requires to receive the *signature* of the clients data in one-shot and thereby initiating the federation with the best fitting number of clusters. This translates to several orders of magnitude in communication savings for PACFL. However, as mentioned in [54], IFCA treats the number of clusters as a hyperparameter which is optimized after running the whole federation with different numbers of clusters which increases the communication cost by several orders of magnitude.

Generalization to Newcomers

PACFL provides an elegant approach to handle *newcomers* arriving after the federation procedure, to learn their personalized model. In general, for all other baselines, it is not clear how they can be extended to handle clients unseen at training (federation). We show in Algorithm 9 how PACFL can simply be generalized to handle clients arriving after the end of the federation, to learn their personalized model. The unseen client will send the signature of its data to the server and the server determines which cluster it belongs to. The server then sends the corresponding model to the newcomer and the newcomer fine-tunes the received model. To evaluate the quality of newcomers' personalized models, we design an experiment under Non-IID label skew (20%) where only 80 out of 100 clients are participating in a federation with 50 rounds. Then, the remaining 20 clients join the network at the end of the federation and receive the model from the server and personalize it for only 5 epochs. The average final local test accuracy of the unseen clients is reported in Table 3.12. Focusing on CIFAR-100, as observed, some of the personalized baselines including LG and PerFedAvg perform as poorly as global baselines and SOLO. PACFL consistently outperforms other baselines by a large margin.

Algorithm 9. Generalization to newcomers after federation

Server: An existing $\mathbf{A}$, $\mathbf{U}$, $\{C_1, ..., C_Z\}$, clustering threshold β

Require: a set of B newcomers and their corresponding first p significant singular vectors, i.e. $\mathbf{U}_{new} = [U_p^1, ..., U_p^B]$

$\mathbf{A}$, $\mathbf{U}$ = PME($\mathbf{A}$, $\mathbf{U}$, $\mathbf{U}_{new}$) {Alg. 8}

$\{C_1, ..., C_Z\} \leftarrow$ HC($\mathbf{A}, \beta$) {Updating the clusters and determining the cluster ID of each new client k}

Each new client k receives the corresponding cluster model $\theta_{g,z}$ from the server

FineTune($k; \theta_{g,z}$): by SGD training {this step is optional}

Table 3.12. Average local test accuracy across unseen clients' on different datasets for Non-IID label skew (20%).

Algorithm	FMNIST	CIFAR-10	CIFAR-100	SVHN
SOLO	95.13 ± 0.42	82.30 ± 1.00	27.26 ± 0.98	91.5 ± 0.64
FedAvg	77.61 ± 3.78	31.01 ± 1.83	32.19 ± 0.32	71.78 ± 3.43
FedProx	74.30 ± 4.70	27.56 ± 3.24	32.41 ± 1.17	74.30 ± 4.70
FedNova	74.66 ± 2.81	31.48 ± 1.49	33.18 ± 0.80	73.04 ± 3.65
Scafold	73.97 ± 1.68	37.22 ± 1.34	23.90 ± 2.61	64.96 ± 4.74
LG	94.58 ± 0.33	77.98 ± 1.61	10.63 ± 0.21	89.48 ± 0.65
PerFedAvg	89.88 ± 0.38	73.79 ± 0.51	30.09 ± 0.35	67.48 ± 2.88
IFCA	96.29 ± 0.04	84.98 ± 0.41	55.66 ± 0.20	94.83 ± 0.14
PACFL	$\mathbf{96.36 \pm 0.20}$	$\mathbf{87.14 \pm 0.15}$	$\mathbf{59.16 \pm 0.42}$	$\mathbf{95.25 \pm 0.08}$

Communication Cost

Learning with Limited Communication In this section we consider circumstances that frequently arise in practice, where a limited amount of communication round is permissible for federation under a heterogeneous setup. To this end, we compare the performance of the proposed method with the rest of SOTA. Herein, we consider a limited communication rounds budget of 80 for all personalized baselines and present the average of final local test accuracy over all clients versus the number of communication rounds for Non-IID label skew (20%, and (30%)) in Fig. 3.15 and 3.16. Focusing on Non-IID label skew (20%), our proposed method requires only 30 communication rounds to converge in CIFAR-10, SVHN, and FMNIST datasets. CFL yields the worst performance on all benchmarks across all datasets, except for CIFAR-100. Per-Fedavg seems to benefit more from higher communication rounds. IFCA is the closest line to ours for CIFAR-10, SVHN, and FMNIST, however, PACFL consistently outperforms IFCA. This can be explained by the fact that IFCA randomly initializes cluster models that are inherently noisy, and many rounds of the federation are required until the formation of clusters is

stabilized. Further, IFCA is sensitive to initialization and a good initialization of cluster model parameters is key for convergence [87]. This issue can be further pronounced by the results presented in Table 3.13 which demonstrate the number of communication rounds required to achieve a designated target accuracy. In this table, "−−" means that the baseline is unable to reach the specified target accuracy. As can be seen, PACFL beats IFCA and all SOTA methods.

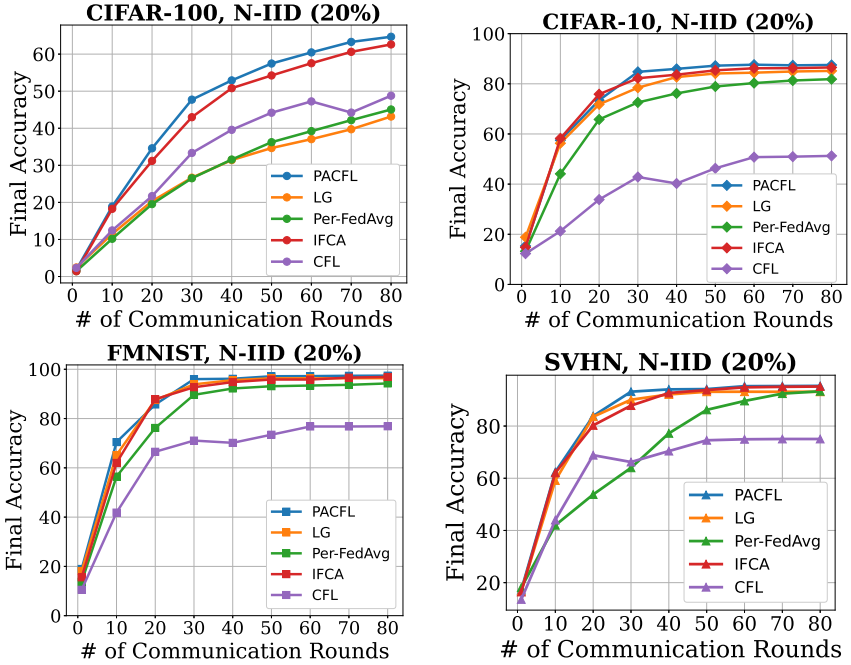


Figure 3.15. Test accuracy versus the number of communication rounds for Non-IID (20%). PACFL converges fast to the desired accuracy and consistently outperforms strong competitors.

Table 3.13. Comparing different FL approaches for Non-IID (20%) in terms of the required number of communication rounds to reach target top-1 average local test accuracy.

Algorithm	FMNIST	CIFAR-10	CIFAR-100	SVHN
Target	75	80	50	75
FedAvg	200	−−	130	150
FedProx	200	−−	115	200
FedNova	−−	−−	120	150
Scafold	−−	−−	−−	−−
LG	13	33	−−	16
PerFedAvg	19	60	110	39
IFCA	14	25	40	17
CFL	47	−−	−−	−−
PACFL	**12**	**24**	**37**	**15**

Further, tables 3.14 and 3.15 respectively demonstrate the communication cost (in **Mb**)

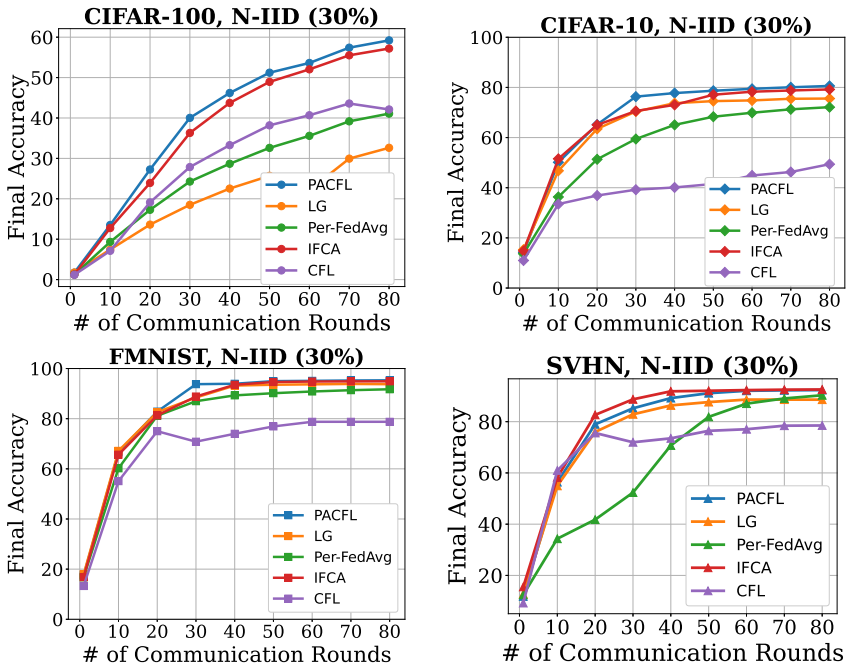


Figure 3.16. Test accuracy versus the number of communication rounds for Non-IID (30%). PACFL converges fast to the desired accuracy and consistently outperforms strong competitors, except in SVHN.

required to achieve the target test accuracies for Non-IID label skew (30%), and Non-IID Dir (0.1) across different datasets.

As can be seen, PACFL is drastically more communication-efficient than all the baselines except for LG in all target accuracies. *Global* baselines either cannot reach the target accuracy or require a high communication cost. Focusing on CIFAR-100 in Table 3.14, PACFL can reduce the communication cost by $\times(1.6-3.2)$ to reach the target accuracies. The communication cost is particularly crucial, as one single transfer of the parameters of ResNet-9 already takes up 211.88 **Mb**. Again considering CIFAR-100 in Table 3.14, to achieve the target accuracy of 50%, IFCA requires 3495.19 **Mb** of cumulative communication, translating to more than 1.7 orders of magnitude in communication savings for PACFL. This is because in every communication round, all clusters (models) at the server will have to be sent to all the participating clients which significantly increases the communication cost. Tables 3.13, 3.14 and 3.15 confirm that PACFL requires fewer communication cost/rounds compared to the SOTA algorithms to reach the target accuracies.

Table 3.14. Comparing different FL approaches for Non-IID label skew (30%) in terms of the required communication cost in **Mb** to reach target top-1 average local test accuracy. We evaluate on FMNIST, CIFAR-10, CIFAR-100, and SVHN.

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

Table 3.15. Comparing different FL approaches for Non-IID Dir (0.1) in terms of the required communication cost in **Mb** to reach target top-1 average local test accuracy. We evaluate on FMNIST, CIFAR-10, and CIFAR-100.

Algorithm	FMNIST	CIFAR-10	CIFAR-100
Target	75%	45%	40%
FedAvg	9.92	−−	3389.47
FedProx	9.92	−−	2965.52
FedNova	9.92	−−	3178.03
Scafold	−−	−−	5402.44
LG	14.09	**2.46**	−−
PerFedAvg	31.74	18.25	−−
IFCA	13.09	13.09	4575.89
CFL	79.36	−−	−−
PACFL	**8.73**	8.73	**2372.84**

Clustering Analysis

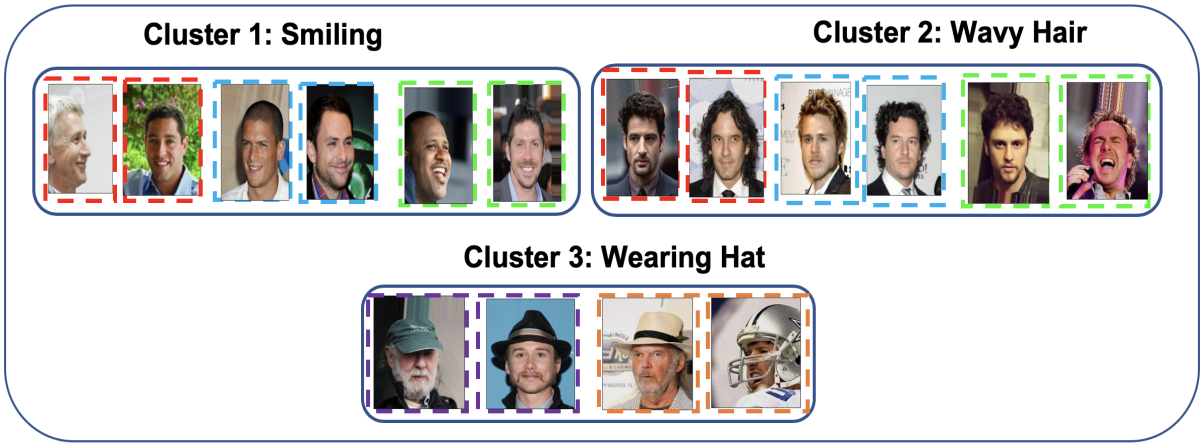


Figure 3.17. A case study on a setting consisting of three clusters that extracted from a random round of federation to testify whether the clients populated into one cluster have similar attribute/data distribution. The images with the same colored edge in each cluster belong to a particular client who participated in that round.

To further evaluate the effectiveness of our proposed approach, i.e., leveraging the principal angle between the subspaces spanned by the first p significant left singular vectors of the clients' data and whether the clients populated into one cluster have similar data/attributes, we conducted a clustering analysis via an illustration. Consider the scenario where we are interested in forming clusters of images for a face recognition task on the CelebA [88] dataset. In particular, assume that we are inclined to form clusters of clients for each attribute and find a good model for each cluster. To do so, we partition the data of CelebA based on three attributes namely "smiling", "wavy hair", and "wearing hat" such that each partition only has one of the attributes and none of the partitions have overlapped with the other partitions on the selected attributes. Then, we divide each of the partitions into shards and randomly assign to all clients and run some rounds of the federation. To intuitively judge whether the clients populated into the same cluster have similar data/attributes, we monitored the clustering result of local models derived from PACFL on the CelebA dataset. As shown in Fig. 3.17, the face recognition task in cluster 1, cluster 2, and cluster 3, is likely to handle Smiling faces, celebrities with wavy hair, and people wearing hats.

3.4.3 Convergence Analysis

We take the perspective of concentrating on one cluster and considering the set of clients assigned to it as performing Local SGD. Since clients are selected at random and data is Non-IID, we adapt the convergence framework presented in [43] and consider how the presence of potentially misidentified clients (meaning a client's data truly belongs in cluster j but is assigned to cluster k) affect the convergence guarantees. We show that the rate does not change, and only the order constant is increased by a quantity proportional to the maximum gradient norm of the loss associated with the misidentified clients.

Formally, we define SGD training as seeking to solve the optimization problem,

$$\min_{\theta} \sum_{k \in C} F_k(\theta)$$

We now place PACFL along the lines of theoretical analysis associated with [43]. This paper analyzed the convergence guarantees of distributed local SGD algorithms with Non-IID data, however with some measure of the similarity or dissimilarity of the data. Using this framework, we consider the convergence of the global parameter set $\theta_{g,z}^{t}$ for some cluster z on the sum of losses corresponding to the data on the clients associated with that cluster. The measure introduced in [43] to indicate data distributions that are not identical but similar is called *Weighted Gradient Diversity*, defined as,

$$\Lambda(\theta, C) := \frac{\sum_{k \in C} \|\nabla F_k(\theta)\|^2}{|C| \, \|\sum_{k \in C} \nabla F_k(\theta)\|^2} \tag{3.3}$$

where we have defined their clients' model parameters to be equal, i.e., $q_k = 1/|C|$ for all k and now included the dependence on $|C|$.

At each round, a set of clients is randomly chosen and a sequence of SGD steps are taken, and then they are averaged across all of the clients, and then the local procedures start again. This procedure is exactly the same as the classic local SGD, as analyzed in [43], with two primary

modifications:

1. There are several concurrent clusters performing independent local SGD, with the clusters defined by the similarity measure. Since generally speaking the structure of the behavior is the same across clusters, an analysis on one is representative.

2. Since clustering is a statistical and noisy procedure, i.e., there is no guarantee that the clusters are *correct*, we must perform the analysis based on the understanding that there exists a decomposition of the clients included in that cluster as correctly or incorrectly identified. In particular, correct identification implies that the bound in Eq. 3.3 holds, and incorrect identification implies no guaranteed similarity in the gradients. We note that this is a worst-case assumption – even if the data is distinct enough to be clustered, we would expect some inter-cluster similarity, even if it's less than intra-cluster similarity.

In the analysis, we shall drop the subscript z identifying the cluster (i.e., $\theta^t_{g,z} = \theta^t_g$ considering one as representative. With this, we introduce notation as follows:

1. c is the total number of clients available associated to the cluster.

2. There are E local SGD steps (the ClientUpdate step in PACFL) that come with the client update.

3. Each SGD step for every client is taken as a minibatch of size B, allowing for variance control.

4. We now consider t the counter of the total number of SGD steps rather than "major" averaging iterations (rounds), as in the description of PACFL.

5. There are K randomly chosen clients belonging to the cluster at each iteration.

This places the algorithm squarely in the framework of local SGD for Non-IID data when we restrict consideration to the performance associated with just one of the clusters. The one

modification we intend to study to this framework is the presence of misidentified clients; due to statistical noise there are some members of the cluster that do not have similar data, and thus do not exhibit gradient similarity. Formally, we make the following assumption regarding correct and incorrect identification,

Assumption 2. *Let us consider that cluster that clients* $\bar{C} = \{1, ..., \bar{c}\}$ *have been correctly identified, i.e.,* $\Lambda(\theta, C) \leq \lambda$. *Furthermore, assume that there exists* U_g *such that*

$$\|\nabla F_k(\theta)\| \leq U_g \tag{3.4}$$

for all $k \in C \setminus \bar{C} = \{\bar{c}+1, ..., c\}$ *and* $\theta \in \{\{\theta_g^t\}, \{\theta_k^t\}\}$.

Thus, we consider the case of $c - \bar{c}$ incorrectly identified clients whose gradients are effectively garbage to the optimization process for the cluster of interest. We will investigate the degree to which they slow down or bias the optimization procedure. Of course, a priori we cannot expect the gradient to be bounded, especially for objectives with a quadratic growth (e.g. PL inequality) property, however, in practice one can see the dependence of the convergence result below on U_g as indicating the effect misidentified clusters present for the convergence rate. Naturally, if a misidentified client's gradients are effectively garbage for a cluster, the effect on the convergence will be dependent on how large these gradients can get.

A significant contribution of PACFL is the use of the *data* rather than the loss value as a mechanism of cluster identification. This distinction from works such as [54] permits analysis to be performed for non-convex objectives, uniquely in the recent cluster literature. Let $F(\theta) = \sum_{k \in C} F_k(\theta)$. Recall that θ_g^t are the parameter values averaged across the clients associated with the cluster and let g_g^t be the averaged stochastic gradients at iteration t while g_j^t are the individual clients' stochastic gradients. We define each SGD iteration as:

$$\theta_j^{t+1} = \theta_j^t - \eta_t g_j^t \tag{3.5}$$

where g_j^t is a stochastic gradient. We present some problems and algorithmic assumptions below.

Assumption 3. *1. Each F_k is L-smooth, i.e., $\|\nabla F_k(\theta) - \nabla F_k(\theta')\| \leq L\|\theta - \theta'\|$*

2. *$F(\theta)$ is lower bounded by f^**

3. *$F(\theta)$ satisfies the Polyak-Łojasiewicz (PL) inequality with constant μ, i.e., $\frac{1}{2}\|\nabla F(\theta)\|^2 \geq \mu(F(\theta) - F(\theta^*))$*

4. *The stochastic gradient estimates are unbiased and have a variance bound according to,*

$$\mathbb{E}\left[\|g_j^t - \nabla F_j(\theta_j^t)\|^2\right] \leq C_1\|\nabla F_j(\theta_j^t)\|^2 + \frac{\sigma^2}{B}$$

where σ is a positive constant reflecting the individual sample variance.

We now present the main convergence results. Their proofs, which involve minor modifications of [43], are in the Appendix. For the first result, let $\kappa = L/\mu$, the ratio of the largest to smallest (throughout the objective landscape) eigenvalue of the Hessian of the objective, be the condition number of the problem.

Theorem 3. *There exist constants a and B depending on the problem and λ such that for sufficiently large E, the number of local stochastic gradient updates, and K, the number of sampled nodes from each cluster, it holds that, after T total iterations,*

$$\begin{aligned}\mathbb{E}\left[F(\theta_g^T) - F^*\right] \leq \tfrac{a^3}{(T+a)^3}\mathbb{E}\left[F(\theta_g^0) - F^*\right] + \\ \tfrac{4\kappa\sigma^2 T(T+2a)}{\mu KB(T+a)^3} + \tfrac{256\kappa^2\sigma^2 T(E-1)}{\mu KB(T+a)^3}\end{aligned} \tag{3.6}$$

This matches the standard rate for objectives with quadratic growth for local SGD, i.e.,

$$\mathbb{E}\left[F(\theta_g^T) - F^*\right] = O\left(\frac{1}{KT}\right)$$

The proof is in subsection C.1.1 of the Appendix C. For the general non-convex objectives we have the following result:

Theorem 4. *Let a constant stepsize η satisfy,*

$$-\frac{\eta}{2}+\frac{\lambda(K+1)L^2\eta^3[2C_1+E(E+1)]}{2K}+\frac{\lambda L^2\eta^2}{2}\left(\frac{C_1}{K}+1\right)\leq 0$$

It holds that,

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla F(\theta_g^t)\|^2 \leq \frac{2(F(\theta_g^t)-f^*)}{\eta T}+ \frac{L\eta}{2}\left(\frac{\sigma^2}{KB}+\left(\frac{c}{K}+1\right)U_g^2\right)+ \frac{\eta^2L^2(K+1)((E+1)\sigma^2+(C_1+E(E+1))U_g^2)}{KB}$$

Thus, with stepsize $\eta=\frac{1}{L}\sqrt{KT}$, the standard sublinear convergence rate on the average expected gradient holds, i.e.,

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla F(\theta_g^t)\|^2=O\left(\frac{1}{\sqrt{KT}}\right)$$

In summary, as long as there is some bound on all of the gradients as evaluated at the parameter values of the misidentified cluster, there is neither a bias in the solution nor a slowdown of the rate of convergence, simply an increase in the constant. We can interpret the convergence as indicating the degree to which being proportional to the square of the largest gradient norm across the parameter values at which a stochastic gradient is evaluated for these clients during training.

3.4.4 Consistency with other Distribution Similarity Metrics

We provide a simple synthetic example to show that our proposed method can effectively capture the similarity/dissimilarity between distributions by leveraging the principal angle between of the data subspaces spanned by the first p significant left singular vectors of the data. In particular, as shown in Table 3.16, in each column we sample 100 data points randomly from two Multivariate Gaussian Distribution of dimension 20 with different μ, and Σ. We then compare our approach with the widely used distance measures including Bhattacharyya distance [89], Maximum Mean Discrepancy (MMD) [79], and Kullback–Leibler (KL) distance [80] in inspecting the similarity/dissimilarity of the two distributions. Taking

the first two columns into account, by fixing Σ and changing μ, we are increasing the distance between the distributions and we expect to see a bigger distance in the second column compared to the first column. A similar discussion can be made about the comparison of the third and fourth columns.

Table 3.16. Demonstrating that our proposed method can capture the similarity/dissimilarity between two distributions and its results are consistent with that of the well-known distance measures. The results of PACFL is presented as $x(y)$, where x obtained from Eq. 3.1, and y obtained from Eq. 3.2. We let p of U_p be 3.

Distribution / Measure	$\mathcal{N}_1(\mu,\Sigma)$ $\mathcal{N}_2(2\mu,\Sigma)$	$\mathcal{N}_1(\mu,\Sigma)$ $\mathcal{N}_2(3\mu,\Sigma)$	$\mathcal{N}_1(\mu,\Sigma)$ $\mathcal{N}_2(\mu,2\Sigma)$	$\mathcal{N}_1(\mu,\Sigma)$ $\mathcal{N}_2(\mu,5\Sigma)$
Bhattacharyya [89]	3.07	10.31	1.14	2.13
KL [80]	15.18	47.12	3.31	5.79
MMD [79]	0.6218	0.6238	0.6234	0.6246
PACFL	10.73 (32.2)	18.41 (55.24)	8.93 (26.79)	13.72 (41.15)

3.4.5 Implementation

To be consistent, we adapt the official public codebase of Qinbin et al. [37] [6] to implement our proposed method and all the other baselines with the same codebase in PyTorch V. 1.9. We used the public codebase of LG-FedAvg [52][7], Per-FedAvg [51] [8], Clustered-FL (CFL) [55] [9], and IFCA [54] [10] in our implementation. For all the global benchmarks, including FedAvg [31], FedProx [32], FedNova [34], Scaffold [33] we used the official public codebase of [37] [6].

Unlike the original paper and the official implementation of LG-FedAvg[52] [7], we did not use the model found by performing many rounds of FedAvg as the initialization for LG-FedAvg in our implementation. Instead, for a fair comparison we initialized the models randomly with the same random seed in all other baselines. Also, the original implementation of LG-FedAvg [7] computes the average of the maximum local test accuracy for each client over

[6] https://github.com/Xtra-Computing/NIID-Bench
[7] https://github.com/pliang279/LG-FedAvg
[8] https://github.com/CharlieDinh/pFedMe
[9] https://github.com/felisat/clustered-federated-learning
[10] https://github.com/jichan3751/ifca

the entire federation, but we compute and report the average of the final local test accuracy of all clients for a fair comparison. In all the experiments, we have reported the results over all baselines on the same 100 clients with the same Non-IID partitions to have a fair comparison.

Implementation Details for MIX-4

As stated in Section 5.3.5 of the Chapter, we set the number of clients to 100 and distribute CIFAR-10, SVHN, FMNIST, and USPS amongst 31, 25, 27, 14 clients such that each client receives 500 samples from all the available classes in the corresponding dataset (50 samples per each class). We further zero-pad FMNIST, and USPS images to make them 32×32, and repeat them to have 3 channels. This pre-processing for FMNIST, and USPS is required to make the images the same size as CIFAR-10 and SVHN so that we can have a consistent model architecture in this task. We used LeNet-5 architecture with the details in Table 3.17 and modified the last layer to have 40 outputs corresponding to the 40 different labels in total (each dataset has 10 classes). The number of communication rounds is set to 50, and each client performs 5 local epochs. Tables 3.19, and 3.20 present more details about other hyper-parameter grids used in this experiment.

Hyper-parameters & Architectures

Tables 3.19 and 3.20 summarize the hyper-parameter grids used in our experiments. In all experiments, we use SGD as the local optimizer and the local batch size is 10. For the hyper-parameters exclusive for certain approaches, e.g. number of clusters in CFL or IFCA, we used the same values as described in the original papers.

For CFL, we lower the learning rate to address its divergence problem on some of the datasets as also specified in the CFL paper [55]. Moreover, the CFL implementation by the authors[9] does not include the parameter γ despite of the large body of discussion in the paper, so we leave it out as well.

For IFCA, we observe that the results with 2, and 3 clusters are almost the same and

Table 3.17. The details of LeNet-5 architecture used for the FMNIST, SVHN, CIFAR-10, and Mix-4 datasets.

Layer	Details
layer 1	Conv2d(i=3, o=6, k=(5, 5), s=(1, 1)) ReLU() MaxPool2d(k=(2, 2))
layer 2	Conv2d(i=6, o=16, k=(5, 5), s=(1, 1)) ReLU() MaxPool2d(k=(2, 2))
layer 3	Linear(i=400 (256 for FMNIST), o=120) ReLU()
layer 4	Linear(i=120, o=84) ReLU()
layer 5	Linear(i=84, o=10 (100 for CIFAR-100, and 40 for Mix-4))

higher than 3 clusters does not improve the results as mentioned in the original paper [54]. Hence, for a fair comparison we used 2 clusters (the best performance of IFCA) in all of our experiments including the results of Tables 3.8, 3.9, and 3.10, except in 5.6 where we used both 2 and 4 clusters.

For PACFL, we used $p = 3$ for reporting the results in Tables 3.8, 3.9, and 5.6. For the results reported in Table 3.10, we used $p = 5$.

We also observed that in most cases the optimal learning rate for FedAvg was also the optimal one for the other baselines, except for Scaffold which had lower learning rates. Tables 3.17 and 3.18 provide the details of the used model architectures, where i stands for the number of input channels of conv layers and the number of input features of fc layers; o stands for the number of output channels of conv layers and the number of output features of fc layers; k stands for kernel size; s stands for stride, finally, g represents the number of groups.

Table 3.18. The details of ResNet-9 architecture used for the CIFAR-100 dataset.

Block	Details	Input
block 1	Conv2d(i=3, o=64, k=(3, 3), s=(1, 1)) GroupNorm(g=32, o=64) ReLU()	image
block 2	Conv2d(i=64, o=128, k=(3, 3), s=(1, 1)) GroupNorm(g=32, o=128) ReLU() MaxPool2d(k=(2, 2))	block 1
block 3	Conv2d(i=128, o=128, k=(3, 3), s=(1, 1)) GroupNorm(g=32, o=128) ReLU() Conv2d(i=128, o=128, k=(3, 3), s=(1, 1)) GroupNorm(g=32, o=128) ReLU()	block 2
block 4	Conv2d(i=128, o=256, k=(3, 3), s=(1, 1)) GroupNorm(g=32, o=256) ReLU() MaxPool2d(k=(2, 2))	block 2 + block 3
block 5	Conv2d(i=256, o=512, k=(3, 3), s=(1, 1)) GroupNorm(g=32, o=512) ReLU() MaxPool2d(k=(2, 2))	block 4
block 6	Conv2d(i=512, o=512, k=(3, 3), s=(1, 1)) GroupNorm(g=32, o=512) ReLU() Conv2d(i=512, o=512, k=(3, 3), s=(1, 1)) GroupNorm(g=32, o=512) ReLU()	block 5
classifier	MaxPool2d(k=(4, 4)) Linear(i=512, o=100)	block 4 + block 5

Table 3.19. The hyper-parameters used for FedAvg, FedProx, FedNova, Scaffold, and SOLO throughout the experiments

Method	Hyper-parameters	CIFAR-100	CIFAR-10	FMNIST	SVHN	Mix4
FedAvg	model	ResNet-9	LeNet-5	LeNet-5	LeNet-5	LeNet-5
	learning rate	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}
	weight decay	0	0	0	0	0
	momentum	0.9	0.9	0.9	0.9	0.9
FedProx	model	ResNet-9	LeNet-5	LeNet-5	LeNet-5	LeNet-5
	learning rate	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}
	weight decay	0	0	0	0	0
	momentum	0.9	0.9	0.9	0.9	0.9
	μ	{0.01, 0.001}	{0.01, 0.001}	{0.01, 0.001}	{0.01, 0.001}	{0.01, 0.001}
FedNova	model	ResNet-9	LeNet-5	LeNet-5	LeNet-5	LeNet-5
	learning rate	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}
	weight decay	0	0	0	0	0
	momentum	0.9	0.9	0.9	0.9	0.9
Scaffold	model	ResNet-9	LeNet-5	LeNet-5	LeNet-5	LeNet-5
	learning rate	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}
	weight decay	0	0	0	0	0
	momentum	0.9	0.9	0.9	0.9	0.9
SOLO	model	ResNet-9	LeNet-5	LeNet-5	LeNet-5	LeNet-5
	learning rate	0.01	0.01	0.01	0.01	0.01
	weight decay	0	0	0	0	0
	momentum	0.5	0.5	0.5	0.5	0.5

Table 3.20. Hyper-parameters used for LG, Per-FedAvg, IFCA, CFL, and our method throughout the experiments

Method	Hyper-parameters	CIFAR-100	CIFAR-10	FMNIST	SVHN	Mix4
LG	model	ResNet-9	LeNet-5	LeNet-5	LeNet-5	LeNet-5
	learning rate	0.01	0.01	0.01	0.01	0.01
	weight decay	0	0	0	0	0
	momentum	0.5	0.5	0.5	0.5	0.5
	number of local layers	7	3	3	3	3
	number of global layers	2	2	2	2	2
Per-FedAvg	model	ResNet-9	LeNet-5	LeNet-5	LeNet-5	LeNet-5
	learning rate	0.01	0.01	0.01	0.01	0.01
	weight decay	0	0	0	0	0
	momentum	0.5	0.5	0.5	0.5	0.5
	α	1e-2	1e-2	1e-2	1e-2	1e-2
	β	1e-3	1e-3	1e-3	1e-3	1e-3
IFCA	model	ResNet-9	LeNet-5	LeNet-5	LeNet-5	LeNet-5
	learning rate	0.01	0.01	0.01	0.01	0.01
	weight decay	0	0	0	0	0
	momentum	0.5	0.5	0.5	0.5	0.5
	number of clusters	2	2	2	2	4
CFL	model	ResNet-9	LeNet-5	LeNet-5	LeNet-5	LeNet-5
	learning rate	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}	{0.1, 0.01, 0.001}
	weight decay	0	0	0	0	0
	momentum	0.5	0.5	0.5	0.5	0.5
	ϵ_1	0.4	0.4	0.4	0.4	0.4
	ϵ_2	1.6	1.6	1.6	1.6	1.6
PACFL	model	ResNet-9	LeNet-5	LeNet-5	LeNet-5	LeNet-5
	learning rate	0.01	0.01	0.01	0.01	0.01
	weight decay	0	0	0	0	0
	momentum	0.5	0.5	0.5	0.5	0.5
	number of clusters	2	2	2	2	4
	number of $\mathbf{U}_p$	3-5	3-5	3-5	3-5	3-5

Chapter 4

Personalized Federated Learning by Structured and Unstructured Pruning

4.1 Introduction

Federated learning (FL) refers to the paradigm of learning a common objective collaboratively with the help of many clients (e.g. mobile devices or data centers) under the coordination of a central server. Such decentralized paradigms have recently drawn significant attention in the context of machine learning and deep learning as they provide several advantages such as scalability to larger datasets, data locality, ownership, and privacy, compared to traditional, centralized learning approaches [31]. In this context, a trusted server aggregates parameters optimized in a decentralized fashion by multiple clients. The resulting model is then distributed back to all clients, ultimately converging to a joint representative model without explicitly having to share the data [50].

Google has pioneered cross-device FL where the emphasis is on edge device applications [90, 91]. For instance, Google makes use of FL in the Gboard mobile keyboards, in features on Pixel phones [90], and in Android Messages [92]. Now, cross-device FL and federated data analysis are being widely applied in electronic devices, such as cross-device FL in iOS 13, and "Hey Siri" [93], etc. Nonetheless, much research has been done recently to address challenges associated with FL, including statistical challenges, the communication cost of sending large-scale matrices of parameters of deep networks [94], computing constraints, and personalization

[95].

FL faces statistical heterogeneity due to the fact that the distribution of the data across the clients is inherently non-IID [56]. In practice, it is an unrealistic assumption that the local data on each client is always IID [31]. The original goal of FL, training a single global model on the union of client datasets, becomes harder with non-IID data [96]. To address the non-IID challenge, [31] demonstrated that FedAvg can work with certain non-IID data. Other studies put efforts to alleviate the statistical heterogeneity via performing personalization in FL [97, 98, 99]. Communication cost as another major bottleneck of FL was studied in the literature. In particular, [100] proposed an aggregation method for FL, allowing a central server to perform the computation of high-dimensional data from mobile devices. [94] suggested structured and sketched updates that reduce communication costs by two orders of magnitude.

4.1.1 Federated Learning

In this Chapter, our focus is on the non-IID properties of the clients' data, as well as the critical factor of the communication cost and personalizations. A FL scenario must also take care of a myriad of practical issues including clients' datasets that change as data is added and deleted; availability of the clients, corrupted updates by the clients, etc. Studying all of these challenges is beyond the scope of the present work; instead, we propose a framework for FL to address the three key issues of statistical heterogeneity, communications cost, and personalization.

4.1.2 Contributions

In our work, we argue that the mentioned three challenges point in the same direction, i.e., we show that a single solution can be proposed to address all the challenges simultaneously. Our main contributions are:

- In the current Chapter, we consider a realistic scenario, where each client owns limited data with non-IID settings. Here we focus on FL from a client-level or personalized perspective. Motivated by the fact that, in practice, the participation of clients in the federation is

contingent upon satisfying their objectives, we leverage the statistical heterogeneity as a blessing factor and propose a new framework for personalized federated learning. In the proposed method, the clients are not only a source of data and model training for the global server, but also can ameliorate the performance on their personalized target distributions.

- Other than that, our underlying method is a straightforward yet effective solution that further alleviates the communication bottleneck as well in FL by finding a subnetwork for each client by a novel hybrid (combination of structured and unstructured) and unstructured pruning strategy on the neural network models of the clients. To the best of our knowledge, there has not been any proposed method in FL to address the mentioned critical challenges simultaneously.

4.2 Related Work

In FL, the edge devices carry most of the load of computation and a central server updates the model parameters using the descending directions returned by the edge devices. However, FL has three unique characteristics that make it different from parallel optimization systems in the following aspects:

Statistical Heterogeneity The main motivation for clients to take part in FL is to improve their own model performance. Especially, clients who have very limited private data benefit the most from collaboratively learned models. However, for some of the clients who have enough private data, there is not much benefit to participating in FL. This issue becomes even worse in the case of statistical heterogeneity of the clients. In fact, due to the non-IID distribution of data across devices, it leads to scenarios where some participants may gain no benefit by participating in FL since the global shared model is less accurate than the local models that they can train on their own [95, 101].

Communication Efficiency A naive implementation of FL framework entails each client sending a full model update back to the central server in each communication round. For large neural networks, this step will be the bottleneck of FL due to the asymmetric nature of internet connection speeds: the uplink is typically much slower than the downlink. The US average internet speed was 55Mbps download vs. 18.9Mbps upload, and with some internet service providers, this issue is worse, e.g., Xfinity provides 125Mbps down vs. 15Mbps up [102]. Hence, it is important to propose methods that necessitate less uplink communication cost [31, 103, 104, 105, 69, 106]. For instance, some existing models can reduce the communication cost with structured updates, and sketched updates [94]. Others do so by compressing the gradients [105].

Personalizations and Accuracy for Individual Clients FL is explicitly designed for non-IID clients, but most of the prior works measured global accuracy and not accuracy for individual clients. A global model can perform well on personalized predictions if the client's context and personal data are nicely featured and embodied in the dataset, which is not the case in most clients [107]. Most techniques for personalization either affect privacy or involve two separate steps where a global model is constituted collaboratively in the first step, and then the global model is personalized for each client using the client's private data in the second step. These two steps might add extra computational overhead [97, 95, 108].

4.3 Method

Deep neural networks (DNN), especially deep Convolutional Neural Networks (CNN), have achieved significant success in various tasks and applications. However, the excellent performance of modern CNNs comes often at significant inference costs due to more stacked layers, and thus more learnable parameters. The usage of these high capacity networks may be largely hindered for FL scenarios where in addition to accuracy, computational efficiency

and small network sizes are crucial enabling factors. For example, a ResNet-152 has more than 60 million parameters and entails more than $20G$ float-point-operations (FLOPs) when training with an image with resolution 224×224 [30]. This is unlikely to be affordable on resource constrained platforms such as embedded edge devices [109].

The recent strategies of ameliorating CNN efficiency mostly focus on compressing models and accelerating inference without significantly sacrificing their accuracy performance. Among adopted methods, progressive pruning appears to be an outstanding one where a deep neural net is trained, then pruned, and then fine tuned to restore performance [110].

4.3.1 Efficient Learning with Pruning

In this Chapter, we show that in an FL scenario, there consistently exist smaller similar subnetworks for clients with similar labels of data that can improve the accuracy of each other in FL. Based on these results, we state the following observation.

Client Subnetwork Observation: *Through extensive experiments on various DNNs and benchmark datasets, we noticed the existence of similar subnetworks for clients with even partially similar data (labels) and the cheap costs needed to reliably find them, and develop a new algorithm to improve their accuracies.*

More specifically, consider a dense feed-forward neural network model $f(x;\theta)$ with initial parameters $\theta = \theta_0 \sim D_\theta$ for each client as well as the central server. Further, assume that optimizing client k's neural net with stochastic gradient descent (SGD) on its own training set, $f(x;\theta_k)$, reaches minimum validation loss L_k at iteration j with test accuracy Acc_k. Besides, consider training $f(x;m_k \odot \theta_k)$ with a mask $m \in \{0,1\}^{|\theta|}$ on client k's parameters reaches minimum validation loss L'_k when being optimized with SGD on the same training set at the j' iteration of training with a test accuracy of Acc'_k. We observed that under some scheduling $Acc_k \leq Acc'_k$ which means the improvement of the accuracy of each client k while the p-percentage of each client's network is pruned. The following sections delineate the conceptual overview of accuracy improvement in our proposed FL method.

4.3.2 Efficient Learning with Pruning

Consider a dense feed-forward neural network model $f(x;\theta)$ with initial parameters $\theta = \theta_0 \sim D_\theta$ for each client as well as the central server. Further, assume that optimizing client k's neural net with stochastic gradient descent (SGD) on its own training set, $f(x;\theta_k)$, reaches minimum validation loss L_k at iteration j with test accuracy Acc_k. Besides, consider training $f(x;m_k \odot \theta_k)$ with a mask $m \in \{0,1\}^{|\theta|}$ on client k's parameters reaches minimum validation loss L'_k when being optimized with SGD on the same training set at the j' iteration of training with a test accuracy of Acc'_k. We observed that under some scheduling $Acc_k \leq Acc'_k$ which means the improvement of the accuracy of each client k while the p-percentage of each client's network is pruned. The following sections delineate the conceptual overview of accuracy improvement in our proposed FL method.

4.3.3 Structured, Unstructured, and Hybrid Pruning

In this Chapter, we aim to find a smaller subnetwork for each client in the FL scenario via three different pruning levels, i.e., channel level pruning (structured), parameter level pruning (unstructured), and hybrid (combination of structured and unstructured) level pruning in deep CNNs. Unstructured-level pruning is motivated by the fact that it gives the highest flexibility and generality for compression rate [111] for both shallow and deep neural networks. On the other hand, channel-level pruning is less flexible as some whole layers need to be pruned. Pruning a whole layer is only effective when the neural network is sufficiently deep [112]. Nonetheless, channel-level pruning provides a fair trade-off between flexibility and ease of implementation. There is also a hybrid level pruning which is realized by the combination of channel and parameter level pruning. It can be applied to any typical CNNs or fully connected networks, which results in a compressed network that can be efficiently inferenced on conventional CNN platforms.

4.3.4 Why Learning with Pruning in FL is Efficient

We consider synchronized algorithms for FL, and we suggest a round consisting of the following steps: *i*) A subset of existing clients is randomly selected, each of which downloads the current model parameters from the server. *ii*) Each client in the subset computes an updated model based on its local data and prunes its neural network according to Algorithm 1 or 2 to obtain its own subnetwork. *iii*) The model updates are sent from the selected clients to the server. *iv*) The server aggregates these models by applying the Sub-FedAvg method where the average is taken only on the intersection of the remaining channels (in structured pruning) or the remaining parameters (in unstructured pruning) of each client to construct an improved global model.

In what follows, we will first provide a conceptual overview of this method of pruning and the proposed Sub-FedAvg on the server by the following remark and then delineate the routine of the algorithm, and finally, show the evaluation performance by benchmarking it with state-of-the-art methods of FL in DNN models based on representative datasets.

Remark-1: At the beginning of each communication round, each client downloads the model from the server which contains the featured and embodied data of all clients, and starts training on its local data. Due to the statistical heterogeneity of clients, part of the channels (filters) and parameters are personalized to each client. By iteratively pruning the parameters and channels, we remove the commonly shared parameters of each layer and keep the personalized parameters that can represent the features of local data in each client. Since the clients with similar data (label overlap) share similar personalized parameters, by the proposed Sub-FedAvg we would average the models of clients on the server on the intersection of the remaining channels and parameters of the clients. This method of aggregation on the server not only does not impact the accuracy performance of each client in a non-IID setting but also improves it significantly as reported in Table 4.3.

4.3.5 Algorithm

Unstructured Pruning At each communication round, the server randomly samples a set of clients $\mathscr{S}$ and sends the model to the selected clients. Each client C_k starts training the local model. Given a target pruning ratio p and a pruning percentage, r_{us}, for each communication round, a binary mask is derived at the end of the first epoch, and at the end of the last epoch w.r.t. the full dense network of client C_k where assigns 0 to the lowest r_{us}-percent of the absolute value of parameters and 1 to the rest. To this end, after training, each client tests its model on the validation data D_k^{val}. If the validation accuracy is above a pre-considered threshold Acc_{th} and if the target pruning rate is not achieved yet and finally, if the Hamming distance between the two masks (mask distance) is above a pre-defined threshold ε, the client C_k prunes its model with the mask obtained at the end of the last epoch. This process is iterated in all communication rounds till the conditions are not satisfied.

Structured Pruning We adopt the same channel pruning as in [113] since it is hardware friendly and aligns best with our mission of efficient training and performance improvement. Herein, we do the same process as in unstructured pruning and follow [113] to consider the scaling factor r in batch normalization (BN) layers as indicators of the corresponding importance of channels. We derive a mask for each filter individually and channels of filters are pruned based on the pruning threshold. For simplicity, the pruning threshold is determined by a percentile among all scaling factors, e.g., $p\%$ of channels are pruned.

Hybrid Pruning Structured pruning is more effective when the depth of the neural network of clients is sufficiently large [114]. On the other hand, due to the benefit of structured pruning in enabling the model with structured sparsity and more efficient memory usage, and more computation acceleration, we would get the benefit of it even in the small networks by leveraging the combination of structured and unstructured pruning, namely hybrid pruning. In this method,

we apply structured pruning on the filters and employ unstructured pruning on the fully connected layers. Algorithm 1 and 2 summarize our methodology. *It is noteworthy that in Algorithm 2, the structured and unstructured pruning processes are independent of each other meaning that when one does satisfy the constraints it applies the mask regardless of if the other one satisfies the constraints or not.* Moreover, the way that the parameters are aggregated on the server is called Sub-FedAvg in this Chapter.

Algorithm 10. Sub-FedAvg with unstructured pruning (Sub-FedAvg (Un))

Server: initialize the server model with θ_g.
Require: $k \leftarrow \max(K \times N)$: N number of available clients, sampling rate K
$\mathscr{S}_t \leftarrow$ (random set of k clients) **for** *each round j = 1, 2,...* **do**
  for *each client* $k \in \mathscr{S}_j$ **in parallel do**
    download θ_g^j from the server and start training; set the pruning ratio r_{us} and target pruning rate p_{us}

    derive the mask $m_k^{j,fe}$ from $\theta_k^{j,fe}$ at the end of first epoch (fe)
    derive the mask $m_k^{j,le}$ from $\theta_k^{j,le}$ at the end of last epoch (le)

    $\theta_k^{j+1} \leftarrow \text{ClientUpdate}(C_k; \theta_k^j; \theta_0)$: using SGD training
  end
  $\theta_g^{j+1} \leftarrow$ aggregate subnetworks of clients, θ_k^{j+1}, and take the avg on the intersection of unpruned parameters (Sub-FedAvg).
end
ClientUpdate$(C_k; \theta_k^j; \theta_0)$:
evaluate θ_k^j on the local validation data D_k^{val} and report the accuracy.
$\Delta_{us} \leftarrow$ Mask Distance of $|m_k^{j,fe} - m_k^{j,le}|$
if accuracy $\geq Acc_{th}$ & target pruning rate p is not achieved yet & $\Delta_{us} \geq \varepsilon_{us}$ **then**
  $\theta_k^{j+1} = \theta_k^{j,le} \odot m_k^{j,le}$: apply the mask
end
return θ_k^{j+1} to server

4.4 Experiments

In this section, we provide extensive experimental results evaluating our proposed algorithms.

4.4.1 Experiment Settings

Datasets and Non-IID Partitions We use MNIST [115], CIFAR-10 [116], EMNIST [117], and CIFAR-100 datasets in our experiments. To produce non-IID partitions, we partition all the training dataset into shards of 250 examples (except for CIFAR-100 where we use 125 examples) and randomly assign two shards to each client. Evaluation data for each client is all the test set for the training dataset labels they have.

Architecture The architecture we used for MNIST, and EMNIST datasets is a 5-layer CNN consisting of two layers of 5×5 convolutional layer with 10 and 20 channels respectively, and each convolutional layer is followed by batch-normalization and 2×2 max pooling layers. Finally, a fully-connected layer with 50 units followed by another fully-connected layer with 10 units to produce the logits (30900 total parameters + 30 channels). For all layers, except the last layer, we use the ReLU non-linear activation function. For CIFAR-10 and CIFAR-100 datasets we use LeNet-5 (62000 total parameters + 16 channels) architecture [118]. In the LeNet-5 architecture, we also add a batch-normalization layer after each convolutional layer.

Hyper-parameter Setting For all experiments, we setup 100 clients with local batch size 10, local epoch 5, and an SGD optimizer with learning rate and momentum of 0.01 and 0.5, respectively. Further, the threshold for distance of masks are 10^{-4}, and 0.05 for unstructured and, hybrid pruning algorithms, respectively.

4.4.2 Main Results

We train LeNet-5 architecture on CIFAR-10/100, MNIST, and EMNIST datasets in a non-IID data distribution setting. We compare the results of our proposed algorithms against the state-of-the-art baselines. Table 4.1 and 4.2 report the results for the average accuracy across all clients, communication cost, and the FLOP and parameters reduction. We first focus on the

Table 4.1. Comparing the performance metrics of our results for the proposed algorithms, Sub-FedAvg (Hy), and Sub-FedAvg (Un) against the state-of-the-arts.

Setting	Algorithm	Accuracy	Pruned % : %Hyprid.	Unstructured: % Param.	Commun. Cost
LeNet-5 CIFAR-10	Standalone	84.44%	0	0	0
	FedAvg	58.99%	0	0	2.48 GB
	MTL	49.87%	0	0	16.12 GB
	LG-FedAvg	76.28%	0	0	2.27 GB
	Sub-FedAvg (Un)	**86.01%**	-	30%	2.12 GB
	Sub-FedAvg (Un)	**84.44%**	-	50%	1.88 GB
	Sub-FedAvg (Un)	**83.6%**	-	70%	1.64 GB
	Sub-FedAvg (Hy)	**83.21%**	50%	47.26%	1.89 GB
	Sub-FedAvg (Hy)	**82.86%**	73%	70%	1.62 GB
	Sub-FedAvg (Hy)	**82.5%**	92%	90%	1.39 GB
LeNet-5 MNIST	Standalone	94.25%	0	0	0
	FedAvg	96.9%	0	0	524.16 MB
	MTL	99.74	0	0	3407.04 MB
	FedProx	97.9%	0	0	1572.48 MB
	LG-FedAvg	98.2%	0	0	513.6 MB
	Sub-FedAvg (Un)	**99.43%**	-	30%	448 MB
	Sub-FedAvg (Un)	**99.28%**	-	50%	397.21 MB
	Sub-FedAvg (Un)	**99.35%**	-	70%	346.43 MB
	Sub-FedAvg (Hy)	**99.57%**	50%	49%	383.39 MB
	Sub-FedAvg (Hy)	**99.54%**	71%	70%	342.31 MB
	Sub-FedAvg (Hy)	**97.46%**	90%	95%	293.40 MB

accuracy comparison against the baselines. The accuracy is measured for both cases when the subnetworks are drawn by unstructured pruning and hybrid pruning. For unstructured pruning, we evaluate the accuracy when 30%, 50%, and 70% of the parameters are pruned. This is while for the hybrid pruning we report the accuracy results when 50%, 70%, and 90% of the parameters are pruned.

We can clearly see the advantage of the proposed Sub-FedAvg (Hy) and Sub-FedAvg (Un) algorithms over the state-of-the-arts in improving the accuracy performance of clients. This is realized due to the fact that by iterative pruning, we let each client find its own partners (clients

Table 4.2. Comparing the performance metrics of our results for the proposed algorithms, Sub-FedAvg (Hy), and Sub-FedAvg (Un) against the state-of-the-arts.

Setting	Algorithm	Accuracy	Pruned % : %Hyprid.	Unstructured: % Param.	Commun. Cost
LeNet-5 EMNIST	Standalone	98.59%	0	0	0
	FedAvg	88.81%	0	0	524.16 MB
	MTL	98.57	0	0	3407.04 MB
	LG-FedAvg	98.93%	0	0	513.6 MB
	Sub-FedAvg (Un)	**99.11%**	-	30%	448 MB
	Sub-FedAvg (Un)	**99.16%**	-	50%	397.21 MB
	Sub-FedAvg (Un)	**97.71%**	-	70%	346.43 MB
	Sub-FedAvg (Hy)	**99.47%**	50%	42%	397.08 MB
	Sub-FedAvg (Hy)	**99.45%**	70%	69%	344.26 MB
	Sub-FedAvg (Hy)	**98.56%**	90%	93%	297.32 MB
LeNet-5 CIFAR-100	Standalone	80.56%	0	0	0
	FedAvg	10.4%	0	0	2.78 GB
	MTL	43.86%	0	0	18 GB
	LG-FedAvg	47.6%	0	0	2.58 GB
	Sub-FedAvg (Un)	**85.5%**	-	30%	2.38 GB
	Sub-FedAvg (Un)	**83.40%**	-	50%	2.11 GB
	Sub-FedAvg (Un)	**83.74%**	-	70%	1.84 GB
	Sub-FedAvg (Hy)	**82.16%**	50%	49%	2.12 GB
	Sub-FedAvg (Hy)	**82.06%**	70%	69%	1.82 GB
	Sub-FedAvg (Hy)	**80.80%**	88%	88%	1.56 GB

with similar labels) and leverage their personalized parameters to ameliorate their accuracy performance. In order to reveal how effective the proposed method is, we also compare the results versus two benchmarks, i.e., the *Standalone*, and *traditional FedAvg*. In the former, each client trains a model locally only by its own local data without federation. In the latter, all clients participate in the federation and the server averages the parameters traditionally by FedAvg. These results validate the effectiveness of our proposed algorithms for the objectives presented in the following Remarks:

Remark-2: In the low data regime, which is the case in most FL scenarios, where the clients

are the edge devices, by structured/unstructured pruning (finding subnetworks) we get rid of the common parameters and keep the personalized ones to let the clients take the advantage of the featured data of the rest of clients with similar labels. This yields around 20% accuracy improvements versus the *Standalone* benchmark which further motivates federation. Compared to the traditional FedAvg benchmark, our results demonstrate around 30% improvement. As is evident from the table, in the low data non-IID settings, the traditional FedAvg performs worse than the *Standalone*, which means that it is not justified for clients to participate in a federation under this setting.

This phenomenon can be explained by the fact that depending upon the data (labels) of each client, by our pruning strategy, which prunes the common parameters and keeps the personal ones, we let the clients find their subnetwork. Through comprehensive experiments, we noticed that it is likely that the clients with label overlap share the same personal parameters. With that, we help all clients to find their corresponding partners in the federation and do the parameter averaging by Sub-FedAvg. With that in mind, the poor performance of the traditional FedAvg in a non-IID environment can be understood.

Remark-3: Our proposed model also contributes to producing a more compressed model which achieves inference speedup compared to the original model. As can be seen from the table, our proposed model highly reduced the number of FLOPs up to 2.4$\times$ compared to the state-of-the-art. This is a crucial metric especially when the clients are edge devices with computational limitations.

Accuracy versus pruning percentage

We have provided experiments on demonstrating how our algorithm can improve the accuracy of the clients. Fig. 4.1 plots the test accuracy result of some sampled clients versus various pruning percentages when iteratively pruning by 5%-10% per iteration for CIFAR-10 on LeNet-5. Also, in Fig. 4.2 we sketched the average test accuracy over all clients versus various

average pruning percentages over all clients. As expected, as the pruning target increases up to a certain level of sparsity, according to Remark-1 and Remark-2, the test accuracy rises as compared to the original network. According to our experiments, almost 85% of the clients have the same pattern of results. As expected, after some pruning rate (around 50%) we observe the accuracy degradation which is due to the fact that by further pruning we are removing the personal parameters as well.

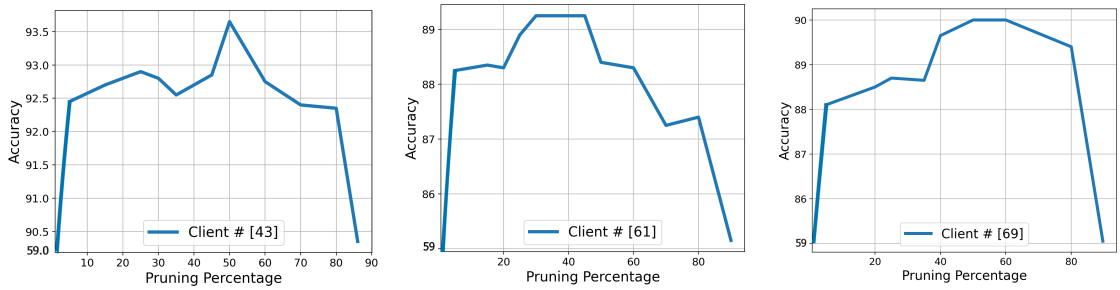


Figure 4.1. Test accuracy vs pruning percentage result of the proposed Sub-FedAvg (Un) on some sampled clients on LeNet-5 for CIFAR-10.

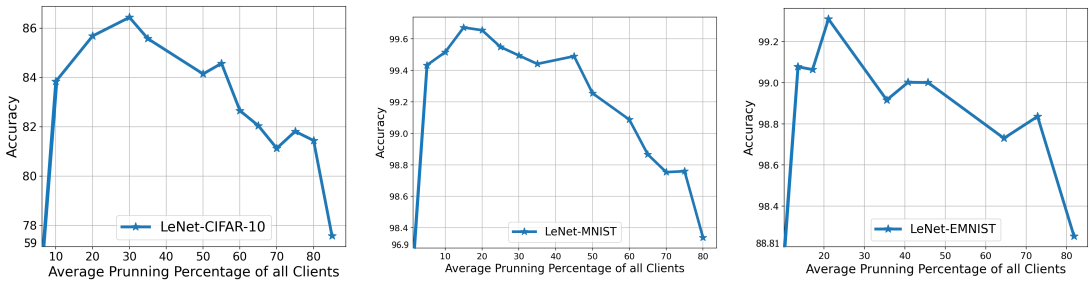


Figure 4.2. Average test accuracy result of the proposed Sub-FedAvg (Un) versus various average pruning percentages over all clients, on LeNet-5 for CIFAR-10, MNIST, and EMNIST benchmarks.

Communication Efficiency

In FL, communication cost is an important bottleneck since the edge devices are typically constrained by an upload bandwidth of 1 MB/s or less. The results in Table 4.3 clearly demonstrate that the proposed Sub-FedAvg (Hy) and Sub-FedAvg (Un) methods provide significant performance improvement in terms of communication cost across prior benchmarks

reported in the table. It is noteworthy that we obtained the communication cost for all of the baselines as $Cost = R \times B \times |W| \times 2$, where, R stands for the total number of communication rounds; B denotes the number of bits based on single-precision floating point (32 bit for floating numbers and 1 bit for integers 0 and 1); and $|W|$ represents the total number of model parameters that are exchanged between each client and the server in each round. Unlike prior works [119] where the communication cost is addressed by only sharing a subset of the parameters during each round of communication by sacrificing the performance, our proposed model can produce a dramatic decrease in communication cost without sacrificing the accuracy due to the following facts:

Firstly, our model prunes common parameters without impacting the accuracy of clients and keeps the personalized ones as mentioned in Remark-2. Hence, the server collects an efficient aggregation of the clients' updates by only collecting the kept ones. Secondly, by the proposed Sub-FedAvg (Hy) and Sub-FedAvg (Un) we end up with getting a compressed version of the neural networks which work efficiently and converge faster. In our experiments, for the Sub-FedAvg (Un) method, we got the reported results for CIFAR-10/100, MNIST, and EMNIST datasets by training the neural networks only for 500, 300, and 300 communication rounds, respectively. This is while other baselines typically perform around 1000 rounds of communication to converge to their desired accuracy performance. This fast convergence along with the mentioned fact, contributed in the reduction of required communication rounds by up to $10\times$ compared to the prior works without accuracy degradation. Fig 4.3 shows the comparison of our method and the state-of-the-art in terms of accuracy versus the number of communication rounds. We achieved around 99.5%, 99.3%, and 86% test accuracy for MNIST, EMNIST, and CIFAR-10 datasets, respectively, after around 100 rounds, while the traditional FedAvg, LG-FedAvg, and MTL achieve a lower test accuracy after 100-800 communication rounds. We observe that Sub-FedAvg (Un) achieves a significant reduction in required communication round by 2-10 times in all cases. For CIFAR-10, FedAvg requires the highest number of communication rounds to obtain the target accuracy, which is about 3 times more than that of our method.

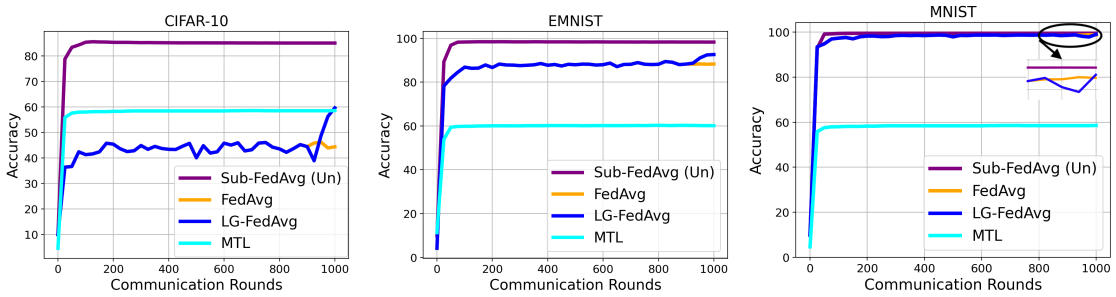


Figure 4.3. Test accuracy vs. communication rounds for the CIFAR-10, EMNIST, and MNIST experiments under statistical heterogeneity.

Flop Reduction

To further highlight our proposed method's efficiency we delve into the percentage of parameter saving and FLOP saving. In speedup analysis, operations such as batch normalization (BN) and pooling are ignorable compared to convolution operations. As such, we count the FLOPs of convolution operations for computation complexity analysis, which is a common trend in prior works [113]. In our experiments, for example, for LeNet-5, 50% of channels are pruned which results in around 50% FLOP reduction while the parameter saving is around 38%. This is due to the fact that for example in LeNet-5 only 11 (out of 22) channels from all the computation-intensive convolutional layers are pruned, while 24k parameters (out of 49k) from the parameter-intensive fully-connected layers are pruned. The results show that our methods not only work efficiently with much fewer filters and channels, and save a significant number of FLOPs and accelerate the inference but also leverage it for accuracy improvement on CIFAR-10/100, MNIST, and EMNIST datasets.

4.5 Discussion and Conclusion

A new framework for personalized federated learning with Non-IID data distributions was proposed. The method works by iteratively pruning the parameters and channels of the neural networks which results in removing the commonly shared parameters of clients' model and keeping the personalized ones. In contrast to other approaches, the proposed framework is also

Table 4.3. Comparing flop, and parameter reduction results for the proposed algorithms, Sub-FedAvg (Hy), and Sub-FedAvg (Un) against the state-of-the-arts on CIFAR-10, MNIST, EMNIST, and CIFAR-100 datasets.

Algorithm	Flop, Parameter Reduction
Standalone	$0\times$
FedAvg ([31])	$0\times$
MTL ([58])	$0\times$
LG-FedAvg ([52])	$0\times$
Sub-FedAvg (Un), $p_{us} = 30$	$0\times, 0.3\times$
Sub-FedAvg (Un), $p_{us} = 50$	$0\times, 0.5\times$
Sub-FedAvg (Un), $p_{us} = 70$	$0\times, 0.7\times$
Sub-FedAvg (Hy), $p_s = 50$	$2.4\times, 0.5\times$
Sub-FedAvg (Hy), $p_s = 70$	$2.4\times, 0.7\times$
Sub-FedAvg (Hy), $p_s = 90$	$2.4\times, 0.9\times$

efficient in terms of communication cost and FLOP count. We found that our method outperforms the state-of-the-art algorithms on CIFAR-10/100, MNIST, and EMNIST benchmarks.

Algorithm 11. Sub-FedAvg with hybrid pruning (Sub-FedAvg (Hy))

Server: initialize the server (θ_g) and clients (θ_0). **Clients:** set the scaling factor r, the pruning ratios r_{us} and r_s, the target pruning rates p_{us}, and p_s; the mask distances ε_s, and ε_{us}.

Require: $k \leftarrow \max(K \times N)$: N number of available clients, sampling rate K

$\mathscr{S}_t \leftarrow$ (random set of k clients)

for *each round j = 1, 2,...* **do**

  for *each client* $k \in \mathscr{S}_j$ **in parallel do**

    download θ_g^j from the server and start training.

    derive the mask by structured pruning based on r_s towards the target ratio p_s, at the end of first epoch $(m_k^{(1)})$ and last epoch $(m_k^{(2)})$.

    derive the mask $m_k^{j,fe}$ from $\theta_k^{j,fe}$ by unstructured pruning on the fc-layers at the end of the first epoch (fe)
    and derive the mask $m_k^{j,le}$ from $\theta_k^{j,le}$ by unstructured pruning on the fc-layers at the end of the last epoch (le)

    $\theta_k^{j+1} \leftarrow \text{ClientUpdate}(C_k; \theta_k^j; \theta_0)$: using SGD training.

  end

  $\theta_g^{j+1} \leftarrow$ aggregate subnetworks of clients θ_k^{j+1}

end

ClientUpdate$(C_k; \theta_k^j; \theta_0)$:

evaluate θ_k^j on the local validation data D_k^{val} and report the accuracy.

$\Delta_s \leftarrow$ Mask Distance $|m_k^{(1)} - m_k^{(2)}|$

$\Delta_{us} \leftarrow$ Mask Distance of $|m_k^{j,fe} - m_k^{j,le}|$

if accuracy $\geq Acc_{th}$ & target pruning rate p_s, p_{us} are not achieved yet: **then**

  if any of the conditions $\Delta_s \geq \varepsilon$ or $\Delta_{us} \geq \varepsilon$ hold: **then**

    apply its corresponding mask

    $\theta_k^{j+1} = \theta_k^{j,le} \odot m_k^{(2)} \odot m_k^{j,le}$

  end

end

return θ_k^{j+1} to server

Chapter 5

New Notion and Standard Benchmarks for Data Heterogeneity in Federated Learning

5.1 Introduction

Deep learning has emerged as a fast development technique in computer vision, natural language processing, and conversational AI. Though successful, the efficacy of machine learning and deep learning algorithms relies on large quantities of data. However, in areas such as health care, the data may be distributed across numerous hospitals or data centers and cannot be accessed by a central server or cloud due to privacy constraints. For instance, hospitals may have only a few of images of particular cancer which must be kept private. A lot of works on privacy preserving data management and data mining [120] in a centralized setting have been proposed so far, however, they cannot tackle the cases of distributed databases. Driven by such realistic requirements, in order to conduct data mining/machine learning, it is necessary to exploit data from such distributed databases while preserving data privacy. Federated learning (FL) [50] rises to this challenge due to its ability to collectively train neural networks while preserving data privacy. One of the standard FL algorithms is FedAvg [50]. In each round of FedAvg, clients train models with their local datasets independently, and then the server aggregates the locally trained models, and finally, the parameters of local models are averaged element-wise by the server to obtain a shared *Global* model [31]. FL however introduces distinct issues not present in classical distributed learning [2]. One of the challenges that currently confine the applicability

of existing FL methods to real-world datasets is the data heterogeneity in the data distribution between participating clients [51, 35].

There have been a plethora of works proposing solutions to FL under Non-IID data in recent years. They can be categorized into four groups: 1) alleviating non-guaranteed and weight divergence [121, 70, 32, 33, 122, 123], where the local objectives of the clients is modified such that the local model is consistent with the global model to some extent; 2) Modifying the aggregation scheme at the server side [124, 125, 34, 126, 122]; 3) data sharing [56, 127, 128, 129], where the server shares a small subset of an auxiliary dataset with the clients to help construct a more balanced and IID data distribution on the client; personalized federated learning [44, 36, 51, 52, 54, 55, 53], take the advantage of a certain level of personalization in training the individual clients models rather than training a single global model.

After aggregating results across 45 papers addressing data heterogeneity in FL, we believe the right approach to tackling data heterogeneity is a highly non-trivial question on which the FL community has barely scratched the surface. In particular, the challenge comes from various facets, including but not limited to:

1. The FL community lacks a true notion of data heterogeneity without which the provided solutions may not be effective.

2. The community suffers from a lack of standardized benchmarks on which all proposed algorithms to be compared. There have been several heterogeneity benchmarks including Non-IID (2), Non-IID (1), Dir(.), and rotated datasets [54, 130, 69] and even more to say. Firstly, not all proposed algorithms compared their method with others on a unique Non-IID setup and secondly, we will show in Section 5.2.3, and 5.3.3 that the above-mentioned Non-IID setups are more like IID.

3. It is still not well understood in the community whether and under which conditions clients benefit from collaboration under a data heterogeneity setting.

To address this situation, we identify issues with current practices, and suggest concrete remedies by defining a new notion of data heterogeneity framework in FL which further facilitates standardized evaluations and comparison of methods. Through extensive studies, we have several key findings:

- It is not clear how the existing FL algorithms tackle the data heterogeneity while they lack a systematic understanding of the data heterogeneity.
- Many of the prior works have emphasized that the statistical data heterogeneity in FL has harmful effects and can lead to poor convergence [44, 51, 131] which necessitate personalization [131, 70]. In contrast, we found that the current data partitioning strategies may not necessarily bring significant challenges in learning the accuracy of FL algorithms. Refer to Sections 5.2.2, 5.2.3, 5.3.3, and 5.3.4.
- Under the new notion of heterogeneity that will be proposed herein, data heterogeneity can have detrimental effects such that for some of the clients it is not justified to participate in federation. Refer to Table 3.8 and Section 5.5.
- None of the existing state-of-the-art (SOTA) FL algorithms beats the others according to the new notion of data heterogeneity that will be presented in this Chapter. Refer to Table 3.8.

5.1.1 Current Non-IID Setups

Current practices [31, 34, 69, 44, 56, 36, 54, 130] have very rigid data partitioning strategies among parties, which are hardly representative and thorough. In the experiments of existing studies, data heterogeneity has been simply modeled as Non-IID label skew (20%), Non-IID label skew (30%), and Dir(α), or has been generated by augmenting the datasets using rotation [54].

For Non-IID label skew (20%) and (30%), 20% and 30% of the total classes in a dataset is randomly assigned to each client, respectively [36]. Then, the samples of each class is

randomly and equally partitioned and distributed amongst the clients who own that particular class. For Non-IID Dir(α), we get random samples for class c from Dirichlet distribution according to $p_c \sim \text{Dir}(\alpha)$ and give each client j random samples of class c according to $p_{c,j}$ proportion. In this setup, heterogeneity can be controlled by the parameter α of Dirichlet distribution [130, 125, 56, 35, 69].

These partitioning strategies cannot design a real and comprehensive view of Non-IID data distribution. *As will be delineated later on, the above-mentioned Non-IID partitions that the prior algorithms have been tested on are more like an IID partition because the data distributions of clients are the sub-distributions of a unique dataset such as CIFAR-10. Besides, all clients have a high percentage of label overlap which mimics IID.* That's why it is a common belief that users can benefit from heterogeneity in the federation. While in practice, the union of the clients data may not be only one dataset. For instance, in mobile phones, or recommendation systems, clients may own very different categories of images like animals, celebrities, nature, and paintings; advertisement platforms might need to send different categories of ad to the customers. Therefore, due to the small intra-class distance (similarity between distribution of the classes) in the used benchmark datasets, all baselines benefited highly from the federation. This is the reason that heterogeneity has never been a challenge. More discussion on this will be provided in the rest of the Chapter. We break the barrier of experiments on Non-IID data distribution challenges in FL by proposing a new look into data heterogeneity. This approach addresses a broad range of data heterogeneity issues beyond simpler forms of Non-IIDness like label skews. Here we formally introduce our proposed paradigm, where the goal is to define a new notion of data heterogeneity and suggest standard and real Non-IID setups. We hope that this notion, along with introduced setups, be standard and inspire the federated learning community.

5.2 Overview

5.2.1 Methodology

In our approach, the data heterogeneity/homogeneity should be identified by analyzing the principal angles between the client data subspaces. More particularly, each client in FL applies a truncated SVD step on its own local data in a single-shot manner to derive a small set of principal vectors, which form the principal bases of the underlying data. These principal bases provide a signature that succinctly captures the main characteristics of the underlying distribution. These principal bases efficiently identify distribution heterogeneity/homogeneity among clients by comparing the principal angles between the clients data subspaces spanned by the provided principal vectors. The greater the difference in data heterogeneity between two clients is, the more orthogonal their subspaces will be.

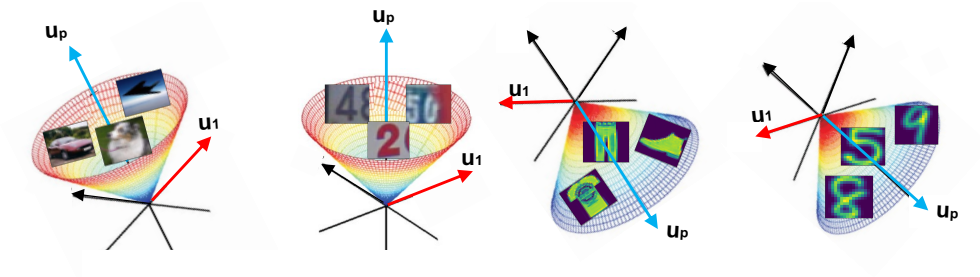


Figure 5.1. There must be a translation protocol enabling the server to understand similarity and dissimilarity in the distribution of data across clients without sharing data. These 2D figures intuitively demonstrate how the principal angle between the client data subspaces captures the statistical heterogeneity. In particular, it shows the subspaces spanned by the $\mathbf{U}_p$s of four different datasets (left to right: CIFAR-10, SVHN, FMNIST, and USPS). As can be seen the principal angle between the corresponding $\mathbf{u}$ vectors of CIFAR-10 and SVHN is smaller than that of CIFAR-10 and USPS. Table I shows the exact principal angles between every pair of these subspaces.

To measure the statistical heterogeneity among different clients' domains, in this Chapter, we leverage the angle between clients data subspaces spanned by the most significant left singular vectors of clients data. To begin, we introduce the data heterogeneity via the new notion presented in this Chapter and then we will generate Non-IID data partitioning across the clients using the proposed method. For a dataset, $\mathbf{D}$, we put the data of each class C_i

in the columns of its corresponding matrix $\mathbf{Q}_i$. We then, perform truncated SVD on $\mathbf{Q}_i$ and obtain $\mathbf{U}_p^i = [\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_p]$, $(p \ll \text{rank}(\mathbf{D}_\text{k}))$. These $\mathbf{u}$ vectors span the class subspace and provide a useful signature for distinguishing the underlying distributions of each class in $\mathbf{D}$ since these principal bases characterize the main trends in the data of clients (like eigenfaces). Then according to the principal angle in between of the class subspaces, we understand how similar/dissimilar two classes are based on which we can generate Non-IID data partitioning. *The more orthogonal two subspaces are the more heterogeneous the data of two classes will be.*

Having the data *signature* of all classes of dataset $\mathbf{D}$, in hand, we can obtain a proximity matrix $\mathbf{A}$ either as in Eq. (5.1) whose entries are the smallest principle angle between the pairs of $\mathbf{U}_p^i$ or as in Eq. (5.2) whose entries are the summation over the angle in between of the corresponding $\mathbf{u}$ vectors (in identical order) in each pair of $\mathbf{U}_p^i$ (where $\mathbf{tr}(.)$ is the trace operator).

$$\mathbf{A}_{i,j} = \Theta_1\left(\mathbf{U}_p^i, \mathbf{U}_p^j\right), \;\; i,j = 1, ..., |\mathscr{C}| \tag{5.1}$$

$$\mathbf{A}_{i,j} = \mathbf{tr}\left(\arccos\left(\mathbf{U}_p^i * \mathbf{U}_p^j\right)\right), \;\; i,j = 1, ..., |\mathscr{C}| \tag{5.2}$$

where $\mathscr{C}$ is the total number of classes of $\mathbf{D}$. *Either of Eq. 5.1 and Eq. 5.2 can be employed in practice, however, theoretically Eq. 3.2 is more rigorous.* Now, in order to capture the similarity/dissimilarity of different classes of $\mathbf{D}$, we could form disjoint clusters of classes. For forming disjoint clusters, we can perform agglomerative hierarchical clustering [68] on the proximity matrix $\mathbf{A}$. The best number of clusters can easily be determined just by analyzing the proximity matrix. Each cluster contain classes which are roughly identically distributed.

Before providing more details about the application of the proposed method in defining new Non-IID partition, we elucidate how the proposed method perfectly distinguishes different datasets based on their hidden data distribution by inspecting the angle between their data subspaces spanned by their first p left singular vectors. For a visual illustration of the result, we refer to Fig. 5.1, Fig. 5.2, and Table 5.1, where the similarity and dissimilarity of the four different datasets have been evaluated by the proposed measure in Eq. 5.1, and Eq. 5.2. Table I further

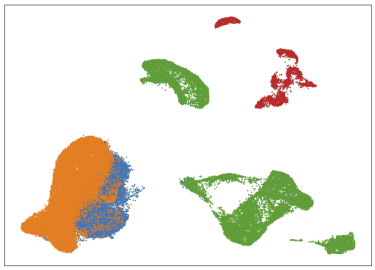


Figure 5.2. UMAP visualization of four different datasets including CIFAR-10 (orange), SVHN (blue), FMNIST (green), USPS (red).

Table 5.1. An example showing how distribution similarities among clients can be perfectly estimated by the principal angles between the client data subspaces. This table shows the proximity matrix of four datasets, where the UMAP visualization has been shown in Fig. 5.3 (c). The entries of this matrix are presented as $x(y)$, where x is the smallest principal angle between two datasets obtained from Eq. 5.1, and y is the summation over the principal angles between two datasets obtained from Eq. 5.2. We let of p in $\mathbf{U}_p$ be 2.

Dataset	CIFAR-10	SVHN	FMNIST	USPS
CIFAR-10	0 (0)	6.13 (12.3)	45.79 (91.6)	66.26 (132.5)
SVHN	6.13 (12.3)	0 (0)	43.42 (86.8)	64.86 (129.7)
FMNIST	45.79 (91.6)	43.42 (86.8)	0 (0)	43.36 (86.7)
USPS	66.26 (132.5)	64.86 (129.7)	43.36 (86.7)	0 (0)

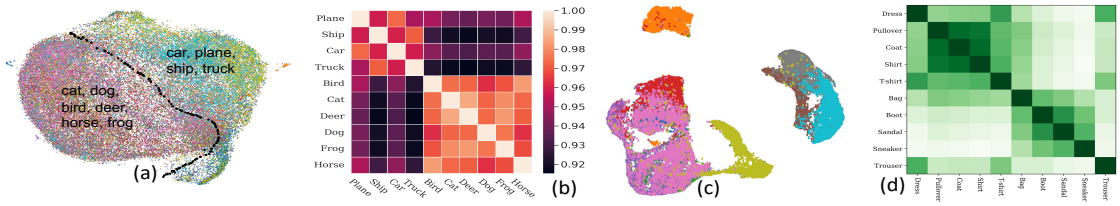


Figure 5.3. The main goal of this figure is to understand the cluster structure of different datasets based on which the Non-IID data partitioning of different datasets can be suggested. (a) depicts the UMAP visualization of CIFAR-10 classes. As can be seen, CIFAR-10 naturally has two super clusters, namely animals (cat, dog, bird, deer, horse, frog) and vehicles (car, plane, ship, truck), which are shown in the purple and green regions, respectively. This means that within each super cluster, the distance between the distribution of the classes is small. While the distance between the distributions of the two super clusters is quite huge. Since the union of clients data is CIFAR-10, two clusters is enough to handle the Non-IIDness across clients. (b) We obtained the proximity matrix A as in Eq. 3.1 and sketched it. The entries of A are the smallest principle angle between all pairs of classes of CIFAR-10. This concurs with (a) showing the cluster structure of CIFAR-10. (c) The data of FMNIST is naturally clustered into three clusters. The structure of the three clusters is also perfectly suggested by our new proposed notion of heterogeneity for this dataset. (d) We did the same thing as in (b) for FMNIST as well and sketched the matrix.

confirmed with UMAP [86] visualization in Fig. 5.2. As can be seen from Fig. 5.2, CIFAR-10 is more similar to SVHN than USPS which reflected in a smaller principal angle between the subspaces of CIFAR-10 and SVHN than that of CIFAR-10 and USPS. It is noteworthy that the smallest principal angle between each pair of classes in each dataset is different as well. For instance, on FMNIST, by setting p of $\mathbf{U}_p$ to 3, the smallest principal angle between Trouser and Dress classes is 22.47° while that of Trouser and Bag classes is 51.7° which stems from more similarity between the distribution of (Trouser, Dress) compared to that of (Trouser, Bag).

Making use of the proposed approach for measuring the similarity/dissimilarity of clients data and invoking the proximity matrix of all clients data under the current Non-IID partitioning method, we will uncover that the existing data partition strategies represent more like an IID partitioning or at most a light Non-IID that may not necessarily be considered as a challenge. Therefore, this motivates us to propose a new partitioning method using our metric, which leads to the following section.

5.2.2 New Non-IID Partitioning Method

Now we are in position to define a new Non-IID partitioning. We say that *a federated network is heterogeneous if each client owns data only from one of super clusters of a dataset.* In particular, we partition all the training samples in each super cluster into shards of n examples and randomly assign two shards to each client only from one of the super clusters. Such Non-IID (2) partitioning is reasonable to expect in heterogeneous federated networks, due to the drastic disparities in the distribution of each client data. In contrast, the data partitioning proposed in previous papers [56, 51], may assign data from all the clusters.

Consider an example. In Table 5.2, we show the average final top-1 test accuracy of all clients on CIFAR-10 for FedAvg under IID (2), the conventional Non-IID (C-NIID) label skew (2), and our proposed Non-IID partitioning which we name as super cluster based Non-IID (SC-NIID). As can be seen from Table 5.2, the C-NIID data partitioning yield accuracy results close to IID data partitioning while our proposed SC-NIID partitioning accuracy results are

far less than that of IID. This shows that the C-NIID data partitioning is not a severe Non-IID and it rather tends to be similar to IID. We will compare the performance of various global and personalized baselines under these Non-IID partitioning methods later on Experiment Section along with some intuitions and remarks.

Table 5.2. Test accuracy comparison on CIFAR-10 across different data partitioning methods. For each partitioning method, the average of final local test accuracy over all clients is reported. We run the FedAVg baseline for each partition 3 times for 100 communication rounds with 10 local epochs and a local batch size of 10.

Dataset	IID (2)	C-NIID (2)	SC-NIID (2)
CIFAR-10	88.15 ± 0.47	83.63 ± 1.27	75.53 ± 3.83

We provide another empirical example to show that our proposed method can effectively produce a more challenging Non-IID data partitioning by leveraging the principal angle between of the data subspaces spanned by the first p significant left singular vectors of the data. In particular, as shown in Table 5.3, we compare the average distance between the data of each partitioning method including IID, C-NIID, and our proposed SC-NIID. We employ the well-known distance measures including Earth Mover's Distance (EMD) [132], Centered Kernel Alignment (CKA) [133], and our proposed method as in Eq. 5.1, and Eq. 5.2 in inspecting the distance between the data of each Non-IID partitioning method. In all of these measures, the smaller the entry is the more IID the data of the partition is. As can be seen, the entries of C-NIID is smaller than that of SC-NIID and are closer to that of IID. Table 5.2, and 5.3 together reveal that firstly the C-NIID is more like an IID partitioning or at least it is not a challenging and severe Non-IID and secondly, they demonstrate that our proposed method as in Eq. 5.1 and Eq. 5.2 can accurately capture the similarity/dissimilarity between two distributions and its results are consistent with that of the well-known distance measures.

The UMAP [86] visualization in Fig. 5.3(a) confirms that the images of CIFAR-10 can naturally be clustered into two super clusters, i.e., cluster of animals (cat, dog, deer, frog, horse, bird) and cluster of vehicles (airplane, automobile, ship, and truck). This shows that the two

Table 5.3. The average distance, which is evaluated by the well-known distance measures, between all 100 participant clients data under different partitioning methods.

Measure	IID	C-NIID (2)	SC-NIID (2)
EMD	0.042	0.17	0.29
Eq. 5.1	0.072	0.202	0.274
Eq. 5.2	0.16	0.28	0.338
CKA	0.94	0.889	0.825

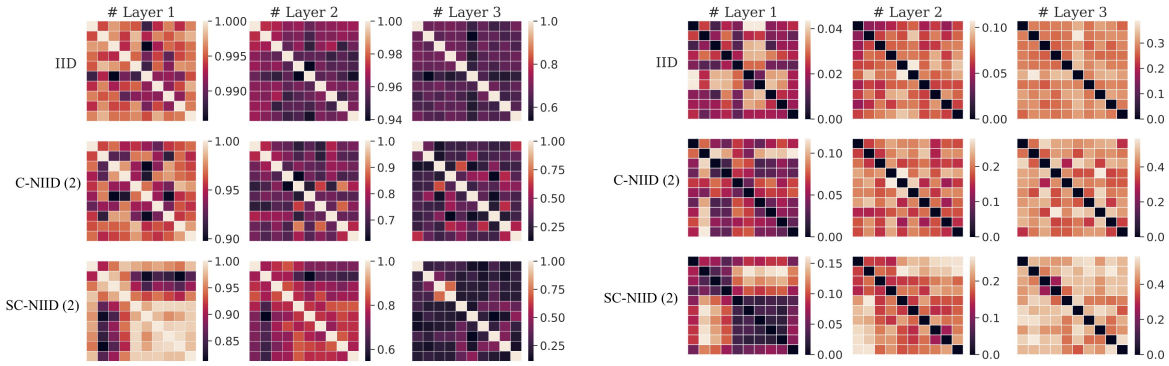


Figure 5.4. Similarities of the outputted feature representation of three different layers of different partitions obtained by CKA (left) and by our proposed measure as in Eq. 5.1 (right) when trained on CIFAR-10. This plot is sketched once the federation is finished.

clusters are the best case for training the local models on partitions of CIFAR-10 dataset in a Non-IID fashion. Fig 5.3(b) also depicts the proximity matrix of CIFAR-10 dataset, whose entries are the principal angle between the subspace of every pair of 10 classes (labels). This further confirms that our proposed notion perfectly captures the heterogeneity, thereby finding the best number of super clusters in each dataset. In particular, our experiments demonstrate that the clients that have the sub-classes of these two big classes have common features and can improve the performance of other clients that own sub-classes of the same big class if they be assigned to the same cluster. A similar discussion can be made about other datasets. In contrast, in almost all of the prior works [56, 36, 70, 32, 31], the data samples with the same label are divided into subsets and each client is only assigned two subsets with different labels which produces a very light Non-IID partition as discussed above. Similar settings have been used where each party only has data samples with a single label [31].

Another method that has been adopted in the literature to simulate Non-IID label skew is

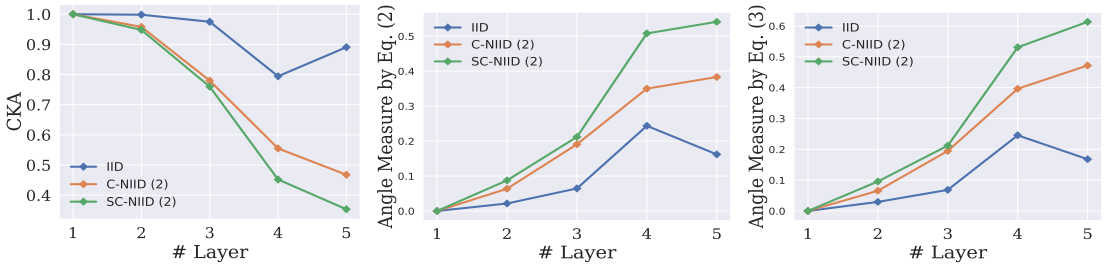


Figure 5.5. The means of the similarities of different layers in different local models obtained by CKA (left) and by our proposed measure as in Eq. 5.1 (middle) and Eq. 5.2 (right).

allocating a proportion of the data of each label/class according to Dirichlet distribution (Dir(α)). In particular, random samples for class c from Dirichlet distribution according to $p_c \sim \text{Dir}(\alpha)$ are selected from the *whole dataset* and to each client j random samples of class c according to $p_{c,j}$ proportion is assigned. While Dir(α) label skew can simulate label imbalance in the network, it may not simulate severe data heterogeneity across clients because the assigned samples are randomly selected from the whole dataset and they are not necessarily from only one of the super clusters of that dataset. We rather suggest random samples for each class be selected from each super-cluster on a dataset according to Dirichlet distribution. This will provide a more severe and challenging Non-IIDness. We postpone the experiments on this new Dir(α) Non-IID setup to sections 5.3.1, 5.3.2, and 5.3.3.

It is noteworthy that the number of formed super clusters in each dataset can be controlled by the distance threshold (clustering threshold) which is a hyperparameter in hierarchical clustering. The smaller the clustering threshold the larger number of super clusters will be formed and in turn, the more heterogeneous the partitioning will be. Therefore, the level of data heterogeneity across clients can be easily controlled by the clustering threshold.

5.2.3 A Closer Look at FL Under Heterogeneity

To understand which of the Non-IID setups is a better benchmark to be considered in heterogeneous FL scenarios, we perform an experimental study on heterogeneous local models. We choose CIFAR-10 with 10 clients and LeNet-5 as a convolutional neural network with 5

layers. We then train the model two times independently with the same random seeds with FedAvg, wherein the first training we partition the data according to the C-NIID (2) and in the second time we partition the data according to our new notion of Non-IID-ness i.e., SC-NIID (2). We train both cases for 100 rounds and each client optimizes for 10 local epochs at each round. For each layer in the models, we use CKA [133] and our proposed measure in Eq. 5.1 to evaluate the similarity of the output features between two local models, given the same input testing samples for each of the cases independently. CKA outputs a similarity score between 0 (not similar at all) and 1 (identical).

We first show the pairwise CKA features similarity of the first three layers of LeNet-5 across local models in Fig. 5.4. Interestingly, as can be seen, we find that features outputted from IID data and our proposed Non-IID setup show lower CKA similarity compared to that of the C-NIID. It indicates that our proposed benchmark provides a more severe heterogeneity across different clients. By averaging the pairwise CKA features similarity in Fig. 5.4, we can obtain a single value to approximately represent the similarity of the feature outputs by each layer across different clients for IID, C-NIID, and our SC-NIID setups. We demonstrated the approximated layer-wise features similarity in Fig. 5.5. These results witness that the models trained on our proposed Non-IID setup consistently produced features across clients for all layers which are less similar to IID in comparison to that of C-NIID.

5.3 Experiments

We perform an extensive empirical analysis using a standard image classification task for multiple popular federated learning datasets and various statistical heterogeneity setups.

5.3.1 Experimental Setup

Datasets and Models. We use an image classification task and 3 popular datasets, i.e., CIFAR-10 [28], CIFAR-100 [28], STL-10 [134] to employ our novel partitioning method. For all experiments, we consider LeNet-5 [27] architecture for CIFAR-10 dataset and ResNet-9 [30]

architecture for CIFAR-100, and STL-10 datasets. Details of the architectures can be found in subsection 3.4.5.

Baselines and Implementation. To assess the performance of our novel Non-IID partitioning method against conventional partitioning, we compare the results over a set of baselines. For SOTA personalized FL methods, the baselines include LG-FedAvg [69], Per-FedAvg [51], Clustered-FL (CFL) [55], and IFCA [54]. Besides, we compare with FedAvg$^+$ [31], FedProx$^+$ [32] FedNova$^+$ [34], and SCAFFOLD$^+$ [33]. It is noteworthy that the superscript $+$ sign on global baselines means that these baselines has been fine-tuned by the clients and thus are considered personalized ones. We report the average results performance over three independent trials.

C-NIID ($\rho\%$) Label Skew and SC-NIID ($\rho\%$). In this setting, the conventional method first randomly assigns $\rho\%$ of the total available labels of a dataset to each client and then randomly distributes the samples of each label amongst clients own those labels as in [37]. In our SC-NIID method, we first form the super clusters according to our proposed method explained in Section 5.2. We then randomly assign all clients to only one of the formed super clusters. The number of assigned clients to each super cluster is proportional to the size (number of samples that cluster contains) of the super cluster which means that if the size of a super cluster is bigger, more clients are assigned to that particular super cluster. Next, the total samples of each super cluster is divided into shards and each client picks $\rho\%$ of the total labels contained in the super cluster to which the client belongs.

Conventional Dir (C-Dir) and SC-Dir. In this setting, the conventional method distributes the training data between the clients based on the Dirichlet distribution. In particular, for N clients data it samples N random numbers $\mathbf{p}_i \sim Dir_N(\alpha)$ from $Dir(\alpha)$ distribution [1] and allocates the

[1]The value of α controls the degree of Non-IID-ness. A big value of α e.g., $\alpha = 100$ mimics identical label distribution (IID), while $\alpha = 0.1$ results in a split, where the vast majority of data on every client are Non-IID.

$\mathbf{p}_{i,j}$ proportion of the training data of class i to client j as in [37]. In our proposed SC-Dir(.) method, we again constitute the super clusters with the help of Eq. 5.1 or Eq. 5.2 as explained in Section 5.2. We then assign a certain number of clients randomly to only one of these super clusters depending upon the size of each super cluster. We then let the data within each super cluster be assigned according to the Dir(.) distribution as in C-Dir(.) to the clients that belong to each cluster.

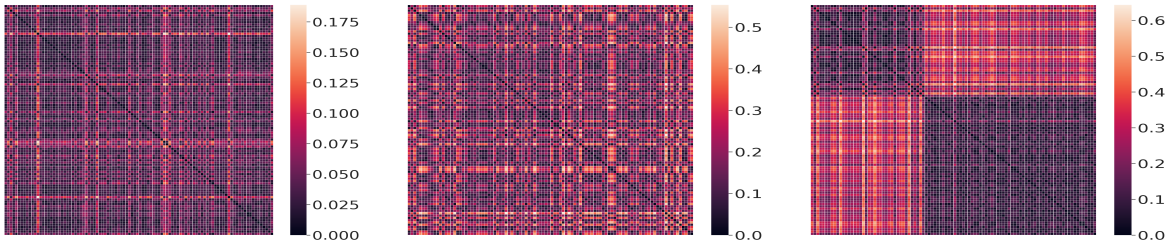


Figure 5.6. We obtained a proximity matrix of 100×100 dimension which corresponds to 100 clients' data distribution by the EMD measure for three different data partitioning method, i.e., IID (left), C-NIID (middle), and SC-NIID (right). This concurs with Fig. 5.3 showing that CIFAR-10 naturally have two Non-IID clusters as well with Fig. 5.7, and Fig. 5.8.

5.3.2 Comparing the Performance of SOTA Baselines on the Conventional and Newly Proposed Non-IID Method

We conduct experiments to compare the above four Non-IID partitioning methods i.e, C-NIID (2), C-Dir(.), SC-NIID (2), and SC-Dir(.) methods on CIFAR-10, CIFAR-100, and STL-10 datasets and present the results in Table 5.4. It can be observed that all SOTA baselines present a significant performance drop when the clients data are distributed according to the newly proposed Non-IID setup compared to the conventional Non-IID setups used in prior arts. Based on the results of table 5.4 and through extensive studies, we have the following key findings: 1) The newly proposed Non-IID partitioning is more challenging compared to the C-NIID. Since effectively addressing data heterogeneity is of paramount concern in federated learning, we suggest that challenging tasks like SC-NIID ($\rho\%$), and SC-Dir(α) should be included in the benchmark for future FL setups. 2) Under the new Non-IID setup none of the existing SOTA

Table 5.4. Evaluating different personalized FL methods under different data partitions. We evaluate on ResNet-9 with CIFAR-100 and STL-10 as well as LeNet-5 on CIFAR-10. For each communication round, a fraction 10%, 30%, 10% of the total 100 clients are randomly selected. We set the local epoch and batch size to 10.

Dataset	Algorithm	C-NIID(2)	SC-NIID(2)	C-Dir(0.5)	SC-Dir(0.5)
CIFAR-10	SOLO	83.62 ± 0.72	75.68 ± 0.47	48.45 ± 0.57	43.44 ± 0.43
	FedAvg+	83.46 ± 1.15	77.02 ± 0.73	53.31 ± 0.85	43.43 ± 1.81
	FedProx+	85.01 ± 0.59	76.54 ± 1.15	52.69 ± 0.60	44.61 ± 1.43
	FedNova+	83.99 ± 0.68	77.36 ± 0.46	53.21 ± 0.82	46.09 ± 0.33
	Scafold+	82.69 ± 2.93	78.88 ± 0.44	41.55 ± 5.82	16.45 ± 2.71
	LG	83.18 ± 0.46	76.40 ± 0.46	31.35 ± 7.42	41.36 ± 0.75
	PerFedAvg	83.60 ± 0.48	75.70 ± 0.74	54.90 ± 0.25	42.63 ± 1.08
	IFCA	86.59 ± 1.16	80.49 ± 0.86	57.78 ± 1.14	51.54 ± 0.98
CIFAR-100	SOLO	76.71 ± 1.12	73.16 ± 0.17	45.24 ± 1.74	40.43 ± 0.85
	FedAvg+	87.99 ± 0.97	81.55 ± 0.59	66.15 ± 2.79	53.41 ± 1.16
	FedProx+	87.68 ± 0.82	80.70 ± 0.81	67.67 ± 0.98	53.86 ± 0.25
	FedNova+	87.22 ± 0.45	80.75 ± 1.28	63.15 ± 2.32	53.77 ± 0.52
	Scafold+	55.97 ± 13.29	15.61 ± 9.85	39.04 ± 23.73	11.43 ± 3.90
	LG	78.27 ± 1.31	72.87 ± 0.80	44.43 ± 1.40	39.44 ± 0.90
	PerFedAvg	77.47 ± 1.30	58.93 ± 0.90	58.02 ± 2.38	37.96 ± 1.24
	IFCA	88.06 ± 0.19	82.23 ± 0.73	69.89 ± 1.64	55.66 ± 0.86
STL-10	SOLO	78.14 ± 2.27	69.88 ± 0.62	51.12 ± 1.16	42.17 ± 0.68
	FedAvg+	85.67 ± 2.23	77.23 ± 1.45	56.80 ± 5.64	49.69 ± 0.66
	FedProx+	86.83 ± 1.83	83.03 ± 1.40	64.97 ± 5.86	56.35 ± 1.85
	FedNova+	88.45 ± 0.27	83.18 ± 2.28	60.00 ± 6.77	52.84 ± 1.03
	Scafold+	31.74 ± 1.80	27.71 ± 4.12	50.31 ± 2.90	31.28 ± 7.52
	LG	84.42 ± 0.77	75.52 ± 0.91	52.56 ± 2.94	46.82 ± 1.47
	PerFedAvg	52.91 ± 2.12	51.64 ± 0.57	30.10 ± 2.74	31.80 ± 3.14
	IFCA	88.99 ± 0.45	81.13 ± 0.46	67.99 ± 1.66	60.73 ± 1.27

FL algorithms outperforms others in all cases. This further indicates the importance of having a more comprehensive Non-IID distribution benchmark.

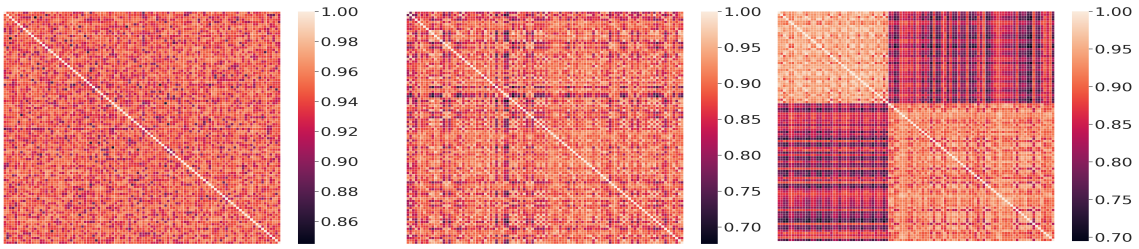


Figure 5.7. We obtained a proximity matrix of 100×100 dimension which corresponds to 100 clients' data distribution by the CKA measure for three different data partitioning methods, i.e., IID (left), C-NIID (middle), and SC-NIID (right). This concurs with Fig. 5.3 showing that CIFAR-10 naturally have two Non-IID clusters as well with Fig. 5.6, and Fig. 5.8

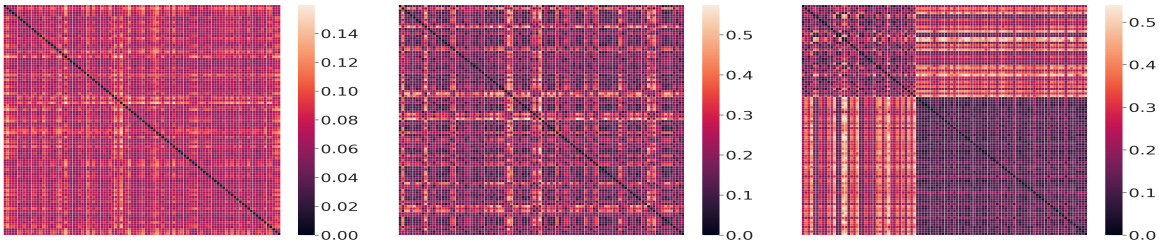


Figure 5.8. We obtained a proximity matrix of 100×100 dimension which corresponds to 100 clients' data distribution by our own proposed measure as in Eq. 3.1 for three different data partitioning methods, i.e., IID (left), C-NIID (middle), and SC-NIID (right). This concurs with Fig. 5.3 showing that CIFAR-10 naturally have two Non-IID clusters as well with Fig. 5.6, and Fig. 5.7

5.3.3 Comparing the Level of Data Heterogeneity of the C-NIID and SC-NIID

Data heterogeneity will impact the convergence of a federated model and hurt its performance. This necessitates a deeper investigation of data heterogeneity which has been missing. Herein, we provide some visualizations to facilitate understanding the heterogeneity level of the setup that has been used as a standard one in the prior works. To this end, we distribute CIFAR-10 according to three data partitioning methods, i.e., IID, C-NIID, and SC-NIID among the clients and then we leverage three distribution similarity/dissimilarity measures namely EMD [132], CKA [133], and our proposed method as in Eq. 5.1, to monitor the significance

of data heterogeneity across 100 participant clients in the federation. Figures 5.6–5.8 visualize the proximity matrix whose entries are the distance between clients data computed by EMD, CKA, and our proposed method, respectively. These figures further confirm that the C-NIID partitioning is more like IID. This is while as can be seen in these three figures, our newly proposed SC-NIID distributes the data across all clients which is very different from IID. This corresponds to the fact that our method takes the intrinsic structure of the dataset into account and from a certain number of Non-IID super clusters according to the entries of the proximity matrix and then distributes the data among the clients from only one of these Non-IID super clusters. According to what we discussed, it is therefore natural to ask the following questions: *How do prior FL approaches tackle the newly proposed Non-IID partitioning?*

Remark-1. It is not clear how the prior personalized FL algorithms which have been proposed to alleviate the statistical data heterogeneity should be extended to tackle the new Non-IID setup. For instance, as mentioned in [130], the proposed regularization method is effective only for light data heterogeneity but would not be beneficial or even lead to dropping in the performance with the increase of the heterogeneity. This could be true about many of the prior arts as also confirmed by the results reported in table 5.4. This further emphasizes that the performance of the prior arts should be evaluated under severe and challenging Non-IID setups to judge their efficacy.

Remark-2. While many works in the literature have highlighted the detrimental impacts of statistical data heterogeneity, showing that heterogeneity can lead to poor performance in federated optimization which necessitates novel forms of personalization, some works [131] discussed distinct benefits of data heterogeneity in their FL analyses. In contrast to this work [131], we believe to judge whether the impact of the corresponding heterogeneity issue is positive or negative, we should first provide a true notion of data heterogeneity as well as standard Non-IID benchmarks without which it will be hard to attest measurable benefits of heterogeneity.

Remark-3. In Table 5.4 we provided the results of SOLO[2] training as well. Considering SC-Dir(0.5) on CIFAR-10, as is evident from Table 5.4, except for a few baselines, the accuracy results of other baselines are worse than SOLO training which means that it is not justified for clients to participate in a federation under the newly proposed Non-IID setting. Because the clients not only do not gain from the federation but the federation also has degraded their performance. This phenomenon can be explained by the fact that depending upon the dataset the proposed Non-IID partitioning distributes highly heterogeneous data to clients under which federation is not beneficial. This is while considering C-Dir(0.5) most of the baselines benefited from the federation and yielded better results compared to SOLO training. The same behavior can be seen on other datasets and on SC-NIID(2).

5.3.4 Under What Heterogeneity Environment Clients Benefit from Collaboration?

In this section, by an empirical pseudo example, we demonstrate that the widely used Non-IID setup, i.e., C-NIID in the literature is not a real Non-IID partition. In doing so, we sample two clients named C1, and C2 and manually assign two labels to each as in Table 5.5 and let them do federation with vanilla FedAvg [31] for 20 communication rounds. When C1 owns labels "ship+truck", and C2 owns labels "plane+car" as in row 4, according to the common belief in the FL community we should have expected that due to the existence of C-NIIDness the average final accuracy is worse than SOLO training as in row 1. But surprisingly it is not the case and even though these clients have no label overlap, their performance improved through federation compared to SOLO. In contrast, as can be seen from row 5, even though C1 and C2 have 50% labels overlap (50% distribution similarity according to C-NIID), by taking part in the federation, the accuracy of C1 drops compared to SOLO. This is while row 3 with the same condition (50% labels overlap) improved the results of C1 through federation. This further confirms that C-NIID cannot represent a true view of non-IIDness in FL. These anomalies can be

[2]In SOLO baseline the client trains a model lonely on its own local data without taking part in the federation.

justified by our newly proposed Non-IID partition which is not based on the simple label skew but based on the angle in between of the clients' data subspaces. In particular, two clients data are considered heterogeneous if their data are drawn from two different super clusters formed by our approach in Eq. 5.1 or Eq. 5.2. The result of C1 in row 4 improved because both of the clients own data from the same super cluster. The results of other rows can be justified in the same fashion.

The authors hope the reader take the preceding discussion as an example showing that the adopted Non-IID partition in the prior works may not represent a true data heterogeneity setup and also the authors do not claim that the proposed method can justify all anomalies. We rather like to encourage the researchers to design more comprehensive alternatives to the current Non-IID setups.

Table 5.5. A pseudo example illustrating that the adopted heterogeneous setup in the prior works which relies upon the label skew can not represent a true view of Non-IIDness. That's why it has not necessarily had a detrimental effect on the clients accuracy performance.

Row	Case	C1 Accuracy	Status
1	ship+truck ; [8,9]	83.20	Solo Training
2	C1: ship+truck, C2:ship+truck; [8,9]	86.02	Full overlap (IID)
3	C1: ship+truck, C2:truck+plane; [9,0]	84.02	One overlap on vehicle
4	C1: ship+truck, C2:plane+car; [0,1,8,9]	83.36	No overlap (only vehicle)
5	C1: ship+truck, C2:truck+bird; [9,2]	82.43	One overlap on animal
6	C1:ship+truck, C2:car+bird; [1,2,8,9]	81.78	No overlap (animal+vehicle)
7	C1: ship+truck, C2:ship+bird; [8,2]	81.72	One overlap on animal
8	C1: ship+truck, C2:cat+dog; [3,5,8,9]	81.63	No overlap (only animal)

5.3.5 A New Benchmark for Non-IID FL

As mentioned earlier, existing studies have been evaluated on simple partitioning strategies, i.e., Non-IID label skews (20%) and Non-IID label skew (30%). In data partitioning with $a\%$ label skew, the union of client data will only be one dataset. Focusing on CIFAR-10 and with 20% label skew, most of the 100 clients can have either 50% label overlap or 100%. This simulates a partially Non-IID setting and cannot represent a full view of Non-IIDness. Because

the data distributions of clients are the sub-distributions of a unique dataset such as CIFAR-10. This is the reason that statistical data heterogeneity has never been a big issue in the proposed personalized FL algorithms.

In order to better assess the potential of the SOTA baselines under a real-world and challenging Non-IID task where the local data of clients have strong statistical heterogeneity, and the data distributions of clients are not the sub-distributions of a unique dataset, we design the following benchmark naming it as MIX-4. We hope Mix-4 becomes a standard benchmark for comparing different SOTA against each other in the FL community. We assume that each client owns data samples from one of the four datasets, i.e., USPS [85], CIFAR-10, SVHN, and FMNIST. In particular, we distribute CIFAR-10, SVHN, FMNIST, and USPS among x, y, z, and v clients, respectively (x+y+z+v= total clients) where each client receives a certain number of samples from all classes but only from one of these datasets. This is a very challenging Non-IID task. Under this circumstance, the union of the clients data is not a single dataset and in each round of communications, there will be some clients whose data distribution varies drastically. In Table 5.6, we present the test accuracy of the SOTA baselines on Mix-4. It can be seen from Table 5.6 that the accuracy performance of all the methods drops significantly compared to the ones reported for C-NIID (the widely used Non-IID setup in prior works) in Table 5.4. These sorts of realistic assumptions have never been adopted in the literature. That's why it is a common belief that all users can benefit from heterogeneity. We hope designing Mix-4 opens up a new avenue to design more standard real-world benchmarks for comparing different personalized SOTA in the FL community.

5.4 Conclusion

In this chapter, we proposed a new notion and framework for Non-IID partitioning in FL setups. In the proposed method, each dataset is divided to several super clusters by analyzing the principal angles between subspaces of different classes of the dataset. Now in order to distribute

Table 5.6. The benefits of personalized SOTA algorithms should be testified when the tasks are severely Non-IID. This table evaluates different FL approaches in the challenging scenario of MIX-4 in terms of test accuracy performance. All approaches have substantial difficulties in handling this scenario with tremendous data heterogeneity. We run each baseline 3 times for 50 communication rounds with 5 local epochs.

Algorithm	MIX-4
SOLO	55.08 ± 0.29
FedAvg	63.68 ± 1.64
FedProx	61.86 ± 3.73
FedNova	60.92 ± 3.60
Scaffold	69.26 ± 0.84
LG	58.49 ± 0.46
PerFedAvg	42.60 ± 0.60
IFCA	70.32 ± 3.57
CFL	61.18 ± 2.63

Table 5.7. The formed super clusters for each dataset. The numbers correspond to the labels according to the standard naming of labels in each dataset.

Dataset	Formed Super Clusters
CIFAR-10	{0,1,8,9}, {2,3,4,5,6,7}
CIFAR-100	{0, 83, 53, 82}, {1, 54, 43, 51, 70, 92, 62}, {23, 69, 30, 95, 67, 73}, {47, 96, 59, 52}, {2, 97, 27, 65, 64, 36, 28, 61, 99, 18, 77, 79, 80, 34, 88, 42, 38, 44, 63, 50, 78, 66, 84 , 8, 39, 55, 72, 93, 91, 3, 4, 29, 31, 7 , 24, 20, 26, 45, 74, 5, 25, 15, 19, 32, 9, 16, 10, 22, 40, 11, 35, 98, 46 , 6, 14, 57, 94, 56, 13, 58, 37, 81, 90, 89, 85, 21, 48, 86, 87, 41, 75, 12, 71, 49, 17, 60, 76, 33, 68}
STL-10	{2, 8, 9}, {0, 1, 7, 3, 4, 5, 6}

heterogeneous data to all clients, all training data in each super cluster is partitioned to different shards. Each client then is assigned a certain number of shards from only one of the super clusters. The proposed method addresses a broad range of data heterogeneity issues beyond simpler forms of Non-IIDness like label skews. Extensive evaluations show that our approach generates a more sever data heterogeneity partitioning compared to the C-NIID. Other than that, via experiments we demonstrated that under the newly proposed Non-IID setup, none of the prior arts consistently beat others. Further, we showed that in contrast to the common belief in the FL community where statistical data heterogeneity has been modeled by Non-IID label

skew, clients can benefit from the Non-IID label skew which witnesses that the adopted C-NIID setup is not a true estimate of data heterogeneity. Furthermore, we presented a new challenging Non-IID setup which can be considered as an standard benchmark for the future FL papers.

Chapter 6

Data Summarization

6.1 Introduction

In many problems in sociology, finance, computer science, and operations research, we have networks of interconnected entities and pairwise relations between them. A problem that arises often in practice is calculating the expected value of a variable in the form of the sum of the values of a smooth function on the nodes of a graph. For example, in semi-supervised learning with Graph Neural Networks (GCNs), the generalization error is specified as the average of the loss functions associated with all the nodes in a graph. Before the elections, opinion polls are usually designed to represent the opinions of a networked population about the candidates in expectation [135]. In social networks, having a small average distance to other individuals in the network is considered a key factor to be influential [136]. In much environmental monitoring, knowing the average temperature, humidity and water quality of various regions allow for taking preventive actions against forest fires and water contamination [137, 138]. Finally, in health care, monitoring various health measures such as a population's average blood pressure, weight, and cholesterol level, allows for designing health strategies and disease prevention actions [139].

In real-world networks containing millions of nodes and billions of edges, it is impractical to evaluate the function at every single node. Therefore, an important question is how to select a small representative subset (coreset) of nodes from a million-node graph such that the weighted sum of function values sampled at the nodes of the subset can be a good estimate of the sum of

function values over the entire graph [140]. Another constraint that often arises in practice is that evaluating the function may incur a cost that is not necessarily equal for all the nodes in the graph. For example, placing sensors in certain areas may be more expensive than others [141]. Similarly, measuring the blood pressure of people in hard-to-reach areas is more expensive. Hence, we wish to find a small representative weighted subset of nodes subject to a limited budget.

There are main challenges in finding such a small coreset. First, the selected subset and the corresponding weights should provide a bound on the estimation error of the first moment (mean) of the function at all the nodes in the graph. Moreover, the method should be simple to implement, computationally inexpensive, and have theoretical guarantees relating coreset size to both computational complexity and the quality of the approximation. Finally, different nodes may have non-uniform cost, and hence we need to be able to bound the error of estimating the mean while finding a low-cost solution.

Very recently, the authors in [139] considered this problem and provided an upper-bound on the estimation error of the mean of the function evaluated over the entire graph with a weighted subset of functions at nodes of an arbitrary subset. The provided quadrature-type bound can be used to bound any mean estimation problem in which the function is sufficiently low-frequency, i.e., the function can be expressed in terms of a small number of eigenfunctions (with large eigenvalue) of a lazy random walk transition matrix on the underlying graph. This includes problems such as measuring average blood pressure in a database [139] and subsampled kernel two sample testing [142]. Intuitively, the placements that minimize the quadrature-type bound have the property that the random walks starting from every node in the subset and weighted by its corresponding weight, overlap very little. However, the problem of finding the near-optimal subset and the associated weights by minimizing the upper-bound has remained unaddressed.

In this work, we address the question of finding a coreset of nodes that minimizes the estimation error of the expected function value over the entire graph, by minimizing the upper-bound provided in [139]. Inspired by the recent work of [143] on Bayesian coreset constructions,

we propose a greedy algorithm to find a small subset of nodes and their weights that closely approximate the first moment of the function over the entire graph. Moreover, we consider an extended problem of having each sampled node come with a non-uniform cost, a problem that arises in applications such as sensor placement, marketing, and other knapsack-type problems. We extend our algorithm to this setting and characterize through a simple parameter the error in estimating the mean of the function one is willing to tolerate in order to seek a low-cost solution.

6.2 Related Work

The ever-increasing size of modern datasets motivated data reduction techniques as a preprocessing step to speed up subsequent optimization problems. Existing graph summarization methods mainly focus on obtaining sparse subgraphs that can be used to approximate properties of the original graph (degree distribution, size distribution of connected components, diameter, or community structure). Core techniques include graph clustering or community detection methods [144, 145], bit compression-based methods [146], sparsification-based [147] and sketching methods [148, 149], and influence-based methods [150]. While these methods maintain the structural properties of the original graph, they cannot guarantee that an algorithm working on the summary provides a solution close to the solution found based on the entire data. For instance, graph summarization algorithms cannot guarantee that the function sampled at the selected points has similar statistics to the function evaluated on the entire network.

In contrast, coresets are weighted subsets of the data, which guarantee that for the specific problem at hand, models fitting the coreset also provide a good fit for the original dataset. This approach has been successfully applied to a variety of problems including K-means and K-median clustering [151], mixture models [152], low rank approximation [153], spectral approximation [154, 155], Nystrom methods [154, 156], and Bayesian inference [143]. Coreset construction methods traditionally perform importance sampling with respect to sensitivity score, defined as the importance of the point with respect to the objective function we wish to minimize,

to provide high-probability solutions [151, 152, 153]. Greedy algorithms, which are special cases of the Frank-Wolfe algorithm, were described more recently to provide worst-case guarantees for problems such as support vector machines [157], and Bayesian inference [143]. In this work, we propose a greedy algorithm to construct coresets for estimating the first moment of a function defined on the nodes of large graphs.

6.3 General Framework

Here, we aim to choose a subset of vertices and weights to be the best representative of the graph. More specifically, we assume that the dataset V has some geometric structure encoded in a graph $G = (V, E)$, where V and E denote the set of nodes and edges. The problem is how to choose a subset $S \subseteq V$ of vertices and weights $w_s > 0$ for every $s \in S$ in order to approximate the mean μ_f with a weighted approximation $\widehat{\mu}_f$, where

$$\mu_f = \frac{1}{|V|} \sum_{v \in V} f(v), \qquad \widehat{\mu}_f = \sum_{s \in S} w_s f(s). \tag{6.1}$$

The graph can either be given a priori, or constructed on a point cloud $V \subset \mathbb{R}^d$ via a kernel $K : V \times V \to \mathbb{R}_+$. We must assume that the function $f : V \to \mathbb{R}$ must have some relationship to the graph, or else the optimal sampling would be a random search. In our context, this assumption takes the form that the function can be expressed in terms of a small number of eigenfunctions (with large eigenvalue) of a lazy walk graph transition matrix on graph G.

Definition 1. *The lazy random walk graph* [1] *transition matrix P on $G = (V, E)$ is constructed by*

$$P = \frac{1}{d_{max}}(A - D) + I,$$

where A denote the (weighted) symmetric adjacency matrix of G such that $A_{ij} = A_{ji} \geq 0$ is

[1] A lazy random walk is a walk on a graph with self-loops. Here the probability of taking self loop is inversely proportional to the degree of the node.

the weight of the edge connecting nodes i and j, D is a diagonal matrix with $D_{ii} = \sum_j A_{ij}$, $d_{max} = \max_i D_{ii}$, *and I is the identity matrix.*

Definition 2. *A function* $f : V \to \mathbb{R}$ *is in* P_λ *for a lazy walk transition matrix P, with eigendecomposition* $P = U\Lambda U^*$, *if*

$$f = \sum_{|\lambda_i| > \lambda} b_i U_i,$$

where λ_i *and* U_i *are the eigenvalues and eigenvectors of P,* $b_i > 0$ *is a weight, and* $0 \leq \lambda \leq 1$ *is a parameter controlling the degree of smoothness.* P_λ *is the subspace spanned by the eigenfunctions of P associated with eigenvalue* $\geq \lambda$.

Spectral smoothness on graphs, as in Definition 2, is necessitated by the fact that there is no traditional notion of gradients on graphs. There's a large body of work that shows that eigenfunctions of the kernel with large eigenvalue are of lower frequency than eigenvectors with small eigenvalue [158, 159]. Thus spectrally band-limited functions must themselves be smooth [139]. Our ability to approximate the mean of the function decays as the frequency of the function increases since the function becomes more chaotic. This is a fundamental limitation, as functions unrelated to the underlying graph structure cannot be approximated with a coreset in any way better than random sampling.

We can also consider the additional constraint that choosing every nodes $v \in V$ may come with a cost $C : V \to \mathbb{R}_+$. We seek an algorithm that will choose the subset of nodes $S \subseteq V$ in a way that is

- Greedy in order to quickly choose and add additional points,
- Minimizes the error in estimation of the mean of f, and
- Incorporates the cost $C(v)$ by choosing low-cost points to the subset.

The motivation for this framework and use of the lazy walk graph transition matrix comes out of the work in [139], in which it was noted that the mean error in f can be bounded in terms

of the choice of points and weights, as in the following

$$\forall f \in P_\lambda, \left| \frac{1}{n} \sum_{v \in V} f(v) - \sum_{s \in S} w_s f(s) \right| \leq \|f\|_{P_\lambda} \min_{\ell \in N} \frac{1}{\lambda^\ell} \left(\left\| P^\ell \sum_{s \in S} w_s \delta_s \right\|_2^2 - \frac{1}{n} \right)^{\frac{1}{2}}, \tag{6.2}$$

for $\ell \in \mathbb{N}$, $0 < \lambda < 1$, and w_s summing to 1. Moreover, n is the number of vertices of the graph, and δ_s is the binary vector taking value zero at every index except for s. This result implies that minimizing the right hand side in terms of w_s and S will yield a stronger bound on the moment estimation of f. $P^\ell \delta_s$ is the probability distribution of a random walker starting in s after ℓ jumps. Therefore, intuitively the subset S that minimizes the quadrature-type bound has the property that the random walks starting from every node s in the subset and weighted by its corresponding weight w_s overlap very little.

6.3.1 Problem Definition

Similar to the concept of duality in convex optimization, in order to achieve the best results for the upper bound in (6.2) we shall minimize it. Therefore, the optimization problem can be formulated in the following form

$$\begin{aligned} \underset{w}{\text{minimize}} \quad & \|f\|_{P_\lambda} \frac{1}{\lambda^\ell} \left(\left\| P^\ell \sum_{s \in S} w_s \delta_s \right\|_2^2 - \frac{1}{n} \right)^{\frac{1}{2}} \\ \text{subject to} \quad & |S| \leq K,\ S \subset V, \\ & \sum_{s \in S} w_s = 1, \quad w_s > 0. \end{aligned} \tag{6.3}$$

where K is the maximum number of selected vertices. Several discrete and continuous methods are relevant when solving the preceding problem in (6.3). However, the main issue with such an optimization is the constraint of choosing $S \subset V$, which leads to a difficult combinatorial optimization problem. For example, one approach related to (6.3) is that of computing a

cardinality constrained minimization by solving sparse principal component analysis (PCA) problem [160]. However, solving the sparse PCA optimization problem entails semidefinite relaxation and a greedy algorithm to calculate a full set of good solutions, which is very expensive with total complexity of $O(n^3)$ [161].

A better way to view this problem is in terms of an L_2-minimization problem on P. Because $\|f\|_{P_\lambda}$ and λ are properties of the function f we're analyzing (see Def. 2) and independent of the choice of points and weights, we will drop these terms in the discussion of the optimization scheme, when appropriate. Similarly, we will drop the dependence on ℓ for notational simplicity and simply deal with an arbitrary lazy random walk matrix P. This is not an issue as P^ℓ is also a lazy random walk transition matrix, with eigenvalues λ^ℓ.

We can also rewrite our cost as a matrix multiplication $Pw = P\sum_s w_s\delta_s$, and define our target function to be the normalized ones vector $\frac{1}{n}1$. We note that the cost function of (6.3) can be reframed by the above notational changes, as well as bringing the $\frac{1}{n}$ inside the norm, to arrive at

$$\left\| P\sum_{s\in S} w_s\delta_s \right\|_2^2 - \frac{1}{n} = \|Pw\|_2^2 + \frac{1}{n} - 2\left\langle Pw, \frac{1}{n}1 \right\rangle = \left\| Pw - \frac{1}{n}1 \right\|_2^2,$$

where the first equality comes from the fact that $\langle Pw, 1\rangle = 1$, and the second equality comes from the fact that $\|\frac{1}{n}1\|_2^2 = \frac{1}{n}$ and completing the square. To this end, (6.3) can be posed in an equivalent form as

$$\begin{aligned} \underset{w}{\text{minimize}} \quad & \left\| Pw - \frac{1}{n}1 \right\|_2 \\ \text{subject to} \quad & \sum_i 1[w_i > 0] \le K \\ & w_i \ge 0 \end{aligned} \tag{6.4}$$

Problem (6.4) shifts the original mean bound in (6.3) into a constrained least squares problem for finding the optimal weights w. Due to the shifting of $\frac{1}{n}1$ inside the norm, we are no longer constrained to have $\sum_{i\in V} w_i = 1$ (discussed in detail in Appendix D.3). With that being said, our

algorithm still has a weight normalization β^* that arises in (6.6) and will push $\|w\|_1$ close to 1.

The optimization problem (6.4) we construct has been considered previously [162, 163], however, these have limitations in the context of our graph problem. Briefly, [162] considers cardinality regularized loss function minimization subject to simplex constraints, which yields a computational complexity $O(n^4)$ for our graph problem. Similarly, [163] has theoretical guarantees only under the restricted isometry property [164]. Moreover, it assumes the size of the subset K is known, which may vary in a posteriori fashion in our applications. Finally, we extend the literature by providing guarantees on the convergence rate explicitly in terms of the number of selected elements and the rate at which the error decreases as we sample more coreset points.

We also note that the Problem (6.4) can be solved by relaxing the nonconvex cardinality constraint $\sum_i 1[w_i > 0] \leq K$ to a simplex constraint $\sum_i \|P_i\| w_i = \sum_i \|P_i\|$ for the columns P_i of the matrix P, and using the Frank–Wolfe (FW) algorithm that iteratively chooses the point most aligned with the residual error. However, there are some problems for which FW performs very poorly for any number of iterations because FW must scale the objective function in Problem (6.4) suboptimally by $\sum_i w_i \|P_i\|$ rather than $\|\frac{1}{n}1\|$, in order to maintain feasibility [143]. In this Chapter, we mainly focus on the above-mentioned constrained optimization problem. In the following section, we provide a new radial optimization algorithm for problem (6.4) and demonstrate that it yields theoretical guarantees at a significantly reduced computational cost. More importantly, in contrast to FW and its extensions [165], the algorithm developed in this work has no correction steps and geometric error convergence.

6.4 Optimization Algorithm

The optimization problem in (6.4), despite involving a least squares cost function, is nonconvex in w due to the cardinality constraint. Inspired from [143], without any loss of generality, the weights, w could be scaled by an arbitrary constant $\beta \geq 0$ without affecting

feasibility. This motivates rewriting (6.4) as

$$\begin{aligned} \underset{w,\beta}{\text{minimize}} \quad & \left\| \beta Pw - \frac{1}{n}1 \right\|_2 \\ \text{subject to} \quad & \sum_i 1[w_i > 0] \leq K \\ & w_i \geq 0, \beta \geq 0 \end{aligned} \tag{6.5}$$

Following [143] we now begin by solving the optimization problem in β. We define β^* as the solution to the problem (6.5) given w which can be computed analytically as

$$\beta^* = \frac{\|\frac{1}{n}1\|}{\|Pw\|} \max\left\{0, \left(\frac{Pw}{\|Pw\|}\right)^T \left(\frac{\frac{1}{n}1}{\|\frac{1}{n}1\|}\right)\right\} = \frac{1}{n\|Pw\|} \max\left\{0, \left(\frac{Pw}{\|Pw\|}\right)^T 1\right\}. \tag{6.6}$$

Substituting β^* in the objective above and expanding the square, the weighted subset of nodes (coreset) can be found by solving:

$$\begin{aligned} \underset{w}{\text{minimize}} \quad & \frac{1}{n}\left(1 - \max\left\{0, \left(\frac{Pw}{\|Pw\|}\right)^T 1\right\}\right) \\ \text{subject to} \quad & \sum_i 1[w_i > 0] \leq K \\ & w_i \geq 0 \end{aligned} \tag{6.7}$$

This result shows that the minimum of the problem in (6.7) occurs by alignment of the vectors Pw and $\frac{1}{n}1$ independent of their norm. Finally, we define $P(w) = \sum_i w_i \frac{P_i}{\|P_i\|}$, and $P^* = \frac{\frac{1}{n}1}{\|\frac{1}{n}1\|} = \frac{1}{\sqrt{n}}1$. With that in mind, we can reformulate (6.7) in an equivalent maximizing problem as in the following

$$\begin{aligned} \underset{w}{\text{maximize}} \quad & P(w)^T P^* \\ \text{subject to} \quad & \sum_i 1[w_i > 0] \leq K \\ & \|P(w)\| = 1 \\ & w_i \geq 0 \end{aligned} \tag{6.8}$$

According to the constraints in (6.8), we are optimizing the objective over a unit hypersphere rather than the simplex. Before solving the problem in (6.8), we extend it to a more general case

where nodes have different selection costs, and then we provide a new algorithm for solving it.

6.4.1 Optimization with Selection Cost

In many applications, selecting some reference points representing the whole data involve some factors such as the cost of selection associated with each data. For example, placing sensors in certain regions may be more expensive than others. Similarly, measuring the blood pressure of people in hard-to-reach areas is more expensive. In problems such as these, it is natural to seek a trade-off between the two goals of minimizing the error (selecting nodes $S \subseteq V$ that maximize alignment of $P(w)$ and P^*) and the cost of choosing nodes in S. To this end, we incorporate a new parameter C into the problem controlling the cost associated with each node. In what follows, we will focus on the reparameterized maximization problem, which can be written as:

$$\begin{aligned} \underset{w}{\text{maximize}} \quad & P(w)^T P^* - \lambda C(S) \\ \text{subject to} \quad & |S| \leq K \text{ for } S = \{i : w_i > 0\} \\ & \|P(w)\| = 1 \\ & w_i \geq 0 \end{aligned} \tag{6.9}$$

Now we provide a greedy algorithm, sample cost greedy iterative geodesic ascent (SCGIGA), for the above optimization problem. At every iteration, the algorithm finds the point indexed by v^* for which the geodesic between $P(w)$ and $P(v^*)$ is most aligned with the geodesic between $P(w)$ and P^*. We then find the set of all vertices for which the alignment is within κ-percent of the alignment of v^*, for a parameter $0 < \kappa \leq 1$. Among such points, we add the vertex with minimum cost to the solution. Once the point has been added, the algorithm reweights the coreset and iterates. SCGIGA detailed in Algorithm 12 outlines how to solve the optimization problem in (6.9). It is noteworthy that SCGIGA is a general algorithm that is also valid for the case where there are equal costs associated with every data point. This corresponds to $\kappa = 1$ in our settings.

A benefit of the greedy approach is that increasing the number of coreset points to $K+1$ simply requires adding the next point for the geodesic ascent and marginally changing the weights. This is unlike [163], in which changing to the $K+1$ simplex requires recomputing the optimization scheme and in no way guarantees to keep the previous K selected data points. Similarly, this formulation allows us to easily incorporate the non-uniform cost of sampling points.

We note that the most expensive computation in Algorithm 12 is $\langle P_v, P(w)\rangle$ across v, but that $P(w)$ is the sum of at most K vectors and each loop only adds one additional vector. Thus we can store previous inner products and only take different weighted combinations on each iteration, so each loop only requires n inner products. Hence, Algorithm 12 has average complexity $O(Knm)$, where m is the average sparsity of a column.

We will establish the convergence guarantees and rate in Section 6.5, and connect this convergence to bounding the estimate of the mean of f as in (6.2). But first, we wish to establish that incorporating the cost of sampling into the optimization does not significantly impact the resulting minimum value, and thus results in a close to optimal greedy bound on the error in estimating the mean of f. The gap can be characterized by the one parameter κ to be chosen by the user, and choosing the minimal cost point among the set of vertices similar to v^* only scales the resulting bound by a factor of κ.

Theorem 5. *Let C^k_{max} be the sum of the k largest elements of C. If we choose $\lambda \leq \frac{1-\kappa}{C^k_{max}\min\|P_i\|\sqrt{n}}$, then the solution to* (6.9)*, $P(w^*)$, satisfies*

$$P(w^*)^T P^* \geq \kappa \max_w P(w)^T P^*.$$

The proof can be found in the Appendix.

Theorem (5) implies we will never incur too much loss to the original objective by incorporating the cost of selecting the nodes that minimize the error. Similarly, this implies that we can make every greedy choice and step in whatever fashion is deemed best for cost, as long

Algorithm 12. Algorithm of SCGIGA

Initialization $w_0 \leftarrow 0$
$\forall v \in V$, $P_v \leftarrow \frac{P_v}{\|P_v\|}$
for $k \in \{0, ..., K\}$ **do**
  $a_k \leftarrow \frac{1-\langle 1, P(w_k)\rangle P(w_k)}{\|1-\langle 1, P(w_k)\rangle P(w_k)\|}$
  $\forall v \in V, a_{kv} \leftarrow \frac{P_v-\langle P_v, P(w_k)\rangle P(w_k)}{\|P_v-\langle P_v, P(w_k)\rangle P(w_k)\|}$
  $v^* \leftarrow \arg\max_{v\in V} {a_k}^T a_{kv}$; find vertex that maximizes alignment
  $W \leftarrow \left\{v \in V | {a_k}^T a_{kv} \geq \kappa {a_k}^T a_{kv^*}\right\}$; find vertices within κ-percent of max
  $v_k \leftarrow \arg\min_{v\in W} C_v$; find vertex in W with min cost
  $\zeta_0 = \langle \frac{1}{\sqrt{n}} 1, P_{v_k} \rangle$
  $\zeta_1 = \langle \frac{1}{\sqrt{n}} 1, P(w_k) \rangle$
  $\zeta_2 = \langle P_{v_k}, P(w_k) \rangle$
  $\delta_k \leftarrow \frac{\zeta_0-\zeta_1\zeta_2}{(\zeta_0-\zeta_1\zeta_2)+(\zeta_1-\zeta_0\zeta_2)}$; choose the step size
  $w_{k+1} \leftarrow \frac{(1-\delta_k)w_k+\delta_k 1_{v_k}}{\left\|(1-\delta_k)P(w_k)+\delta_k P_{v_k}\right\|}$; update the weight
end
$w = \beta w_k$ {Scale weights by β}
$S = \{v \in V | w_v > 0\}$
Sample f at nodes in S
$\widehat{\mu}_f = \sum_{v\in S} w_v f(v)$
return $\widehat{\mu}_f$, w

as the choice is within κ of the optimal step direction.

6.5 Theoretical Aspects

Here we examine the guarantees that Algorithm 12 yields for bounding the error in estimating the mean of $f \in P_\lambda$, where P_λ is the subspace spanned by the eigenfunctions of P associated with eigenvalues $\geq \lambda$. We will derive the general theorem for arbitrary κ, and as a special case, we recover the results when all the nodes have equal selection cost ($\kappa = 1$).

Theorem 6. *Let $f \in P_\lambda$ with mean μ_f as in* (6.1)*, and assume there is a cost for selecting every node $v \in V$, i.e., $C(v) : V \to \mathbb{R}_+$ and a slack parameter κ. If we choose the set of points S and*

weights w_s using Algorithm 12 such that $|S| = K$, then

$$\left| \mu_f - \sum_{s \in S} w_s f(s) \right| \leq \frac{\|f\|_{P_\lambda}}{\lambda^\ell} \frac{\eta \, v_K}{\sqrt{n}},$$

where $v_K = O((1-\kappa^2\varepsilon^2)^{K/2})$ for some ε and $\eta = \sqrt{1 - \kappa^2 \max_{i \in V} \left\langle \frac{P_i}{\|P_i\|}, \frac{1}{\sqrt{n}} 1 \right\rangle^2}$.

The proof can be found in the Appendix.

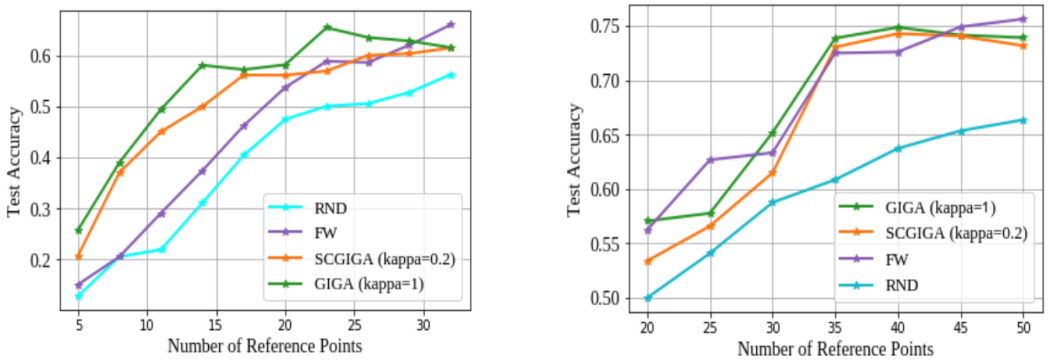
The main ideas of this theorem are three-fold. The first series of lemmas that must be proved are extensions of the core lemmas in [143], which establish that the error is a contractive map after each update, which is later used in a fixed point theorem to show convergence and rate. The extensions we make here are 1) generalize their results to graphical geometries, rather than the log-likelihood construction proposed in [143], and 2) allow for a κ relaxation of the greedy choice and prove how this relaxation affects the contractive mapping.

The second main idea behind this theorem is to show that the κ-relaxation does not significantly affect the fixed point argument to show convergence of the cost-aware greedy algorithm as the number of chosen points grows. This again borrows from [143] in this more general setting, but the argument effectively comes down to propagating the additional error accrued by the cost-aware algorithm.

The final main idea behind this theorem is connecting the coreset choice to the first moment quadrature bound in [139]. This comes from a shifting of the bound in (6.2) to allow for weights to not necessarily have to sum to 1, and the recognition that this bound for general points and weights chosen is equivalent to the bound being minimized in (6.4).

We wish to note that the bound established in Theorem 6 may not be sharp for the first few points greedily sampled, but does become sharp asymptotically due to fixed point convergence. The bound on the first moment estimate of f in terms of (6.2) is close to sharp if there is a spectral gap at λ [139].

As a particular case of this theorem, when we always choose the optimal greedy node



Stochastic block model Cora citation dataset

Figure 6.1. Classification accuracy obtained from a GCN model after a number of semi-supervised training iterations for different algorithms.

independent of cost, we recover the following guarantee.

Corollary 1. *Let $f \in P_\lambda$, and choose the set of points S and weights w_s using Algorithm 12 such that $|S| = K$. Then*

$$\left| \mu_f - \sum_{s \in S} w_s f(s) \right| \leq \frac{\|f\|_{P_\lambda}}{\lambda^\ell} \frac{\eta \, v_K}{\sqrt{n}},$$

where $v_K = O((1-\varepsilon^2)^{K/2})$ for some ε and $\eta = \sqrt{1 - \max_{i \in V} \left\langle \frac{P_i}{\|P_i\|}, \frac{1}{\sqrt{n}} 1 \right\rangle^2}$.

The proof of this corollary is a special application of Theorem 6, which is proved in the Appendix.

6.6 Empirical Evidence

In this section, we evaluate our model on several sets of experiments. In order to compare both cases of our algorithm, i.e., when there are non-uniform costs on selecting different data points ($\kappa \neq 1$) and when there are equal costs associated with every data point ($\kappa = 1$) we consider a fixed cost on each data randomly generated from a uniform distribution over $[0,1)$. Similarly, in all examples involving point clouds, the generated graph comes from a K-nearest neighbor construction with 10 nearest neighbors.

6.6.1 Graph CNN Classification

One example of the importance of bounding means comes in semi-supervised learning. In such a problem, the goal is to approximate a function $h : V \to \mathbb{R}^m$ using a parametric model h_θ (such as the graph convolutional network). The goal is to minimize

$$L_{\theta,V} = \frac{1}{|V|} \sum_{v \in V} \|h_\theta(v) - h(v)\|^2,$$

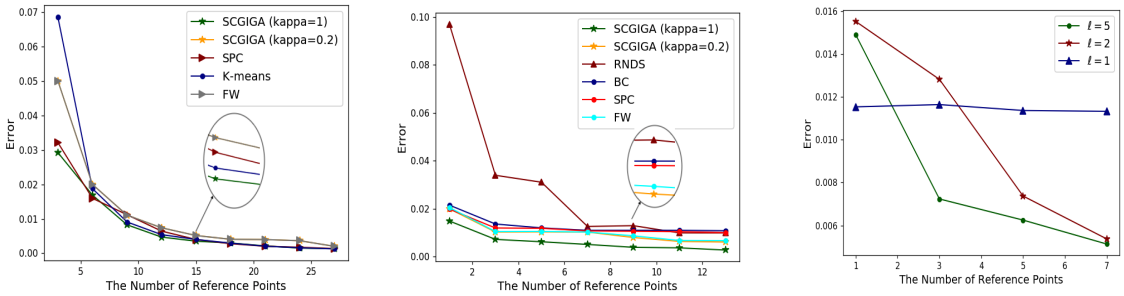
where L is a loss function, chosen judiciously depending upon the application. However, in the event that sampling h is expensive, the goal is to choose a coreset $S \subset V$ in order to estimate $L_{\theta,V}$ with a quadrature formula and the empirical risk

$$L_{\theta,S} = \sum_{v \in S} w_v \|h_\theta(v) - h(v)\|^2.$$

The quadrature error in this context, $|L_{\theta,S} - L_{\theta,V}|$, is exactly the *generalization error*, and is now re-expressed as the estimation of the mean of the function $f(v) = \|h_\theta(v) - h(v)\|^2$. Theorem 6 applies in the situation that $h \in P_\lambda$ and that h_θ is sufficiently regularized to satisfy $h_\theta \in P_\lambda$. We note that the $\|\cdot\|^2$ remains mostly low-frequency as the frequency of the pointwise product of eigenfunctions can be bounded [166]. This implies that by sampling h only at S, and training h_θ on S, one can bound the generalization error on $V \setminus S$ using Theorem 6.

Given the above discussion, we seek to test our algorithm on semi-supervised document classification in citation network datasets, namely, Cora and semi-supervised node classification in a stochastic block model graph with 200 vertices and 10 clusters. We train a two-layer graph convolutional neural network (GCN) as described in [167] and we follow the same pre-processing techniques as well. We compare against the same baseline methods as in the experiments presented in this Chapter tested on Cora dataset which is a real citation network dataset with $2,708$ nodes and $5,429$ edges as well as a stochastic cluster-based graph dataset.

A GCN takes a feature matrix and an adjacency matrix as inputs and every vertex of



Gaussian model | Stochastic Block Model | Stochastic Block Model for ℓ

Figure 6.2. The error comparison of estimating the mean of the function defined on the clusters (the two left ones), and the impact of the shaping parameter ℓ on the performance (Right one).

the graph produces a vector, whose elements correspond to the score of belonging to different classes. An identity matrix is added to the original adjacency matrix in order to enforce self loops.

The neural network is trained in a semi-supervised setting, where the network is fed with the feature and adjacency matrices of the entire graph while the loss is only computed on the labeled vertices. Here the labeled vertices are the subset of vertices that are picked by the proposed method on the normalized adjacency matrix. We train both datasets for a maximum of 100 epochs using Adam [168] with a learning rate of 0.01 and early stopping with a window size of 10, i.e. we stop training if the validation loss does not decrease for 10 consecutive epochs (as was done in [167]). The results are summarized in Figure 6.1. Despite small fluctuations due to random initialization, as expected, the test accuracy tends to increase as more labeled points are utilized for training. Further, as can be seen from the figure our proposed algorithm, SCGIGA, has superior performance in selecting the subset of data that comprises the most representative points of clusters. Lastly, because of the existence of outliers in a random graph, the accuracy of the proposed algorithm starts to improve slowly at about 60% accuracy. However, we note that the model is trained with only 10% of data which was considered to be the labeled ones, so this also implicitly suggests that our algorithm successfully picks out the most informative nodes.

6.6.2 Mean Function on Clustered Data

A standard unsupervised learning task is to learn clusters from data, either on a graph or a point cloud, and use those clusters to select points and weights for averaging a function. Standard clustering algorithms include K-means clustering and spectral clustering [2].

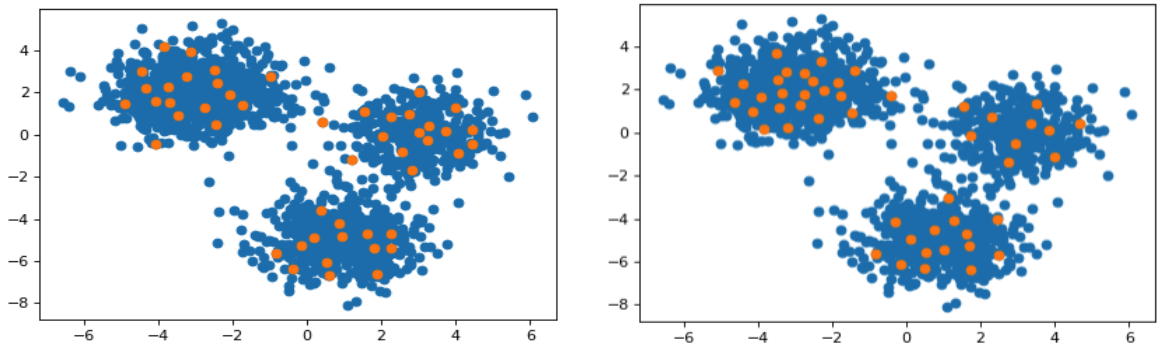
The first experiment we run is on the Gaussian model with three components. The components have the mean vectors of $[1\ -3], [-3\ \ 2], [3\ \ 0]$, and all have the covariance matrix of the identity matrix, I. The components contain 20%, 30%, and 50% of the data. The function we choose to model is a simple smooth function that is an indicator function of the small cluster (1 on the small cluster, 0 on other clusters). The results in Fig. 6.2 for the Gaussian Model show the error comparison of three algorithms including FW, K-means, and spectral clustering (SPC) versus the number of clusters (centroids) or reference points. The performance of the algorithms is quantified by an error metric which is defined as the Err $= \left|\sum_{|S|=k} a_s f(S) - \mu_f\right|^2$. For fairness to comparable algorithms, a_s in K-means, FW, and SPC is defined as the ratio of the data in each cluster to the whole data, while $a_s = w_s$ in our algorithm. Here, f is the indicator function on the small cluster and μ_f is the mean of the function.

As is evident from the figure, our algorithm outperforms K-means, FW, and spectral clustering for the whole range of the number of reference points. As can be seen in this figure the maximum number of reference data is 14 out of 10000 and our results reveal that the cost of the optimal solution (C_{COS}) is 5.920, while the cost we got with our cost aware algorithm, i.e., the cost of sub-optimal solution (C_{CSO}) is 0.106. The more surprising observation in these figures is that even in the case where a fixed cost associated with each data in which results in our algorithm yielding a sub-optimal solution (the solution which is in the κ percent of the optimal) our algorithm continues to do very well and work better than standard algorithms. Also note that for only 3 reference points, which would lead to an ideal coreset of one point per cluster, SCGIGA considerably reduces the error compared to the competing algorithms.

[2] In multivariate statistics spectral clustering techniques get rid of some of the eigenvalues of the similarity matrix of the data to perform nonlinear dimensionality reduction before clustering in fewer dimensions.

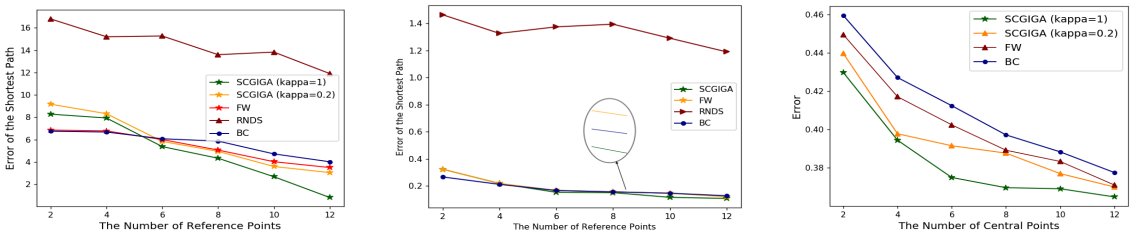
We consider another experiment on clustered graphical data. In Fig. 6.2 a stochastic block model with three clusters is designed where the first cluster contained 10% of the data and the other two clusters contained 50% and 40% of the data. Then we define an indicator function on the small cluster, and we look for the average function value on these three clusters. As can be seen from these figures, our algorithm estimates the mean very well while random sampling (RNDS) needs more reference points to catch up with the mean. More importantly, in this experiment by selecting 28 reference data, $C_{\text{COS}} = 15.159$ while $C_{\text{CSO}} = 0.075$.

Further, Fig. 6.2 sketches the impact of the shaping parameter, ℓ from (6.3), in our formulation of the error behavior of the same stochastic block model. Clearly, incorporating a multi-step kernel yields much better performance in estimating the cluster coresets and weights.



Cost associated No cost associated

Figure 6.3. 2% of coresets selected two cases, variable sampling cost with $\kappa = 0.2$ (left) and uniform cost (right). The upper left-hand component, the middle component, and the lower component contain respectively 50%, 20%, 30% of the data, and the average cost per data in each component is 3.25, 0.51, and 2.04, respectively.



A Powerlaw tree graph A random generated graph Facebook ego network

Figure 6.4. Shortest path comparison of different algorithms on different graphs.

Finally Fig. 6.3 shows three Gaussian clusters with the same mean vectors and covariance matrices as in the first experiment considered. These two figures demonstrate the way that the proposed algorithm selects points when there is varying cost associated with each point and the special case of equal costs for choosing each point. In the figure, when there is a higher average cost of sampling on the largest cluster compared to the smallest, the algorithm adapts to resample in a way not simply proportional to the number of points in each cluster. In contrast, when no cost is associated the number of selected points are obviously less than that of the largest component.

6.6.3 Shortest Path on Graph

In this set of experiments, we assume that a graph $\mathscr{G}$ or an adjacency matrix is given. The goal is to find the average distance from a vertex to the rest of the graph, where the distance between two points is computed using Dijkstra's algorithm [3]. In this experiment, rather than computing the shortest path between a given node and every other node on the graph, we calculate the shortest path between the given node and just the reference points S. This is of particular cost benefit on trees that have a large diameter (maximum distance between two nodes) but low average path distance (e.g., trees, powerlaw graphs, etc).

Fig. 6.4 shows the results of the comparison of our algorithm, FW, Betweenness Centrality (BC), and random sampling (RS) on a Powerlaw Tree graph [169]. The error is defined as the difference between the weighted average distance of each vertex of the graph from the reference vertices and the average distance of each vertex from all the vertices of the graph. Fig. 6.4 also demonstrates the same results for a randomly generated graph. In these two figures, both cases i.e., when fixed but different costs are associated with each data (vertex) and when equal costs on the data are included.

Ego Networks In this experiment, we consider a special type of network called an Ego Network. In an Ego Network, there is a "central" vertex (ego vertex) which the network

[3] Dijkstra's algorithm is a greedy nature algorithm that looks for the minimum weighted vertex on every iteration.

highly depends on or revolves around. We consider the real-world Facebook Ego Networks dataset with 4,964 nodes [170]. The dataset contains the aggregated network of some users' Facebook friends. In this dataset, vertices represent individuals on Facebook, and edges between two users mean they are Facebook friends. The Ego Network connects a Facebook user to all of his Facebook friends and is then aggregated by identifying individuals who appear in multiple Ego Networks. Algorithms such as betweenness centrality (BC) were proposed to measure the importance (centrality) of a user in the network. These algorithms select a user as a central one by looking at how many shortest paths pass through that user (vertex). The more shortest paths that pass through the user, the more central the user is in the Facebook network. We run our algorithm on the Facebook dataset to choose the central nodes. We then compare our algorithm with the BC and FW in terms of the error performance metrics described in the first experiment. The results in Fig. 6.4 show that the developed algorithm in this work performs better in selecting the central points in the Facebook graph.

6.7 Discussion and Conclusion

In this Chapter, we introduced a scalable, and theoretically-sound algorithm for minimizing the estimation error of the expected value of a function over the entire graph. Our proposed method can be applied to semi-supervised classification in graph neural networks (GCNs), social network graphs, point clouds, and on many other applications where a similarity metric between data points can be defined. We provided theoretical guarantees on the convergence of the estimated mean and validated its efficiency empirically on real and synthetic datasets. Our work also opens up the notion of optimizing when there is a non-uniform cost of sampling and provides a simple parameter for the trade-off between accuracy and cost.

Appendix A

Preliminaries

A.1 Principal angles between two subspaces.

Let $\mathscr{U} = \text{span}\{\mathbf{u}_1, ..., \mathbf{u}_\text{p}\}$ and $\mathscr{W} = \text{span}\{\mathbf{w}_1, ..., \mathbf{w}_\text{q}\}$ be p and q-dimensional subspaces of $\mathbf{R^n}$ where $\{\mathbf{u}_1, ..., \mathbf{u}_p\}$ and $\{\mathbf{w}_1, ..., \mathbf{w}_q\}$ are orthonormal, with $1 \leq p \leq q$. There exists a sequence of p angles $0 \leq \Theta_1 \leq \Theta_2 \leq ... \leq \Theta_p \leq \pi/2$ called the principal angles, which are defined as

$$\Theta(\mathscr{U}, \mathscr{W}) = \min_{\mathbf{u} \in \mathscr{U}, \mathbf{w} \in \mathscr{W}} \arccos\left(\frac{|\mathbf{u}^T \mathbf{w}|}{\|\mathbf{u}\| \, \|\mathbf{w}\|}\right) \tag{A.1}$$

where $\|.\|$ is the induced norm. The smallest principal angle is $\Theta_1(\mathbf{u}_1, \mathbf{w}_1)$ with vectors $\mathbf{u}_1$ and $\mathbf{w}_1$ being the corresponding principal vectors. The principal angle distance is a metric for measuring the distance between subspaces [171].

A.2 Truncated Singular Value Decomposition (SVD)

Truncated SVD of a real $m \times n$ matrix M is a factorization of the form $\tilde{\mathbf{M}} = \mathbf{U}_p \Sigma_p \mathbf{V}_p^T$ where $\mathbf{U}_p = [\mathbf{u_1}, \mathbf{u_2}, ..., \mathbf{u_p}]$ is an $m \times p$ orthonormal matrix, Σ_p is a $p \times p$ rectangular diagonal matrix with non-negative real numbers on the main diagonal, and $\mathbf{V}_p = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_p]$ is a $p \times n$ orthonormal matrix, where $\mathbf{u}_i \in \mathbf{U}_p$ and $\mathbf{v}_i \in \mathbf{V}_p$ are the left and right singular vectors, respectively.

A.3 Hierarchical Clustering

When constituting disjoint clusters where the number of clusters is not known a priori, hierarchical clustering (HC) [68] is of interest. Agglomerative HC is one of the popular clustering techniques in machine learning that takes a proximity (adjacency) matrix as input and groups similar objects into clusters. Initially each data point is considered as a separate cluster. At each iteration, it repeatedly do the following two steps: (1) identify the two clusters that are the closest, and (2) merge the two most similar clusters. This iterative process continues until one cluster or C clusters are formed. In order to decide which clusters should be merged, a linkage criterion such as single, complete, average, etc is defined [68]. For instance, in “single linkage", the pairwise L_2 (Euclidean) distance between two clusters is defined as the smallest distance between two points in each cluster.

Appendix B

Curricula in Federated Learning

B.1 Strongly Convex Problems

Lemma 1. *The stochastic gradient second moment satisfies:*

$$\mathbb{E}[\|g_k^{(t,j)}\|^2] \leq 2(3+2M)L(f(\theta_k^{(t,j)}) - f^*) + (3+2M)B^{(t,j)} + 3\sigma^2$$

Proof. Follows from,

$$\begin{aligned}
&\mathbb{E}[\|g_k^{(t,j)}\|^2] \leq 3\|\nabla f(\theta_k^{(t,j)})\|^2 + 3\|b_k^{(t,j)}(\theta_k^{(t,j)})\|^2 + 3\mathbb{E}[\|n_k^{(t,j)}(\theta_k^{(t,j)}, \xi_k^{(t,j)})\|^2] \\
&\leq (3+2M)\|\nabla f(\theta_k^{(t,j)})\|^2 + (3+2M)B^{(t,j)} + 3\sigma^2 \\
&\leq 2(3+2M)L(f(\theta_k^{(t,j)}) - f^*) + (3+2M)B^{(t,j)} + 3\sigma^2
\end{aligned}$$

where in the last line we used $\nabla f(\theta^*) = 0$ and L-smoothness.

Define,

$$g^{(t,j)} = \frac{1}{|S_t|}\sum_{k\in S_t} g_k^{(t,j)},\ \bar{g}^{(t,j)} = \frac{1}{|S_t|}\sum_{k\in S_t} \nabla f(x_k^{(t,j)})$$

Note that Assumption 1, in particular (2.4) and (2.4) imply that $\mathbb{E}[g_k^{(t,j)}] = \bar{g}^{(t,j)}$.

The next Lemma is similar to [47, Lemma 5] with a simpler proof of simply adding the terms across agents up using the previous result.

Lemma 2.

$$\mathbb{E}[\|g^{(t,k)} - \bar{g}^{(t,j)}\|^2] \leq \frac{(3+2M)}{Q^2} \sum_{k \in S^{(t)}} \left[2L(f(\theta_k^{(t,j)}) - f^*) + B^{(t,j)} \right] + \frac{3\sigma^2}{Q}$$

The next Lemma is similar to [47, Lemma 6] which in turn follows [172, Lemma 2.1]. Consider the sequence,

$$\hat{\theta}^{(t,j+1)} = \hat{\theta}^{(t,j)} - \alpha^{(t,j)} g^{(t,j)}$$

and note that by this construction $\hat{x}^{(t,J)} = \hat{x}^{(t+1,0)}$. The proof is a straightforward application of strong convexity. It holds that,

Lemma 3.

$$\|\hat{\theta}^{(t,j)} - \alpha^{(t,j)} \bar{g}^{(t,j)} - \theta^*\|^2 \leq \|\hat{\theta}^{(t,j)} - \theta^*\|^2 + \frac{2\alpha^{(t,j)}}{Q} \sum_{k \in S^{(t)}} \Big[(\alpha^{(t,j)} L - 1/2)(f(\theta_k^{(t,j)}) - f(\theta^*)) - \frac{\mu}{2} \|\theta_k^{(t,j)} - \theta^*\|^2 \Big] + \frac{2\alpha^{(t,j)} L}{Q} \sum_{k \in S^{(t)}} \|\hat{\theta}^{(t,j)} - \theta_k^{(t,j)}\|^2$$

Finally, we obtain our first derivation of the expected convergence below.

Lemma 4. *Let $\bar{L} := (L + (3+2M)2L/Q)$ and assume that $\alpha^{(t,j)} \leq \frac{1}{4\bar{L}}$. It holds that the expected distance of the average parameter to the solution satisfies the recursion,*

$$\mathbb{E}[\|\hat{\theta}_k^{(t,j+1)} - \theta^*\|^2] \leq \left(1 - \alpha^{(t,j)}\mu\right) \|\hat{\theta}^{(t,j)} - \theta^*\|^2 - \frac{\alpha^{(t,j)}}{2} \left(f(\hat{\theta}^{(t,j)} - f(\theta^*) \right) + \frac{2\alpha^{(t,j)} L}{Q} \sum_{k \in S^{(t)}} \|\hat{\theta}^{(t,j)} - \theta_k^{(t,j)}\|^2 + \frac{(3+2M)(\alpha^{(t,j)})^2 B^{(t,j)}}{Q} + \frac{3\sigma^2 (\alpha^{(t,j)})^2}{Q}$$

Proof. Using the previous set of results,

$$\begin{aligned}
\mathbb{E}[\|\hat{\theta}_k^{(t,j+1)} - \theta^*\|^2] &\leq \|\hat{\theta}^{(t,j)} - \alpha^{(t,j)}\bar{g}^{(t,j)} - \theta^*\|^2 + (\alpha^{(t,j)})^2\mathbb{E}[\|g^{(t,j)} - \bar{g}^{(t,j)}\|^2] \\
&\leq \|\hat{\theta}^{(t,j)} - \theta^*\|^2 + \frac{2\alpha^{(t,j)}}{Q}\sum_{k\in S^{(t)}}\Big[(\alpha^{(t,j)}L - 1/2)(f(\theta_k^{(t,j)}) - f(\theta^*)) \\
&\quad -\frac{\mu}{2}\|\theta_k^{(t,j)} - \theta^*\|^2\Big] + \frac{2\alpha^{(t,j)}L}{Q}\sum_{k\in S^{(t)}}\|\hat{\theta}^{(t,j)} - \theta_k^{(t,j)}\|^2 \\
&\quad + \frac{(3+2M)(\alpha^{(t,j)})^2}{Q^2}\sum_{k\in S^{(t)}}\Big[2L(f(\theta_k^{(t,j)} - f(\theta^*)) + B^{(t,j)}\Big] + \frac{3\sigma^2(\alpha^{(t,j)})^2}{Q} \\
&\leq \|\hat{\theta}^{(t,j)} - \theta^*\|^2 + \frac{2\alpha^{(t,j)}}{Q}\sum_{k\in S^{(t)}}\Big[(\alpha^{(t,j)}\bar{L} - 1/2)(f(\theta_k^{(t,j)}) - f(\theta^*)) \\
&\quad -\frac{\mu}{2}\|\theta_k^{(t,j)} - \theta^*\|^2\Big] + \frac{2\alpha^{(t,j)}L}{Q}\sum_{k\in S^{(t)}}\|\hat{\theta}^{(t,j)} - \theta_k^{(t,j)}\|^2 \\
&\quad + \frac{(3+2M)(\alpha^{(t,j)})^2B^{(t,j)}}{Q} + \frac{3\sigma^2(\alpha^{(t,j)})^2}{Q}
\end{aligned}$$

By assumption $\alpha^{(t,j)}\bar{L} - 1/2 \leq -\frac{1}{4}$ and then applying Jensen's inequality we have

$$\begin{aligned}
\frac{1}{Q}\sum_{k\in S^{(t)}}\Big[-\frac{1}{4}(f(\theta_k^{(t,j)}) - f(\theta^*)) - \frac{\mu}{2}\|\theta_k^{(t,j)} - \theta^*\|^2\Big] \leq \\
-\left(\frac{1}{4}(f(\hat{\theta}^{(t,j)}) - f(\theta^*)) + \frac{\mu}{2}\|\hat{\theta}^{(t,j)} - \theta^*\|^2\right)
\end{aligned} \tag{B.1}$$

Plugging this expression into the last displayed equation, the conclusion follows.

Next, from Lemma 1 we can conclude that

$$\begin{aligned}
&\tfrac{1}{Q}\sum_{k\in S^{(t)}}\mathbb{E}\left[\left\|g_k^{(t,j)} - g^{(t,j)}\right\|^2\right] \leq \\
&\tfrac{2(3+2M)L}{Q}\sum_{k\in S^{(t)}}\Big(f(\theta_k^{(t,j)}) - f(\theta^*)\Big) + (3+2M)B^{(t,j)} + 3\sigma^2
\end{aligned} \tag{B.2}$$

Next, we derive a recursion on the average parameter deviation.

Lemma 5. *Let $\mu > 0$. The average iterate deviation satisfies the bound,*

$$\begin{aligned}\mathbb{E}\left[\tfrac{1}{Q}\sum_{k\in S^{(t)}}\|\hat{\theta}^{(t,j+1)}-\theta_k^{(t,j+1)}\|^2\right] &\leq (1-\alpha^{(t,j)}\mu/2)\mathbb{E}\left[\tfrac{1}{Q}\sum_{k\in S^{(t)}}\|\hat{\theta}^{(t,j)}-\theta_k^{(t,j)}\|^2\right] \\ &+\tfrac{2(3+2M)L(\alpha^{(t,j)})^2}{Q}\sum_{k\in S^{(t)}}\left(f(\theta_k^{(t,j)})-f(\theta^*)\right) \\ &+\alpha^{(t,j)}\left(\alpha^{(t,j)}(3+2M)+B^{(t,j)}/\mu\right)B^{(t,j)}+3(\alpha^{(t,j)})^2\sigma^2\end{aligned}$$

Proof. Indeed, compute directly,

$$\begin{aligned}\mathbb{E}\left[\tfrac{1}{Q}\sum_{k\in S^{(t)}}\|\hat{\theta}^{(t,j+1)}-\theta_k^{(t,j+1)}\|^2\right] &\leq \mathbb{E}\left[\tfrac{1}{Q}\sum_{k\in S^{(t)}}\|\hat{\theta}^{(t,j)}-\theta_k^{(t,j)}\|^2\right] \\ +\tfrac{(\alpha^{(t,j)})^2}{Q}\sum_{k\in S^{(t)}}\mathbb{E}\left[\|g^{(t,j)}-g_k^{(t,j)}\|^2\right] &- \tfrac{2\alpha^{(t,j}}{Q}\sum_{k\in S^{(t)}}\mathbb{E}\left[\left\langle\theta_k^{(t,j)}-\hat{\theta}^{(t,j)},g_k^{(t,j)}-g^{(t,j)}\right\rangle\right]\end{aligned}$$

For the second term in the above expression, we can apply (B.2). For the third, we note that,

$$\begin{aligned}&-\sum_{k\in S^{(t)}}\mathbb{E}\left[\left\langle\theta_k^{(t,j)}-\hat{\theta}^{(t,j)},g_k^{(t,j)}-g^{(t,j)}\right\rangle\right] \\ &=-\sum_{k\in S^{(t)}}\left\langle\theta_k^{(t,j)}-\hat{\theta}^{(t,j)},\nabla f(\theta_k^{(t,j)})+b_k^{(t,j)}(\theta_k^{(t,j)})\right\rangle \\ &+\sum_{k\in S^{(t)}}\left\langle\theta_k^{(t,j)}-\hat{\theta}^{(t,j)},\tfrac{1}{Q}\sum_{k\in S^{(t)}}\left[\nabla f(\theta_k^{(t,j)})+b_k^{(t,j)}(\theta_k^{(t,j)})\right]\right\rangle \\ &=\sum_{k\in S^{(t)}}\left\langle\hat{\theta}^{(t,j)}-\theta_k^{(t,j)},\nabla f(\theta_k^{(t,j)})+b_k^{(t,j)}(\theta_k^{(t,j)})\right\rangle \\ &\leq\sum_{k\in S^{(t)}}\left[f(\hat{\theta}^{(t,j)})-f(\theta_k^{(t,j)})-\tfrac{\mu}{2}\|\theta_k^{(t,j)}-\theta^{(t,j)}\|^2\right]+\sum_{k\in S^{(t)}}B^{(t,j)}\|\theta_k^{(t,j)}-\theta^{(t,j)}\|\end{aligned}$$

where we used strong convexity in the inequality. Applying Young's inequality to obtain $B^{(t,j)}\|\theta_k^{(t,j)}-\theta^{(t,j)}\| \leq \frac{\mu}{4}\|\theta_k^{(t,j)}-\theta^{(t,j)}\|^2+\frac{1}{\mu}(B^{(t,j)})^2$ yields the final result.

Now we want to use the previous Lemma in order to bound the contribution of the average iterate discrepancy to the overall descent appearing in Lemma 4. Taking a sum for a

given t, for $j = 1, ..., J$, we can see that

$$\begin{aligned}\sum_{j=0}^{J} \mathbb{E}\left[\frac{1}{Q}\sum_{k\in S^{(t)}} \|\hat{\theta}^{(t,j)} - \theta_k^{(t,j)}\|^2\right] \leq \sum_{j=0}^{J} \frac{2\alpha^{(t,j)}L}{Q} \prod_{l=j}^{J}(1-\alpha^{(t,l)}\mu/2) \\ \left[2\alpha^{(t,j)}(3+2M)L\sum_{k\in S^{(t)}}\left(f(\theta_k^{(t,j)}) - f(\theta^*)\right)\right. \\ \left. + \left(\alpha^{(t,j)}(3+2M) + B^{(t,j)}/\mu\right)B^{(t,j)} + 3\alpha^{(t,j)}\sigma^2\right]\end{aligned} \tag{B.3}$$

With that, we proceed with the main result:

Proof. of Theorem 1 From Lemma 4 and (B.3)

$$\begin{aligned}\mathbb{E}[\|\hat{\theta}^{(t+1,0)} - \theta^*\|^2] \leq \prod_{j=0}^{J}(1-\alpha^{(t,j)}\mu/2)\|\hat{\theta}^{(t,0)} - \theta^*\|^2 + \sum_{j=0}^{J}\frac{2\alpha^{(t,j)}L}{Q}\prod_{l=j}^{J}(1-\alpha^{(t,l)}\mu/2) \\ \left[\left(2\alpha^{(t,j)}(3+2M)L - \tfrac{1}{2}\right)\sum_{k\in S^{(t)}}\left(f(\theta_k^{(t,j)}) - f(\theta^*)\right)\right. \\ \left. + \left(2\alpha^{(t,j)}(3+2M) + B^{(t,j)}/\mu\right)B^{(t,j)} + 6\alpha^{(t,j)}\sigma^2\right]\end{aligned}$$

Noting that the assumption on the Theorem implies that the term involving the objective value difference is negative, we obtain the statement of the main result.

B.2 Nonconvex Objectives

Proof. of Theorem 4 As standard, we begin by applying the Descent Lemma across subsequent averaging steps.

$$\begin{aligned}f(\theta^{(t+1,0)}) - f(\theta^{(t,0)}) \leq \left\langle \nabla f(\theta^{(t,0)}), \theta^{(t+1,0)} - \theta^{(t,0)}\right\rangle \\ + \tfrac{L}{2}\|\theta^{(t+1,0)} - \theta^{(t,0)}\|^2 \leq -\left\langle \nabla f(\theta^{(t,0)}), \frac{1}{Q}\sum_{k\in S^{(t)}}\sum_{j=0}^{J}\alpha^{(t,j)}g_k^{(t,j)}\right\rangle \\ + \tfrac{L}{2}\left\|\frac{1}{Q}\sum_{k\in S^{(t)}}\sum_{j=0}^{J}\alpha^{(t,j)}g_k^{(t,j)}\right\|^2\end{aligned}$$

Now, we consider the discrepancy of $g_k^{(t,j)}$ to $\nabla f(\theta^{(t,0)})$ to obtain a perturbation from the decrease we expect to get, that we wish to eventually bound relative to said decrease. Specifically, taking

total expectations (and implicitly using the tower property):

$$\mathbb{E}\left[\frac{1}{Q}\sum_{k\in S^{(t)}}\sum_{j=0}^{J} g_k^{(t,j)}\right] = \frac{1}{Q}\sum_{k\in S^{(t)}}\sum_{j=0}^{J}\alpha^{(t,j)}\mathbb{E}\Big[\nabla f(\theta^{(t,0)}) + b_k^{(t,0)}(\theta^{(t,0)})$$
$$-f(\theta^{(t,0)}) - b_k^{(t,0)}(\theta^{(t,0)}) + \nabla f(\theta^{(t,0)},\xi_k^{(t,j)}) - \nabla f(\theta^{(t,0)},\xi_k^{(t,j)}) + \nabla f(\theta^{(t,j)},\xi_k^{(t,j)})\Big]$$
$$= \frac{1}{Q}\sum_{k\in S^{(t)}}\sum_{j=0}^{J}\alpha^{(t,j)}\left[\nabla f(\theta^{(t,0)}) - \mathbb{E}\left[\nabla f(\theta^{(t,0)},\xi_k^{(t,j)}) + \nabla f(\theta^{(t,j)},\xi_k^{(t,j)})\right]\right]$$

and so, combining the previous two sets of equations,

$$f(\theta^{(t+1,0)}) - f(\theta^{(t,0)}) \leq -\frac{\sum_{j=0}^{J}\alpha^{(t,j)}}{Q}\|\nabla f(\theta^{(t,0)})\|^2 + \frac{\sum_{j=0}^{J}\left(\alpha^{(t,j)}\sum_{l=j}^{J}\alpha^{(t,l)}\right)}{Q}LG^2 + \frac{\sum_{j=0}^{J}\left(\alpha^{(t,j)}\right)^2 LG^2}{2Q}$$

from which we obtain the final result.

Appendix C

Clustered Federated Learning

C.1 Proof of Theorems

In this part, we provide proof of the theoretical results discussed in this chapter.

C.1.1 Proof of Theorem 3.

The proof follows along the same lines as [43, Theorem 4.2]. We proceed with the same structure with minor adjustments as needed.

We start with a descent Lemma, whose proof is unchanged.

Lemma 6. *[43, Lemma B.1]*

$$\mathbb{E}\left[\mathbb{E}_{\mathscr{P}_t}[F(\theta_g^{t+1}) - F(\theta_g^t)]\right] \leq -\eta_t \mathbb{E}\left[\mathbb{E}_{\mathscr{P}_t}[\langle \nabla F(\theta_g^t), g_g^t \rangle]\right] + \frac{\eta_t^2 L}{2} \mathbb{E}\left[\mathbb{E}_{\mathscr{P}_t}[\|\theta_g^t\|^2]\right]$$

The next result is a minor alteration of [43, Lemma B.2].

Lemma 7.

$$\mathbb{E}\left[\mathbb{E}_{\mathscr{P}_t}[\|g_g^t\|^2]\right] \leq \left(\tfrac{c}{K} + 1\right) \tfrac{1}{c} \left[\sum_{j=1}^{c} \|\nabla F_j(\theta_j^t)\|^2\right] + \tfrac{\sigma^2}{KB} \leq$$
$$\tfrac{\lambda}{c^2} \left(\tfrac{c}{K}+1\right) \left\| \sum_{j=1}^{\bar{c}} \nabla F_j(\theta_j^t) \right\|^2 + \left(\tfrac{c}{K}+1\right) \tfrac{1}{c} \left[\sum_{j=\bar{c}+1}^{c} \|\nabla F_j(\theta_j^t)\|^2\right] + \tfrac{\sigma^2}{KB}$$

The next statement remains unchanged;

Corollary 2. *[43, Corollary B.4]*

$$-\eta_t \mathbb{E}\left[\mathbb{E}[\langle \nabla F(\theta_g^t), g_g^t \rangle]\right] \leq -\mu\eta_t(F(\theta_g^t) - f^*) - \frac{\eta_t}{2c}\left\|\sum_{j=1}^{c} \nabla F_j(\theta_j^t)\right\|^2 + \frac{\eta_t L^2}{2c}\sum_{j=1}^{c}\|\theta_g^t - \theta_j^t\|^2.$$

Again, we have a slight modification of the result [43, Lemma B.5];

Lemma 8.

$$\mathbb{E}\left[\frac{1}{c}\sum_{j=1}^{c}\|\theta_g^t - \theta_j^t\|^2\right] \leq \left(\frac{K+1}{K}\right) \times \left(\frac{C_1+E}{p}\sum_{k=t_c+1}^{t-1}\eta_k^2\left[\frac{\lambda}{p}\left\|\sum_{j=1}^{\bar{c}}\nabla F_j(\theta_j^{(k)})\right\|^2 + \sum_{j=\bar{c}+1}^{c}\|\nabla F_j(\theta_j^k)\|^2\right]\right)$$
$$+ \left(\frac{K+1}{K}\right) \times \sum_{k=t_c+1}^{t-1}\frac{\eta_k^2\sigma^2}{B},$$

From this, we finally derive the following descending equation,

$$\mathbb{E}[F(\theta_g^{t+1})] - F^* \leq \Delta_t \mathbb{E}[F(\theta_g^t) - F^*] + c_t + \frac{\eta_t}{2\bar{c}^2}\left[-1 + \lambda L\eta_t\left(\frac{C_1+K}{K}\right)\right]\left\|\sum_{j=1}^{\bar{c}}\nabla F_j(\theta_j^t)\right\|^2$$
$$+ B_t\sum_{k=t_c+1}^{t-1}\eta_k^2\left\|\sum_{j=1}^{\bar{c}}\nabla F_j(\theta_j^k)\right\|^2, \tag{C.1}$$

with $\Delta_t = 1 - \mu\eta_t$, $B_t = \frac{\lambda(K+1)\eta_t L^2}{K^2}(C_1 + E)$, and

$$c_t = \eta_t\left[\frac{L\sigma^2}{KB} + U_g^2 \max\left\{\frac{C_1}{K} + 1, C_1 + E\right\}\right] \times \left[\frac{\eta_t}{2} + \frac{L(K+1)}{K}\sum_{k=t_c+1}^{t-1}\eta_k^2\right], \tag{C.2}$$

and we can follow the same reasoning as in [43, Lemma B.7] to see that

$$\mathbb{E}[f(\theta_g^{t+1})] - F^* \leq \Delta_t \mathbb{E}\left[F(\theta_g^t) - F^*\right] + c_t \tag{C.3}$$

for sufficiently large E i.e., the form of c_t does not affect the requirements in regards to the constants to ensure the form of Δ_t. The form of c_t does change the final set of constants in the

result, which are detailed as:

$$\begin{aligned}\mathbb{E}\left[F(\theta_g^t)-F^*\right] \leq \tfrac{a^3}{(T+a)^3}\mathbb{E}\left[F(\theta_g^0)-F^*\right] + \left[\tfrac{4\kappa T(T+2a)}{\mu(T+a)^3} + \tfrac{256\kappa^2 T(E-1)}{\mu(T+a)^3}\right] \\ \times\left(\tfrac{\sigma^2}{KB} + \tfrac{U_g^2}{L}\max\left\{\tfrac{C_1}{K}+1, C_1+E\right\}\right).\end{aligned} \tag{C.4}$$

Thus, the incorrectly identified clients within the cluster add an additional error term proportional to the square of the gradient bound on these clients.

C.1.2 Proof of Theorem 4

We consider summing over the iterations in the bound of Lemma 8 to obtain

$$\begin{aligned}&\tfrac{1}{Tc}\sum_{t=0}^{T-1}\sum_{j=1}^{c}\mathbb{E}[\mathbb{E}_{\mathscr{P}^t}\|\theta_g^t-\theta_j^t\|^2] \leq \tfrac{\lambda\eta^2(2C_1+E(E+1))}{Tc}\tfrac{(K+1)}{K}\sum_{t=0}^{T-1}\left\|\sum_{j=1}^{\bar{c}}\nabla F_j(\theta_j^t)\right\|^2 \\ &+\tfrac{\eta^2(2C_1+E(E+1))}{Tc}\tfrac{(K+1)}{K}\sum_{j=\bar{c}+1}^{c}\|\nabla F_j(\theta_j^t)\|^2 + \tfrac{\eta^2(K+1)(E+1)\sigma^2}{KB} \\ &\leq \tfrac{\lambda\eta^2(2C_1+E(E+1))}{Tc}\tfrac{(K+1)}{K}\sum_{t=0}^{T-1}\left\|\sum_{j=1}^{\bar{c}}\nabla F_j(\theta_j^t)\right\|^2 + \\ &\tfrac{\eta^2(K+1)((E+1)\sigma^2+(C_1+E(E+1))U_g^2)}{KB}.\end{aligned}$$

we can combine this with Lemmas 6 and 7 and proceed the same chain of reasoning as in [43, Theorem 4.4] to obtain:

$$\begin{aligned}\tfrac{1}{T}\sum_{t=0}^{T-1}\left[\mathbb{E}[F(\theta_g^{t+1})]-F(\theta_g^t)\right] &\leq -\tfrac{1}{T}\sum_{t=0}^{T-1}\tfrac{\eta}{2}\|\nabla F(\theta_g^t)\|^2 \\ &+\tfrac{L\eta^2}{2}\left(\tfrac{\sigma^2}{KB}+\left(\tfrac{c}{K}+1\right)U_g^2\right) \\ &+\tfrac{\eta^3L^2(K+1)((E+1)\sigma^2)}{KB} \\ &+\tfrac{\eta^3L^2(K+1)(C_1+E(E+1)U_g^2)}{KB}\end{aligned}$$

from which the final result follows.

Appendix D

Data Summarization

D.1 Proof of Theorem 5

Because P is a bistochastic matrix, and we know $P^* = \frac{1}{\sqrt{n}}1$, we can lower bound

$$1 \geq P(w)^T P^* = \sum_i w_i \frac{P_i^T}{\|P_i\|} \frac{1}{\sqrt{n}} 1 = \frac{1}{\sqrt{n}} \sum_i \frac{w_i}{\|P_i\|} \geq \frac{1}{\min \|P_i\| \sqrt{n}}$$

Similarly, $C(S) \leq C_{max}^k$. Now given $\lambda \leq \frac{1-\kappa}{C_{max}^k \min \|P_i\| \sqrt{n}}$, we compute

$$\begin{aligned}
\max_w P(w)^T P^* - \lambda C(S) &\geq \max_w P(w)^T P^* - \frac{1-\kappa}{\min \|P_i\| \sqrt{n}} \\
&\geq \max_w P(w)^T P^* \left(1 - \frac{1-\kappa}{\min \|P_i\| \sqrt{n}} \frac{1}{P(w)^T P^*}\right) \\
&\geq max_w P(w)^T P^* \left(1 - \frac{1-\kappa}{\min \|P_i\| \sqrt{n}} \min \|P_i\| \sqrt{n}\right) \\
&\geq \kappa \max_w P(w)^T P^*.
\end{aligned}$$

D.2 Auxiliary Lemmas

First, we make a few statements related to initialization of the process. Lemma 3.4 from [143] directly applies to this problem, and thus $\delta_k \in [0,1]\ \forall k$.

Lemma 9. $\langle Pw_1, \frac{1}{n}1 \rangle \geq \frac{\kappa}{\sqrt{n} \sum_i \|P_i\|}$

Proof follows equivalently to Lemma 3.1 from [143], with added caveat that our choice of weights is within κ of maximum value.

Lemma 10. *The cost aware geodesic alignment $\langle a_t, a_{t,v_k}\rangle$ satisfies*

$$\langle a_k, a_{k,v_k}\rangle \geq \kappa\tau\sqrt{J_t} \vee f(t),$$

for

$$f(x) = \kappa \frac{\sqrt{1-x}\sqrt{1-\beta^2}\varepsilon + \sqrt{x}\beta}{\sqrt{1-\left(\sqrt{x}\sqrt{1-\beta^2}\varepsilon - \sqrt{1-x}\beta\right)^2}}$$

and

$$\beta = 0 \wedge \min\langle \ell_n, \frac{1}{\sqrt{n}}1\rangle \text{ s.t. } \langle \ell_n, \frac{1}{\sqrt{n}}1\rangle > -1.$$

Proof. The lemma is equivalent to proving Lemma 3.6 in [143] with one caveat. Here our choice of node is v_k, which comes from choosing the cheapest cost node location from the set $S = \{v \in V | \langle a_k, a_{kv}\rangle \geq \kappa\langle a_k, a_{kv_k}\rangle\}$ Because of this, we can recover all results from $\langle a_k, a_{kv_k}\rangle$ with only a constant κ in front, as our choice satisfies $\langle a_k, a_{kv_k}\rangle \geq \kappa\langle a_k, a_{kv_k}\rangle$.

We apply Lemma 10 to prove the following Theorem that is needed, and mirrors the results from [143].

Theorem 7. *Assume a cost of sensor placement $C(v) : V \to \mathbb{R}_+$ and a slack parameter κ. If we choose the set of points W and weights a_w using Algorithm 12 such that $|W| = K$, then*

$$\|Pw - \frac{1}{n}1\| \leq \frac{\eta\, v_K}{\sqrt{n}},$$

where $v_K = O((1-\kappa^2\varepsilon^2)^{K/2})$ for some ε and $\eta = \sqrt{1-\kappa^2 \max_{i\in V}\left\langle \frac{P_i}{\|P_i\|}, \frac{1}{\sqrt{n}}1\right\rangle^2}$.

Proof. We mimic the results from [143], incorporating the additional cost parameter. We

denote $J_k := 1 - \langle \frac{Pw_k}{\|Pw_k\|}, \frac{1}{\sqrt{n}}1 \rangle$. If we substitute this into the formula for δ_t, we get

$$J_{k+1} = J_k(1 - \langle a_t, a_{kv_k} \rangle^2).$$

Applying our bound from Lemma 10, we get

$$J_{k+1} \leq J_k \left(1 - \kappa^2 \tau^2 J_k\right).$$

By applying the standard induction argument used in [143], we get

$$J_k \leq B(k) := \frac{J_1}{1 + \kappa^2 \tau^2 (k-1)}.$$

Because $B(k)$ still goes to 0, and $f(B(k)) \to \kappa\varepsilon$, there exists a k^* such that $f(B(k)) \geq \kappa\tau\sqrt{B(k)}$, and since f is monotonic decreasing, $f(J_t) > f(B(k))$. Using Lemma 10, we finish with

$$J_k \leq B(k \wedge k^*) \prod_{s=k^*+1}^{k} (1 - f^2(B(s)))$$

We note that $\frac{1}{n} J_k = \|\beta^* P(w) - P^*\|^2$, so this means

$$\|\beta^* Pw - \frac{1}{n}1\| \leq \frac{\eta C_K}{\sqrt{n}},$$

for constant C_K combining the denominator in $B(k)$ and the product of $\prod_{s=k^*+1} k1 - f^2(B(s))$, and $\sqrt{J_1} = \eta$. Notice that $f(B(k)) \to \kappa\varepsilon$ shows a rate of decay of $v = \sqrt{1 - \kappa^2\varepsilon^2}$.

D.3 Proof of Theorem 6

We note that [139] proves multiple bounds on $\left| \frac{1}{n} \sum_{v \in V} f(v) - \sum_{s \in S} w_s f(s) \right|$. The main bound comes from using the fact that they assume $\sum_s w_s = 1$, which allows them to break up the inner product $\|P\sum_s w_s \delta_s - \frac{1}{n}1\|$ into its subsequent terms $\left(\|P\sum_s w_s \delta_w\|^2 - \frac{1}{n}\right)^{1/2}$. We step away from

this assumption and will instead work directly with the following

$$\left\| P\sum_s w_s\delta_s - \frac{1}{n}1 \right\| = \left\| Pw - \frac{1}{n}1 \right\|. \tag{D.1}$$

By the same logic as in [139], we know

$$\left| \frac{1}{n}\sum_{v\in V} f(v) - \sum_{s\in S} w_s f(s) \right| \leq \frac{\|f\|_{P_\lambda}}{\lambda^\ell} \min_{\beta,w} \|\beta P^\ell w - \frac{1}{n}1\|.$$

We can simply replace $\widetilde{P} = P^\ell$ and inherit on $\widetilde{P}$ in Theorem 7, in particular that we still have $\sum_i \frac{1}{n}\widetilde{P}_i = \frac{1}{n}1$. Thus, we can apply the guarantees of Algorithm 12 and Theorem 7 to bound $\|\beta P^\ell w - \frac{1}{n}1\| \leq \frac{\eta v_K}{\sqrt{n}}$ and attain the desired result.

Bibliography

[1] S. Vahidian, S. Kadaveru, W. Baek, W. Wang, V. Kungurtsev, C. Chen, M. Shah, and B. Lin, "When do curricula work in federated learning?" vol. abs/2212.12712, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2212.12712

[2] S. Vahidian, M. Morafah, W. Wang, V. Kungurtsev, C. Chen, M. Shah, and B. Lin, "Efficient distribution similarity identification in clustered federated learning via principal angles between client data subspaces." https://arxiv.org/abs/2209.10526, 2022.

[3] M. Morafah, S. Vahidian, W. Wang, and B. Lin, "FLIS: clustered federated learning via inference similarity for non-iid data distribution," *CoRR*, vol. abs/2208.09754, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2208.09754

[4] S. Vahidian, M. Morafah, and B. Lin, "Personalized federated learning by structured and unstructured pruning under data heterogeneity," in *41st IEEE International Conference on Distributed Computing Systems Workshops, ICDCSW 2021, Washington, DC, USA, July 7-10, 2021*. IEEE, 2021, pp. 27–34. [Online]. Available: https://doi.org/10.1109/ICDCSW53096.2021.00012

[5] S. Vahidian, M. Morafah, C. Chen, M. Shah, and B. Lin, "Rethinking data heterogeneity in federated learning: Introducing a new notion and standard benchmarks," in *NeurIPS 2022 workshop*, 2022.

[6] S. Vahidian, B. Mirzasoleiman, and A. Cloninger, "Coresets for estimating means and mean square error with limited greedy samples," in *Proceedings of the Thirty-Sixth Conference on Uncertainty in Artificial Intelligence, UAI 2020, virtual online, August 3-6, 2020*, ser. Proceedings of Machine Learning Research, R. P. Adams and V. Gogate, Eds., vol. 124. AUAI Press, 2020, pp. 350–359.

[7] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," vol. 382. ACM, 2009, pp. 41–48.

[8] L. Jiang, D. Meng, Q. Zhao, S. Shan, and A. G. Hauptmann, "Self-paced curriculum learning," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, B. Bonet and S. Koenig, Eds. AAAI Press, 2015, pp. 2694–2700.

[9] T. Pi, X. Li, Z. Zhang, D. Meng, F. Wu, J. Xiao, and Y. Zhuang, "Self-paced boost learning for classification," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, S. Kambhampati, Ed. IJCAI/AAAI Press, 2016, pp. 1932–1938.

[10] X. Zhang, G. Kumar, H. Khayrallah, K. Murray, J. Gwinnup, M. J. Martindale, P. McNamee, K. Duh, and M. Carpuat, "An empirical exploration of curriculum learning for neural machine translation," *CoRR*, vol. abs/1811.00739, 2018.

[11] X. Zhang, P. Shapiro, G. Kumar, P. McNamee, M. Carpuat, and K. Duh, "Curriculum learning for domain adaptation in neural machine translation," *CoRR*, vol. abs/1905.05816, 2019. [Online]. Available: http://arxiv.org/abs/1905.05816

[12] J. Yang, S. Shi, Z. Wang, H. Li, and X. Qi, "ST3D: self-training for unsupervised domain adaptation on 3d object detection," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, 2021, pp. 10 368–10 378.

[13] E. Sangineto, M. Nabi, D. Culibrk, and N. Sebe, "Self paced deep learning for weakly supervised object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 3, pp. 712–725, 2019.

[14] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *J. Mach. Learn. Res.*, vol. 21, pp. 140:1–140:67, 2020.

[15] X. Wu, E. Dyer, and B. Neyshabur, "When do curricula work?" in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.

[16] S. Guo, W. Huang, H. Zhang, C. Zhuang, D. Dong, M. R. Scott, and D. Huang, "Curriculumnet: Weakly supervised learning from large-scale web images," in *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part X*, vol. 11214. Springer, 2018, pp. 139–154.

[17] A. Jiménez-Sánchez, M. Tardy, M. Á. G. Ballester, D. Mateus, and G. Piella, "Memory-aware curriculum federated learning for breast cancer classification," *CoRR*, vol. abs/2107.02504, 2021. [Online]. Available: https://arxiv.org/abs/2107.02504

[18] Z. Ren, D. Dong, H. Li, and C. Chen, "Self-paced prioritized curriculum learning with coverage penalty in deep reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 6, pp. 2216–2226, 2018.

[19] Y. Tay, S. Wang, A. T. Luu, J. Fu, M. C. Phan, X. Yuan, J. Rao, S. C. Hui, and A. Zhang, "Simple and effective curriculum pointer-generator networks for reading comprehension over long narratives," in *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, 2019, pp. 4922–4931.

[20] C. Gong, J. Yang, and D. Tao, "Multi-modal curriculum learning over graphs," vol. 10, no. 4, 2019.

[21] Y. Guo, Y. Chen, Y. Zheng, P. Zhao, J. Chen, J. Huang, and M. Tan, "Breaking the curse of space explosion: Towards efficient NAS with curriculum search," in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, vol. 119. PMLR, 2020, pp. 3822–3831.

[22] T. Zhou, S. Wang, and J. A. Bilmes, "Curriculum learning by dynamic instance hardness," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020.

[23] A. Pentina, V. Sharmanska, and C. H. Lampert, "Curriculum learning of multiple tasks," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*. IEEE Computer Society, 2015, pp. 5492–5500.

[24] Z. Jiang, C. Zhang, K. Talwar, and M. C. Mozer, "Exploring the memorization-generalization continuum in deep learning," *CoRR*, vol. abs/2002.03206, 2020. [Online]. Available: https://arxiv.org/abs/2002.03206

[25] M. Toneva, A. Sordoni, R. T. des Combes, A. Trischler, Y. Bengio, and G. J. Gordon, "An empirical study of example forgetting during deep neural network learning," *CoRR*, vol. abs/1812.05159, 2018. [Online]. Available: http://arxiv.org/abs/1812.05159

[26] G. Hacohen and D. Weinshall, "On the power of curriculum learning in training deep networks," in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 2019, pp. 2535–2544.

[27] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[28] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[29] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[31] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1273–1282.

[32] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proceedings of Machine Learning and Systems 2020, MLSys 2020, Austin, March 2-4, 2020*. mlsys.org, 2020.

[33] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, "SCAFFOLD: stochastic controlled averaging for federated learning," in *Proceedings of the 37th International Conference on Machine Learning, ICML*, vol. 119. PMLR, 2020, pp. 5132–5143.

[34] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 7611–7623.

[35] M. Morafah, S. Vahidian, W. Wang, and B. Lin, "FLIS: clustered federated learning via inference similarity for non-iid data distribution," *CoRR*, vol. abs/2208.09754, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2208.09754

[36] S. Vahidian, M. Morafah, and B. Lin, "Personalized federated learning by structured and unstructured pruning under data heterogeneity," *IEEE ICDCS*, 2021.

[37] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on non-iid data silos: An experimental study," *arXiv preprint arXiv:2102.02079*, 2021.

[38] H. Zhu, J. Xu, S. Liu, and Y. Jin, "Federated learning on non-iid data: A survey," *Neurocomputing*, vol. 465, pp. 371–390, 2021.

[39] W. Chen, S. Horváth, and P. Richtárik, "Optimal client sampling for federated learning," *CoRR*, vol. abs/2010.13723, 2020. [Online]. Available: https://arxiv.org/abs/2010.13723

[40] I. Mohammed, S. Tabatabai, A. I. Al-Fuqaha, F. E. Bouanani, J. Qadir, B. Qolomany, and M. Guizani, "Budgeted online selection of candidate iot clients to participate in federated learning," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5938–5952, 2021.

[41] Y. J. Cho, J. Wang, and G. Joshi, "Client selection in federated learning: Convergence analysis and power-of-choice selection strategies," *CoRR*, vol. abs/2010.01243, 2020. [Online]. Available: https://arxiv.org/abs/2010.01243

[42] S. Q. Zhang, J. Lin, and Q. Zhang, "A multi-agent reinforcement learning approach for efficient client selection in federated learning," in *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 2022, pp. 9091–9099.

[43] F. Haddadpour and M. Mahdavi, "On the convergence of local descent methods in federated learning," *arXiv preprint arXiv:1910.14425*, 2019.

[44] Y. Deng, M. M. Kamani, and M. Mahdavi, "Adaptive personalized federated learning," *CoRR*, vol. abs/2003.13461, 2020. [Online]. Available: https://arxiv.org/abs/2003.13461

[45] A. Ajalloeian and S. U. Stich, "On the convergence of sgd with biased gradients," *arXiv preprint arXiv:2008.00051*, 2020.

[46] B. Karimi, B. Miasojedow, E. Moulines, and H.-T. Wai, "Non-asymptotic analysis of biased stochastic approximation scheme," in *Conference on Learning Theory*. PMLR, 2019, pp. 1944–1974.

[47] A. Khaled, K. Mishchenko, and P. Richtárik, "Tighter theory for local sgd on identical and heterogeneous data," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 4519–4529.

[48] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine Learning and Systems*, vol. 2, pp. 429–450, 2020.

[49] F. Zhou and G. Cong, "On the convergence properties of a k-step averaging stochastic gradient descent algorithm for nonconvex optimization," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018, pp. 3219–3227.

[50] B. McMahan and D. Ramage, "Federated learning: Collaborative machine learning without centralized training data," *Google Research Blog*, vol. 3, 2017.

[51] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," *Advances in Neural Information Processing Systems*, vol. 33, pp. 3557–3568, 2020.

[52] P. P. Liang, T. Liu, L. Ziyin, R. Salakhutdinov, and L.-P. Morency, "Think locally, act globally: Federated learning with local and global representations," *arXiv preprint arXiv:2001.01523*, 2020.

[53] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," *CoRR*, vol. abs/2002.10619, 2020.

[54] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," in *Advances in Neural Information Processing Systems 33*, 2020.

[55] F. Sattler, K. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 32, no. 8, pp. 3710–3722, 2021.

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

[57] Y. Jiang, J. Konečný, K. Rush, and S. Kannan, "Improving federated learning personalization via model agnostic meta learning," *CoRR*, vol. abs/1909.12488, 2019.

[58] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4424–4434.

[59] A. R. Gonçalves, F. J. V. Zuben, and A. Banerjee, "Multi-task sparse structure learning with gaussian copula models," *J. Mach. Learn. Res.*, vol. 17, pp. 33:1–33:30, 2016. [Online]. Available: http://jmlr.org/papers/v17/15-215.html

[60] D. Li and J. Wang, "Fedmd: Heterogenous federated learning via model distillation," *CoRR*, vol. abs/1910.03581, 2019. [Online]. Available: http://arxiv.org/abs/1910.03581

[61] H. Chang, V. Shejwalkar, R. Shokri, and A. Houmansadr, "Cronus: Robust and heterogeneous collaborative learning with black-box knowledge transfer," *CoRR*, vol. abs/1912.11279, 2019.

[62] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[63] 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," *CoRR*, vol. abs/2008.06180, 2020. [Online]. Available: https://arxiv.org/abs/2008.06180

[64] C. He, M. Annavaram, and S. Avestimehr, "Group knowledge transfer: Federated learning of large cnns at the edge," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[65] S. Cheng, J. Wu, Y. Xiao, Y. Liu, and Y. Liu, "Fedgems: Federated learning of larger server models via selective knowledge fusion," *CoRR*, vol. abs/2110.11027, 2021. [Online]. Available: https://arxiv.org/abs/2110.11027

[66] 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," *CoRR*, vol. abs/2008.06180, 2020.

[67] W. Jeong, J. Yoon, E. Yang, and S. J. Hwang, "Federated semi-supervised learning with inter-client consistency & disjoint learning," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[68] W. H. Day and H. Edelsbrunner, "Efficient algorithms for agglomerative hierarchical clustering methods," *Journal of classification*, vol. 1, no. 1, pp. 7–24, 1984.

[69] P. P. Liang, T. Liu, L. Ziyin, R. Salakhutdinov, and L.-P. Morency, "Think locally, act globally: Federated learning with local and global representations," *arXiv preprint arXiv:2001.01523*, 2020.

[70] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," *CVPR*, vol. abs/2103.16257, 2021.

[71] D. hyun Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," 2013.

[72] Y. Ge, D. Chen, and H. Li, "Mutual mean-teaching: Pseudo label refinery for unsupervised domain adaptation on person re-identification," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[73] Q. Xie, Z. Dai, E. H. Hovy, T. Luong, and Q. Le, "Unsupervised data augmentation for consistency training," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[74] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. Raffel, E. D. Cubuk, A. Kurakin, and C. Li, "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[75] Y. Zou, Z. Yu, B. V. K. V. Kumar, and J. Wang, "Unsupervised domain adaptation for semantic segmentation via class-balanced self-training," in *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III*, ser. Lecture Notes in Computer Science, vol. 11207. Springer, 2018, pp. 297–313.

[76] J. Yang, S. Shi, Z. Wang, H. Li, and X. Qi, "ST3D: self-training for unsupervised domain adaptation on 3d object detection," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*. Computer Vision Foundation / IEEE, 2021, pp. 10 368–10 378.

[77] G. Qian, S. Sural, Y. Gu, and S. Pramanik, "Similarity between euclidean and cosine angle distance for nearest neighbor queries," in *Proceedings of the 2004 ACM symposium on Applied computing*, 2004, pp. 1232–1237.

[78] A. Talwalkar, S. Kumar, M. Mohri, and H. Rowley, "Large-scale svd and manifold learning," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 3129–3152, 2013.

[79] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 723–773, 2012.

[80] J. R. Hershey and P. A. Olsen, "Approximating the kullback leibler divergence between gaussian mixture models," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, vol. 4, 2007, pp. IV–317–IV–320.

[81] D. Hallac, J. Leskovec, and S. P. Boyd, "Network lasso: Clustering and optimization in large graphs," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*. ACM, 2015, pp. 387–396.

[82] Y. Sarcheshmehpour, M. Leinonen, and A. Jung, "Federated learning from big data over networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*. IEEE, 2021, pp. 3055–3059.

[83] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," 2011.

[84] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2016, pp. 770–778.

[85] J. J. Hull, "A database for handwritten text recognition research," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 16, no. 5, pp. 550–554, 1994.

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

[87] H. J. Rad, M. Abdizadeh, and A. Szabó, "Federated learning with taskonomy for non-iid data," *CoRR*, vol. abs/2103.15947, 2021.

[88] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proc. of Int'l Conf. on Computer Vision (ICCV)*, December 2015.

[89] T. Kailath, "The divergence and bhattacharyya distance measures in signal selection," *IEEE transactions on communication technology*, vol. 15, no. 1, pp. 52–60, 1967.

[90] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, "Federated learning for mobile keyboard prediction," *arXiv preprint arXiv:1811.03604*, 2018.

[91] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. F. Wang, and J.-B. Huang, "A closer look at few-shot classification," in *Int'l Conf. on Learning Representations (ICLR)*, 2019.

[92] "support.google. your chats stay private while messages improves suggestions," *https://support.google.com/messages/answer/9327902*, Aug., 2019.

[93] A. WWDC, "Apple. designing for privacy (video and slide deck)," *https://developer.apple.com/ videos/play/wwdc2019/708*, 2019.

[94] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[95] T. Yu, E. Bagdasaryan, and V. Shmatikov, "Salvaging federated learning by local adaptation," *arXiv preprint arXiv:2002.04758*, 2020.

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

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

[98] K. Wang, R. Mathews, C. Kiddon, H. Eichner, F. Beaufays, and D. Ramage, "Federated evaluation of on-device personalization," *arXiv preprint arXiv:1910.10252*, 2019.

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

[100] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[101] F. Hanzely and P. Richtárik, "Federated learning of a mixture of global and local models," *arXiv preprint arXiv:2002.05516*, 2020.

[102] "speedtest.net. speedtest market report." *http://www.speedtest.net/reports/united-states/*, 2016.

[103] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[104] Y. Han, A. Özgür, and T. Weissman, "Geometric lower bounds for distributed parameter estimation under communication constraints," *arXiv preprint arXiv:1802.08417*, 2018.

[105] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," *arXiv preprint arXiv:1712.01887*, 2017.

[106] A. Reisizadeh, H. Taheri, A. Mokhtari, H. Hassani, and R. Pedarsani, "Robust and communication-efficient collaborative learning," in *Advances in Neural Information Processing Systems*, 2019, pp. 8388–8399.

[107] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[108] K. C. Sim, P. Zadrazil, and F. Beaufays, "An investigation into on-device personalization of end-to-end automatic speech recognition models," *arXiv preprint arXiv:1909.06678*, 2019.

[109] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," *arXiv preprint arXiv:1608.08710*, 2016.

[110] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," *arXiv preprint arXiv:1803.03635*, 2018.

[111] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[112] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," in *Advances in neural information processing systems*, 2016, pp. 2074–2082.

[113] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2736–2744.

[114] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *European conference on computer vision*. Springer, 2016, pp. 646–661.

[115] Y. LeCun, C. Cortes, and C. Burges, "Mnist handwritten digit database," *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, vol. 2, 2010.

[116] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[117] G. Cohen, S. Afshar, J. Tapson, and A. V. Schaik, "Emnist: Extending mnist to handwritten letters," *2017 International Joint Conference on Neural Networks (IJCNN)*, 2017.

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

[119] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 2015, pp. 1310–1321.

[120] N. Hynes, D. Dao, D. Yan, R. Cheng, and D. Song, "A demonstration of sterling: A privacy-preserving data marketplace," *Proc. VLDB Endow.*, vol. 11, no. 12, pp. 2086–2089, 2018.

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

[122] H. Wang, M. Yurochkin, Y. Sun, D. S. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[123] D. A. E. Acar, Y. Zhao, R. M. Navarro, M. Mattina, P. N. Whatmough, and V. Saligrama, "Federated learning based on dynamic regularization," *International Conference on Learning Representations*, 2021.

[124] T. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *CoRR*, vol. abs/1909.06335, 2019.

[125] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," *CoRR*, vol. abs/2006.07242, 2020.

[126] M. Yurochkin, M. Agarwal, S. Ghosh, K. H. Greenewald, T. N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 2019, pp. 7252–7261.

[127] W. Hao, M. El-Khamy, J. Lee, J. Zhang, K. J. Liang, C. Chen, and L. Carin, "Towards fair federated learning with zero-shot data augmentation," *CoRR*, vol. abs/2104.13417, 2021.

[128] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S. Kim, "Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data," *CoRR*, vol. abs/1811.11479, 2018. [Online]. Available: http://arxiv.org/abs/1811.11479

[129] J. Goetz and A. Tewari, "Federated learning via synthetic data," *CoRR*, vol. abs/2008.04489, 2020. [Online]. Available: https://arxiv.org/abs/2008.04489

[130] M. Luo, F. Chen, D. Hu, Y. Zhang, J. Liang, and J. Feng, "No fear of heterogeneity: Classifier calibration for federated learning with non-iid data," in *Advances in Neural Information Processing Systems, NeurIPS 2021, December 6-14*, 2021, pp. 5972–5984.

[131] D. K. Dennis, T. Li, and V. Smith, "Heterogeneity for the win: One-shot federated clustering," in *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, vol. 139. PMLR, 2021, pp. 2611–2620.

[132] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *Int. J. Comput. Vis.*, vol. 40, no. 2, pp. 99–121, 2000.

[133] S. Kornblith, M. Norouzi, H. Lee, and G. E. Hinton, "Similarity of neural network representations revisited," in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, ser. Proceedings of Machine Learning Research, vol. 97. PMLR, 2019, pp. 3519–3529.

[134] A. Coates, A. Ng, and H. Lee, "An Analysis of Single Layer Networks in Unsupervised Feature Learning," in *AISTATS*, 2011, https://cs.stanford.edu/~acoates/papers/coatesleeng_aistats_2011.pdf.

[135] W. Lippmann, *Public opinion*. Routledge, 2017.

[136] Y. Abbasi, P. L. Bartlett, V. Kanade, Y. Seldin, and C. Szepesvari, "Online learning in markov decision processes with adversarially chosen transition probability distributions," in *NeurIPS*, 2013.

[137] A. Krause, J. Leskovec, C. Guestrin, J. VanBriesen, and C. Faloutsos, "Efficient sensor placement optimization for securing large water distribution networks," *Journal of Water Resources Planning and Management*, 2008.

[138] L. Yu, N. Wang, and X. Meng, "Real-time forest fire detection with wireless sensor networks," in *International Conference on Wireless Comm., Networking and Mobile Computing*, vol. 2. IEEE, 2005, pp. 1214–1217.

[139] G. Linderman and S. Steinerberger, "Numerical integration on graphs: Where to sample and how to weigh," 2020.

[140] A. Anis, A. Gadde, and A. Ortega, "Efficient sampling set selection for bandlimited graph signals using graph spectral proxies," *IEEE Trans. Sig. Proc.*, 2016.

[141] R. K. Iyer and J. A. Bilmes, "Submodular optimization with submodular cover and submodular knapsack constraints," in *NeurIPS*, 2013.

[142] A. Cloninger, "Bounding the error from reference set kernel maximum mean discrepancy," in *arXiv preprint arXiv:1812.04594*, 2018.

[143] T. Campbell and T. Broderick, "Bayesian coreset construction via greedy iterative geodesic ascent," in *ICML*, 2018.

[144] S. Fortunato, "Community detection in graphs," *Physics reports*, vol. 486, no. 3-5, pp. 75–174, 2010.

[145] A. Lancichinetti and S. Fortunato, "Community detection algorithms: a comparative analysis," *Physical review E*, vol. 80, no. 5, p. 056117, 2009.

[146] S. Navlakha, R. Rastogi, and N. Shrivastava, "Graph summarization with bounded error," in *ACM SIGMOD Inter. Conf. on Management of data*. ACM, 2008.

[147] D. A. Spielman and N. Srivastava, "Graph sparsification by effective resistances," *SIAM Journal on Computing*, vol. 40, no. 6, pp. 1913–1926, 2011.

[148] K. J. Ahn, S. Guha, and A. McGregor, "Graph sketches: sparsification, spanners, and subgraphs," in *ACM SIGMOD PoDS*. ACM, 2012.

[149] E. Liberty, "Simple and deterministic matrix sketching," in *ACM SIGKDD Inter. Conf. on KDD*. ACM, 2013, pp. 581–588.

[150] M. Mathioudakis, F. Bonchi, C. Castillo, A. Gionis, and A. Ukkonen, "Sparsification of influence networks," in *ACM SIGKDD Inter. Conf. on KDD*, 2011.

[151] S. Har-Peled and S. Mazumdar, "On coresets for k-means and k-median clustering," in *ACM Symposium on Theory of Computing*. ACM, 2004, pp. 291–300.

[152] M. Lucic, M. Faulkner, A. Krause, and D. Feldman, "Training gaussian mixture models at scale via coresets," *JMLR*, vol. 18, no. 1, pp. 5885–5909, 2017.

[153] M. B. Cohen, C. Musco, and C. Musco, "Input sparsity time low-rank approximation via ridge leverage score sampling," in *ACM-SIAM SODA*. SIAM, 2017, pp. 1758–1777.

[154] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan, "Approximating extent measures of points," *Journal of the ACM*, 2004.

[155] M. Li, G. L. Miller, and R. Peng, "Iterative row sampling," in *IEEE FoCS*. IEEE, 2013, pp. 127–136.

[156] C. Musco and C. Musco, "Recursive sampling for the nystrom method," in *NeurIPS*, 2017, pp. 3833–3845.

[157] K. L. Clarkson, "Coresets, sparse greedy approximation, and the frank-wolfe algorithm," *ACM TALG*, vol. 6, no. 4, p. 63, 2010.

[158] R. R. Coifman and S. Lafon, "Diffusion maps," *ACHA*, vol. 21, no. 1, pp. 5–30, 2006.

[159] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *ACHA*, vol. 30, no. 2, pp. 129–150, 2011.

[160] A. d'Aspremont, F. R. Bach, and L. E. Ghaoui, "Optimal solutions for sparse principal component analysis," *JMLR*, 2008.

[161] A. d'Aspremont, L. E. Ghaoui, M. I. Jordan, and G. R. G. Lanckriet, "A direct formulation for sparse PCA using semidefinite programming," *SIAM Review*, 2007.

[162] M. Pilanci, L. E. Ghaoui, and V. Chandrasekaran, "Recovery of sparse probability measures via convex programming," in *NeurIPS*, 2012, pp. 2420–2428.

[163] A. Kyrillidis, S. Becker, V. Cevher, and C. Koch, "Sparse projections onto the simplex," in *ICML*, 2013, pp. 235–243.

[164] E. J. Candes *et al.*, "The restricted isometry property and its implications for compressed sensing," *Comptes rendus mathematique*, 2008.

[165] S. Lacoste-Julien and M. Jaggi, "On the global linear convergence of frank-wolfe optimization variants," in *NeurIPS*, 2015, pp. 496–504.

[166] J. Lu, C. D. Sogge, and S. Steinerberger, "Approximating pointwise products of laplacian eigenfunctions," *Journal of Functional Analysis*, vol. 277, no. 9, pp. 3271–3282, 2019.

[167] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *CoRR*, vol. abs/1609.02907, 2016. [Online]. Available: http://arxiv.org/abs/1609.02907

[168] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[169] W. Aiello, F. C. Graham, and L. Lu, "A random graph model for power law graphs," *Experimental Mathematics*, vol. 10, no. 1, pp. 53–66, 2001.

[170] J. Leskovec and J. J. Mcauley, "Learning to discover social circles in ego networks," in *NeurIPS*, 2012, pp. 539–547.

[171] P. Jain, P. Netrapalli, and S. Sanghavi, "Low-rank matrix completion using alternating minimization," in *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, D. Boneh, T. Roughgarden, and J. Feigenbaum, Eds. ACM, 2013, pp. 665–674.

[172] S. U. Stich, "Local sgd converges fast and communicates little," in *ICLR 2019-International Conference on Learning Representations*, no. CONF, 2019.