Federated Learning for Heterogeneous Networks: Algorithmic and System Design

Hongda Wu

A Dissertation submitted to the Faculty of Graduate Studies
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

Graduate Program in Electrical Engineering and Computer Science
York University
Toronto, Ontario

December 2023

Abstract

Building reliable machine learning models depends on access to data samples. With the increasingly advanced sensing and computing capabilities on edge devices, the ever-stringent data privacy legislation, and growing user privacy concerns, it is crucial to build learning models from separate, heterogeneous data sources without violating user privacy. Federated Learning (FL) can facilitate collaborative machine learning without accessing user-sensitive data and has emerged as an attractive paradigm for mobile edge networks. However, federated optimization builds on a heterogeneous environment, which brings challenges beyond traditional distributed learning. Though FL is viewed as a promising technique for enabling intelligent applications, the current FL system suffers from high communication costs, restricting it from being applied in mobile edge networks. To fully release the potential, the FL design must be communication-efficient, adaptive, and robust to the heterogeneous training environment.

In this thesis, we aim to address the practical challenges of FL in a conscientious manner. Particularly, we try to understand and address some of those challenges in federated networks and build FL systems that fulfill the accuracy, efficiency, and robustness requirements. Starting with the primary challenge, i.e., data heterogeneity, we study how it impacts the model accuracy and communication cost in the collaborative training system. To address this concern, we develop new and scalable algorithms that can quantify the contribution from participating devices, thus alleviating the negative impact of data heterogeneity and reducing the overall communication burden. To handle another major challenge, i.e., the heterogeneity of computation capabilities among different types of edge devices, we devise a new sub-model

training method to enable devices with heterogeneous computation capabilities to participate in and contribute to the FL system, making it robust to the straggler effect. The proposed solutions are rigorously compared with popularly adopted benchmarks from theoretical and empirical perspectives. Finally, we provide a preliminary discussion on personalized FL and point out the potentially interesting research directions in the related fields. Although the proposed methods and designs originate from the practical application of FL, the theoretical insights gained from this thesis can be extended to a broader context of trustworthy machine learning.

To my parents and grandparents

Acknowledgement

First and foremost, I am deeply thankful to my supervisor, Professor Ping Wang, for her invaluable guidance and support during my PhD study. The amazing Ph.D. journey would have never been basked in my achievements without her expertise, patience, and encouragement. She guided me through every aspect of research with great patience, enthusiasm, and vision. Her commitment to excellence and rigor has been a constant source of inspiration. I am genuinely grateful for the opportunity to work under her mentorship, and I appreciate the time and dedication she has devoted to enriching my academic journey.

I am also grateful to Professor Aijun An and Professor Uyen Trang Nguyen for serving as my supervisory committee members and offering constant encouragement and advice. Furthermore, I would like to thank my thesis examining committee members, Professor Lian Zhao, Professor Gunho Sohn, and Professor Manos Papagelis, for their time and valuable feedback in improving the dissertation's quality. Their precious time and efforts devoted to this dissertation are highly appreciated.

I am grateful to have gained more insight into practical challenges in the industry through my coop at Montreal Research Centre and the guidance and collaboration with researchers there: Dr. Guojun Zhang, Dr. Xi (Alex) Chen, Ke Sun, and Dr. Sobhan Hemati. I would like to thank Sony R&D, Japan, for offering me the internship opportunity and the guidance and mentoring from Maruyama Shinya, Jimbo Masanobu, and Dr. Hamza Jeljeli.

My Ph.D. life wouldn't have been so fulfilling without the help from my friends within and outside of York U. Fortunately, I received constant care and support from them that helped me go through the tough times.

Finally, I wish to express my deepest gratitude to Erin for her love and for standing firmly by my side. I am indebted to express my gratitude to my parents and grandparents for their unconditional love, support, and understanding. This thesis is dedicated to them.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ANNs Artificial neural networks

BP Back propagation

DL Deep learning

DGD Deterministic gradient descent

DRL Deep reinforcement learning

ERM Empirical risk minimization

FL Federated learning

GS Gradient sparsification

i.i.d. Independent and identically distributed

IST Independent subnet training

KD Knowledge distillation

MAC Multi-access channel

MEC Mobile edge computing

ML Machine learning

MLR Multinomial logistic regression

Non-i.i.d. Non-Independent-identically-distributed

OFDMA Orthogonal frequency division multiple access

OTA Over-the-air

SGD Stochastic gradient descent

UAV Unmanned aerial vehicles

US-Nets Universal slimmable networks

Chapter 1

Introduction

1.1 Federated Learning

With rapid advancement, edge devices (e.g., sensors, mobile phones, and connected vehicles) constantly generate an unprecedented amount of data [1]. These devices are equipped with enhanced sensors, computing, and communication capabilities. Coupled with the rise of Deep Learning (DL), the edge devices unfold countless opportunities for various tasks of modern society, e.g., road congestion prediction [2] and perceptive control (e.g., Unmanned Aerial Vehicles (UAVs) swarm navigation [3].

In the traditional cloud-centric approaches, data generated and collected by edge devices is uploaded and processed in a data center. It is predicted that the data generation rate will exceed the capacity of today's Internet in the near future [4]. Mobile Edge Computing (MEC) has naturally been proposed to incorporate data processing outside the cloud. With computing and storage capability, MEC systems formulate an end-edge-server architecture. Multiple edge servers can perform large-scale distributed tasks involving local processing and remote execution under the coordination of a remote cloud. MEC approaches balance training efficiency and communication cost by bringing model training toward where the data

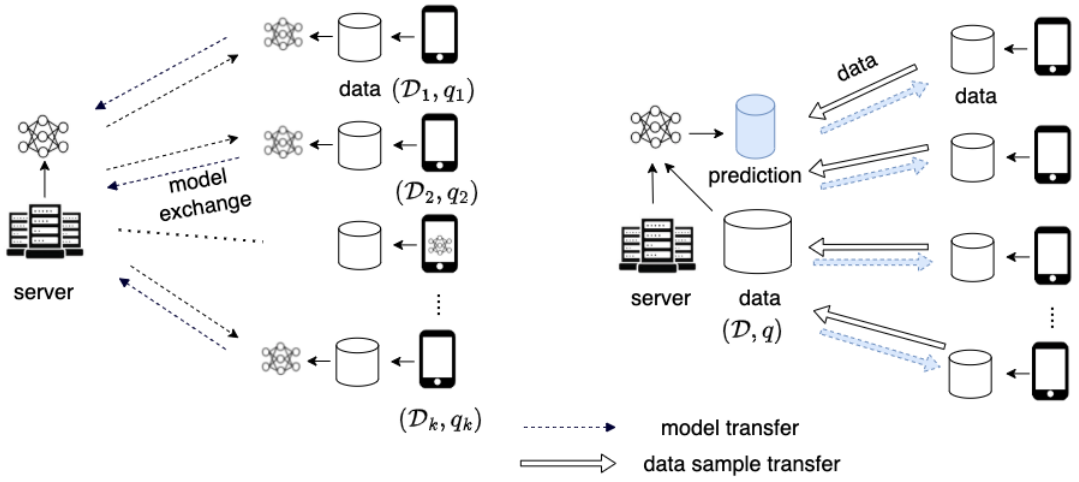

Figure 1.1: Schematic model learning comparison: the edge-server based federated learning system vs. traditional centralized machine learning system. By exploiting the computation potential of edge devices, federated learning differentiates itself from conventional learning approaches from data acquisition, storage, and training, making privacy-preserving model learning and communication efficiency possible.

is generated. However, computation offloading and data processing at the edge server still involve sensitive data transmission.

In either centralized cloud training or MEC approaches, collecting data for model training is unrealistic from a privacy, security, regulatory, or necessity perspective. To maintain privacy-sensitive data and to facilitate collaborative Machine Learning (ML) among distributed devices, *Federated Learning* (FL) has emerged as an attractive paradigm, where local devices collaboratively train a task model under the orchestration of a central server without accessing end-user data [5, 6]. In FL, local devices cooperatively train an ML model required by the central server by utilizing their local data. By transferring local model updates to the central server for model aggregation and acquiring a global model for local training rather than sending raw data, user data privacy is well protected. FL distinguishes itself from conventional approaches in data acquisition, storage, and training, has been deployed by major service providers, and plays a vital role in supporting privacy-sensitive applications, including computer vision for autonomous vehicles, natural language processing tasks for

mobile users, and health monitoring with wearable devices [7].

1.2 Challenges of Federated Networks

FL unfolds the commensal opportunity for collaborative training and privacy protection. Meanwhile, FL also involves several unique characteristics that are listed as follows.

1) **Expensive Communication**. FL is capable of producing highly accurate statistical models by consolidating knowledge from diverse data sources. Communication costs in FL become the bottleneck most of the time. i) Model optimization in the FL context involves iterative message transmission (e.g., model update in each communication round) instead of sending entire raw data over the network. For complex DL models, e.g., ResNet, which is laborious to optimize, each round of model exchange involves millions of parameters [8]. The high dimensional update results in communication costs. ii) Generally, federated networks and the learning process compromise a massive number of devices, among which the communication cost is even prohibited. For example, millions of smartphones on the task of Next-Word Prediction [9], or modern IoT networks consisting of wearable sensors, autonomous vehicles, and rode-side units, sensors of smart city with massive connectivity [10] [11]. To reduce the overall communication cost in FL, two main aspects are focused on: i) designing faster convergence algorithms to reduce the communication round. ii) reducing the size of the transmitted parameters at each round.

2) **Statistical Heterogeneity**. Devices generate and collect data samples with an underlying independent preference, which may not be associated with amongst devices. For example, IoT devices capture different ambient information varying from location. Unlike traditional centralized ML or distributed ML, where the model training can access all data samples or run on independent and identically distributed (i.i.d) data samples from a large dataset, this prerequisite is impractical for FL since the local dataset is only accessible by

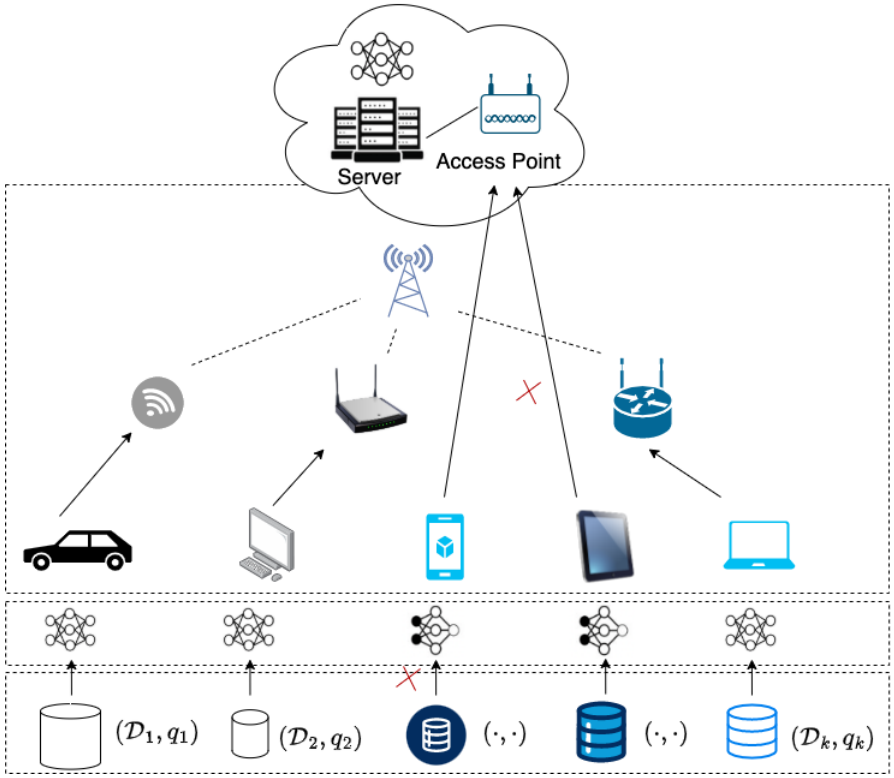

Figure 1.2: Federated networks with heterogeneous edge devices and network identities. The system, model, and statistical heterogeneity are shown in different dashed boxes from top to bottom. The FL training can be interrupted by either the communication or the local computation process, exacerbating the accuracy and communication cost issue. The overarching goal of this thesis is to develop principled approaches to improve communication efficiency and system robustness by considering these heterogeneities.

the local device. Therefore, the participating devices may have local datasets that follow different data distributions, i.e., non-independent and identically distributed (non-i.i.d) data samples across participating devices, making it difficult for distributed model optimization to converge.

The canonical FL problem aims to optimize the surrogated objectives of local devices, where the statistical heterogeneity incurs discrepancy between the overall (global) learning objective and local objectives, i.e., minimizing the local objective may not positively contribute to the global objective minimization. The convergence rate of the FL model over non-i.i.d.

data is slow compared with the case of i.i.d. data. Data heterogeneity stagnates model convergence, reduces the model accuracy substantially, and invokes additional communication rounds to resource-constrained edge devices [12, 13].

3) **System Heterogeneity**. Due to the system-related constraints and network size, FL results in a small portion of devices being active in each global round, randomly selected from a massive number of candidate devices [5]. However, the candidate devices are heterogeneous regarding network connectivity (4G, 5G, wifi), variability in hardware (CPU, memory, energy), channel condition, and even the willingness to participate. It is not surprising that active devices drop out in the optimization process due to the energy level or connectivity. The straggler effect exacerbates the learning stagnation and delays the convergence. Overall, such system-level heterogeneity aggravates the challenge of robust FL system design for straggler mitigation and fault tolerance.

1.3 Motivation and Research Contribution

Initially brought by Google in 2016, privacy-preserved federated learning has emerged as a promising technology to embrace machine learning in many applications. However, the vanilla federated averaging (`FedAvg`) [5] algorithm is affected by many factors in real deployments, including slow convergence in statistical heterogeneity scenarios, straggler effect due to system heterogeneity, and a natural trade-off between model generalization and personalization, etc. To be practically useful, a federated learning approach needs to deliver an accurate model with efficiency in training and robustness to statistical and system heterogeneity. **This thesis aims to understand and address some of these challenges in federated networks and build FL systems that fulfill the accuracy, efficiency, and robustness requirements.**

The main research contribution is covered in chapters 3 - 5. Although the proposed

methods and designs originate from practical applications of FL, the theoretical insights gained from this thesis can be extended to many more scenarios.

Chapter 3: Quantifying the Device Contribution in Global Aggregation. `FedAvg` is the de facto optimization method due to its simplicity. However, it behaves inefficiently in non-i.i.d. scenarios. We propose `Fed`erated `Ada`ptive Weighting (`FedAdp`) [14] algorithm that aims to improve the FL learning performance through assigning distinct weights for participating devices in global model aggregation. We observe that devices with heterogeneous datasets contribute differently. Therefore, our main intuition is to measure the contribution of the participating device based on the gradient information, then assign different weights accordingly and adaptively at each communication round for global model aggregation. With theoretical analysis, we show how weighting impacts the expected training loss decrement of the learning objective, where the weighting strategy can be tuned accordingly. We empirically evaluate the learning performance of `FedAdp` and compare it with commonly adopted `FedAvg` via extensive experiments. While the algorithmic modification is minor and in a simple full-participation setting, `FedAdp` untangles the direction to speed up FL training in non-i.i.d. scenarios by quantifying device contribution.

Chapter 4: Improving the FL Convergence by Probabilistic Device Selection. Random device selection [5] is effective in general i.i.d. settings but poses learning difficulty in non-i.i.d. datasets due to the misalignment between the global objective and local objectives. A careful device selection strategy is necessary to reveal the FL scenario where large-scale devices with non-i.i.d. datasets are potentially involved. We propose a Probabilistic Device Selection framework, `FedPNS` [15], with contribution-related criteria to choose active devices in each global round. To align local model updates with minimizing the global objective, we use `Optimal Aggregation` to determine the optimal subset of local updates of the participating devices in global model aggregation, From which the data heterogeneity can be profiled. The result of `Optimal Aggregation` is further used to adjust the probability for each device to

be selected in the subsequent global rounds. `FedPNS` involves minor calculations on the server side, does not impose additional communication costs, and is easy to implement in a scalable fashion.

Chapter 5: Handling the Computation Heterogeneity by Partial Model Training. The implicit assumption that all devices are capable of doing model training and exchanging model information is unrealistic in building robust FL systems. To accommodate different types of devices with heterogeneous computational capabilities, *model-heterogeneous* FL is proposed in this chapter, where participants are allowed to train models with different complexity (i.e., the subset of a learning model). We propose a novel layer-wise model-splitting method to match the device's capability so that the involved devices can contribute to the global model and avoid the straggler effect. We answer the question that *given limited computation and/or communication, which part of the whole model should be updated and/or protected in FL?* Theoretically, we find the proposed partial model training strategy enjoys a similar convergence rate to FedAvg in strongly-convex and smoothness case. However, by allowing adaptive partial model allocation, all participating devices can contribute to the global model punctually, making the task completion time shorter and the FL system robust to straggler effects and training disruption.

1.4 Outlines of the Thesis

The remainder of this thesis is organized as follows. Chapter 2 presents preliminaries of federated optimization and a literature review of algorithmic designs on improving the convergence rate under the above-mentioned challenges. Chapters 3 and 4 present two works to handle the statistical heterogeneity in FL. Particularly, we propose using an adaptive weighting strategy in Chapter 3 to aggregate the contribution-different local updates to improve the global model convergence. In Chapter 4, we propose a probabilistic device

scheduling design for local training. The proposed design can preferentially select devices that propel faster model convergence and can be easily extended to a large-scale fashion. To handle the computation heterogeneity on local devices, a partial model training strategy is proposed in Chapter 5, which allows participating devices with lower computation capability to train and contribute to parts of the designated model, thus better utilizing local resources and avoiding the straggler effect caused by computation heterogeneity. The superiority of all proposed methods in Chapter 3 - 5 are theoretically analyzed, empirically verified, and compared with existing benchmarks. Finally, Chapter 6 summarizes the thesis and points out the social impact and future directions in this research field.

1.5 Publication List

The author's publications in the doctoral study are listed below.

Journal and Conference papers:

[J1] **Hongda Wu**, Ping Wang, "Fast-Convergent Federated Learning with Adaptive Weighting," *IEEE Transactions on Cognitive Communications and Networking*, vol.7, no.4, pp. 1078-1088, 2021. [14]

[J2] **Hongda Wu**, Ping Wang, "Node Selection Toward Faster Convergence for Federated Learning on Non-IID Data," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 5, pp. 3099-3111, 2022. [15]

[J3] **Hongda Wu**, Ping Wang, C V Aswarth Narayana, "Straggler-resilient Federated Learning: Tackling Computation Heterogeneity with Layer-wise Partial Model Training in Mobile Edge Network," under review, *IEEE Transactions on Mobile Computing*

[C1] **Hongda Wu**, Ping Wang, "Fast-convergent Federated Learning with Adaptive Weighting," *IEEE Conference on Communication* (**ICC 2021**), Montreal, QC, Canada [16]

[C2] **Hongda Wu**, Ping Wang, "Probabilistic Node Selection for Federated Learning with Heterogeneous Data in Mobile Edge," *IEEE Wireless Communications and Networking Conference* (**WCNC 2022**), Austin, TX, USA [17]

The Best Paper Award WCNC 2022 (1 of 4 out of 742 submissions)

[C3] **Hongda Wu**, Ping Wang, C V Aswarth Narayana, " Model-heterogeneous Federated Learning with Partial Model Training,"*IEEE International Conference on Communications in China* (**ICCC 2023**), Dalian, China [18]

Other publications that are not involved in this thesis:

[J4] **Hongda Wu**, Ali Nasehzadeh, Ping Wang, "A Deep Reinforcement Learning-Based Caching Strategy for IoT Networks with Transient Data,"*IEEE Transactions on Vehicular Technology*, vol. 71, no. 12, pp. 13310-13319, Dec. 2022.

[J5] Shufeng Li, Mingyu Cai, Libiao Jin, Yao Sun, **Hongda Wu**, Ping Wang, "An Ultra-Reliable Low-Latency Non-Binary Polar Coded SCMA Scheme," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 6, pp. 6518-6533, June 2022.

Technique Report:

[T1] **Hongda Wu**, Guojun Zhang, "On Personalized Federated Learning Approaches: A Technical Report," Montreal Research Centre, Feb. 2023

Chapter 2

Preliminaries and Literature Review

2.1 Preliminaries

2.1.1 Canonical Federated Learning

In general, federated learning methods [5, 19, 20] are designed to handle the consensus learning task (learning a *single, global* model) in a decentralized manner, where a central server coordinates the global learning objective, and multiple devices train the model with locally collected data. Consider the network with a set of local devices $|\mathcal{K}|$ (i.e., $k \in \{1, 2, \cdots, |\mathcal{K}|\}$), the goal in federated optimization is to learn a parametric model $\mathbf{w}$ that fits data samples in a distributed setting by minimizing some loss function $F(\mathbf{w})$. Formally, the FL objective is to minimize a surrogated function as follows

$$\min_{\mathbf{w}} F(\mathbf{w}) := \sum_{k=1}^{|\mathcal{K}|} \frac{|\mathcal{D}_k|}{\sum_{k=1}^{|\mathcal{K}|} |\mathcal{D}_k|} F_k(\mathbf{w}). \tag{2.1}$$

We assume each local device $k \in \mathcal{K}$ has a training set $\mathcal{D}_k$ that follows a data distribution q_k, i.e., each sample $z_{k,1}, z_{k,2} \cdots z_{k,|\mathcal{D}_k|}$ is drawn from q_k distribution randomly, where each of which consists of a pair of feature and response denoted by $z_{k,s} = \{\mathbf{x}_{k,s}, y_{k,s}\}$. Let $\ell(\mathbf{w}; z_{k,s}) : \Theta \to \mathbb{R}$

be the loss function associated with data sample $z_{k,s}$, where $\Theta = \mathbb{R}^d$ is the parameter space. The population loss function for each device k is defined as $F_k(\mathbf{w}) := \mathbb{E}_{z_{k,s} \backsim q_k}[\ell(\mathbf{w}; z_{k,s})]$. Because each device has a small number of data samples, population distribution on the device is not fully observed. Instead of minimizing the population loss function, each device targets the Empirical Risk Minimization (ERM) problem defined as

$$F_k(\mathbf{w}) = \frac{1}{|\mathcal{D}_k|} \sum_{z_{k,s} \in \mathcal{D}_k} \ell(\mathbf{w}; z_{k,s}). \tag{2.2}$$

Further, for general classification problem with cross-entropy loss, we have local objective represented as follows

$$\begin{aligned} F_k(\mathbf{w}) &= \mathbb{E}_{z_{k,s} \backsim q_k}\left[-\sum_{j=1}^{C} \mathbb{1}_{y=j} \log l_j(\mathbf{w}, z_{k,s})\right] \\ &= \frac{1}{|\mathcal{D}_k|} \sum_{z_{k,s} \in \mathcal{D}_k} -\left(\sum_{c=1}^{C} q_k(y_{k,s} = c) \mathbb{E}_{\mathbf{x}_{k,s}|y_{k,s}=c}\left[\log l_c(\mathbf{w}, \mathbf{x}_{k,s}, y_{k,s})\right]\right), \end{aligned} \tag{2.3}$$

where $\log l_c(\mathbf{w}, \mathbf{x}_{k,s}, y_{k,s})$ denotes the probability that the data sample $\{\mathbf{x}_{k,s}, y_{k,s}\}$ is classified as the c-th class given model $\mathbf{w}$. $q_k(y_{k,s} = c)$ denotes the data distribution on device k over class $c \in [C]$.

2.1.2 Federated Averaging with Partial Device Participation

The most commonly used algorithm to solve (2.1) is Federated Averaging (`FedAvg`) [5], which enables the partial devices participation in the server side, and applies a Stochastic Gradient Descent (SGD) optimizer on local function $F_k(\cdot)$ with the same learning rate across all devices. `FedAvg` process is viewed as a variant of SGD with multiple global rounds, where each global round consists of multiple steps of local update (e.g., τ steps) followed by model synchronization process between participating devices and the server. Denoting

$t = \{1, 2, \cdots, T\}$ as the index of FL global rounds, one round of FedAvg is described as follows

1. The server randomly samples a subset of devices $\mathcal{S} \subseteq \mathcal{K}$ (with a pre-defined fraction $\check{c}$ to control the algorithm efficiency, where $|\mathcal{S}| = \check{c}|\mathcal{K}|$) and broadcasts the latest model $\mathbf{w}^t$ to the selected devices $k \in \mathcal{S}$.

2. Each selected device views $\mathbf{w}^t$ as an initial model, updates it by τ steps of SGD[1] over its empirical risk objective defined in (2.2), and sends the updated model $\mathbf{w}_k^{t+1}$ back to the server[2].

3. The server aggregates received local models $\mathbf{w}_k^t, k \in \mathcal{S}$ with weight a_k and gets the new global model $\mathbf{w}^{t+1}$, where a_k can be defined by different criteria, e.g., proportional to data size $|\mathcal{D}_k| / \sum_{k=1}^{|\mathcal{S}|} |\mathcal{D}_k|$ in [5].

The above steps are repeated until a satisfying learning result, e.g., the learning accuracy in classification tasks, is met.

2.1.3 `FedAvg` for non-IID data

The independent and identically distributed (i.i.d.) sampling condition of training data is important, which ensures that the stochastic gradient is an unbiased estimate of the full gradient [21]. `FedAvg` is shown to be effective even with simple aggregation, given that the data distribution across different devices is the same as centrally collected data. McMahan *et al.* [5] show that tuning the optimization parameter to achieve fast model convergence is important. Particularly, the number of local epochs E plays a crucial role. On one hand,

[1]According to the vanilla `FedAvg`, $\tau = \frac{|\mathcal{D}_k|}{B} E$ is related to the number of local epochs E and batch size B.

[2]Typically, there are two ways for devices to upload their local model to the server, either by uploading model parameters $\mathbf{w}_k^t$ or by uploading the model difference Δ_k^{t+1}. Although the same amount of data is to be sent in both ways, conveying Δ_k^{t+1} is proven to be more amenable for compression [6]. We will be using $\mathbf{w}_k^{t+1}$ and Δ_k^{t+1} interchangeable in this thesis.

devices executing multiple epochs E (so as τ) can greatly improve the convergence speed and alleviate the communication burden in federated networks, compared to Federated SGD, where each local device transmits model update after only one step of (full) gradient descent.

However, on the other hand, a large E might slow down the convergence in non-i.i.d. FL training where heterogeneous local objectives are involved. The data distribution determined by usage patterns across local devices is typically non-i.i.d., i.e., p_k is different across participating devices. Since local objective $F_k(\mathbf{w})$ is closely related to data distribution p_k, a large number of local updates lead the model towards optima of its local objective $F_k(\mathbf{w})$ as opposed to the global objective $F(\mathbf{w})$, which might be potentially different. The inconsistency between local models $\mathbf{w}_k$ and global model $\mathbf{w}$ is accumulated along with local training, causing training instability that makes the FL model struggle to converge. As such, local training with multiple local updates potentially hurts convergence and even leads to divergence with the presence of non-i.i.d. data [5, 12, 20, 22].

We conducted a simple experiment to demonstrate the impact of non-i.i.d. data on model convergence. We trained a two-layer convolutional neural network (CNN) model with the same neural network architecture in [5] using Pytorch on the MNIST dataset (containing 60,000 samples with 10 different types of handwriting digits from 0 to 9 and each sample has 28 $\times$ 28 pixel) until the model achieves 95% test accuracy[3]. 10 devices are selected, each with 600 samples that are selected based on their label criteria. If a device is at *i.i.d. setting*, 600 samples are randomly selected over the whole training set. If a device is at *x-class non-i.i.d. setting*, 600 samples are randomly selected over a subset composed of x class data samples. Each x-class is selected at random and can be overlapped. The skewness of datasets is measured and reflected by the value of x.

We use the same notations for `FedAvg` algorithm as [5]: B, the local minibatch size, and

[3]In this thesis, we following the definition of test accuracy in [5], which is achieved at the server side, and is defined as the proportion between the number of correctly-classified data samples and the number of total test samples on the held-out test dataset of the used dataset, e.g., MNIST.

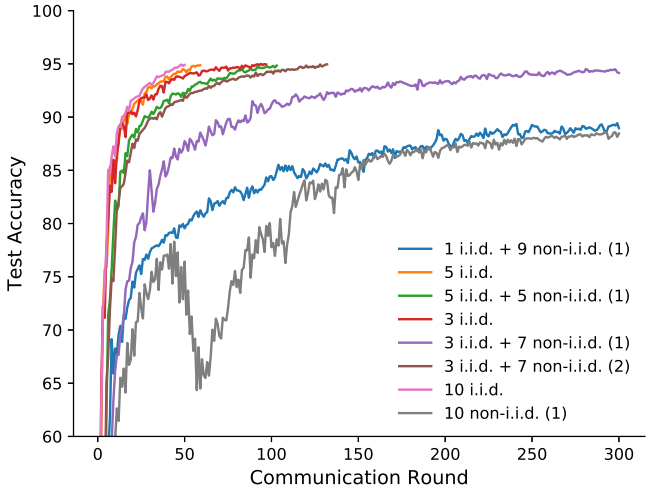

Figure 2.1: Test accuracy over communication rounds of `FedAvg` with heterogeneous data distribution over participating devices. X i.i.d. + Y non-i.i.d. (1) (or (2)) represents X devices are at *i.i.d. setting* and Y devices are at *1-class (or 2-class) non-i.i.d. setting*.

E, the number of local training epochs. In this experiment, $B = 32$, $E = 1$, $\eta = 0.01$ and learning rate decay of 0.995 per communication round. We can conclude from Fig. 2.1

- Model convergence highly depends on i.i.d. devices. The presence of non-i.i.d. devices imposes variance to model training, which slows the convergence of FL (e.g., 5 i.i.d. case converges faster than 5 i.i.d. + 5 non-i.i.d. (1) case).

- The skewness of data affects model convergence. With the participation of the non-i.i.d. device, the model converges much slower when the skewness of the dataset increases (e.g., 3 i.i.d. + 7 non-i.i.d. (2) case converges much faster than 3 i.i.d. + 7 non-i.i.d. (1) case).

The statistical heterogeneity is one of the most significant characteristics that differentiate FL from traditional federated optimization from theoretical and empirical perspectives. In what follows, we introduce some assumptions generally used in the convergence rate analysis of FL.

2.1.4 Assumptions for Convergence Analysis

In this part, we spell out all the assumptions that are adopted for the convergence analysis of federated optimization (or distributed machine learning, in a broad perspective) in this thesis. The listed assumptions mainly focus on local function in FL and the SGD optimization of local objectives.

Assumption 1. ***μ-strong convexity***. *$F_k(\mathbf{w}), k \in \mathcal{S}$ is μ-strong convex, i.e. $F_k(\mathbf{w}) \geq F_k(\mathbf{w}') + (\mathbf{w} - \mathbf{w}')^\top \nabla F_k(\mathbf{w}') + \frac{\mu}{2}\|\mathbf{w} - \mathbf{w}'\|^2$, for all $\mathbf{w}, \mathbf{w}'$, where $(\cdot)^\top$ denotes the transpose operation of vector.*

Assumption 2. ***L-smoothness***. *$F_k(\mathbf{w}), k \in \mathcal{S}$ is L-smooth, i.e. $F_k(\mathbf{w}) \leq F_k(\mathbf{w}') + (\mathbf{w} - \mathbf{w}')^\top \nabla F_k(\mathbf{w}') + \frac{L}{2}\|\mathbf{w} - \mathbf{w}'\|^2$, for all $\mathbf{w}, \mathbf{w}'$ (or $\|\nabla F_k(\mathbf{w}) - \nabla F_k(\mathbf{w}')\| \leq L\|\mathbf{w} - \mathbf{w}'\|$ in another form).*

Assumption 3. ***Bounded local gradient***. *The expected squared norm of the local stochastic gradient is bounded, i.e., $\mathbb{E}\|\nabla F_k(\mathbf{w}_k^t, \xi_k)\|^2 \leq G_k^2$ for all device $k \in \mathcal{S}$ and $t = 1, 2, \cdots, T$, and $G_k^2 \leq G^2, \forall k \in \mathcal{K}$.*

Please note that assumptions 1 and 2 are made to the local loss function, which can be satisfied when the logistic regression with cross-entropy loss is adopted. More examples include ℓ_2-norm regularized linear regression with mean square error and the support vector machine with hinge loss. Based on Assumptions 1 and 2, the definition of $F(\mathbf{w})$, and triangle inequality, we can easily derive that the global objective $F(\mathbf{w})$ satisfies μ-strong convex and L-smoothness conditions.

Assumption 3 is made to the result of SGD optimizer and local objective, and a similar assumption has been made in previous works [20, 22–25]. Besides, with $\mathbf{w}$ trained by heterogeneous data, G_k is different for different devices, which is closely related to the data distribution on each device. If the data distribution on device k is more similar to the

population distribution over all devices, G_k is smaller, and vice versa. This observation is empirically illustrated in Chapter 4.

2.2 Federated Learning for Wireless Network

To build a feasible and satisfying FL model, a number of rounds of communication between local devices and the server is required. Employing high dimensional models [8] results in enormous communication costs. Additionally, the heterogeneous device identity, communication network (4G, 5G, WiFi), and unstable network conditions cause straggler effects, making communication the bottleneck of the FL network and the obstacle for FL to be implemented in real scenarios. To improve the communication efficiency in FL, one can reduce the communication round by improving the convergence speed of model training (discussed in Section 2.2.1 and 2.2.3) or reducing the actual transmission cost of wireless links in each round of communication (see Section 2.2.3).

FL tasks typically involve $10^1 - 10^6$ devices and $10^2 - 10^4$ global rounds [26], and each of the global rounds has multiple configurable hyperparameters for both participating devices (e.g., the number of local updates) and the server (e.g., device scheduling policy). As such, it is *infeasible to seek the entire configuration space* for a global optimum systematic design. In what follows, we separate the existing literature aiming at FL convergence optimization into two streams, where the main idea applies on either the device side or the server side, and discuss the representative method separately. It is worth mentioning that many of those methods are proposed for scenarios with specific features, for example, to handle statistical and/or system heterogeneity, which will be emphasized in the following discussion.

2.2.1 Improving the Convergence via Device-level Optimization

We categorize the device-level methods into **Tuning Local Update**, **Local Optimizer**, and **Local Objective**, as discussed in the following.

Using a simple averaging and local SGD, `FedAvg` shows effectiveness in consensus learning with decentralized devices [5]. Particularly, the number of global rounds can be significantly reduced by allowing flexibility in the number of local updates before synchronization, as opposed to the conventional approach `FedSGD`, where synchronization is done after every single step of local update. Li *et al.* [19] proposed to allow participating devices to perform a variable number of local updates rather than applying the same amount of workload for each device [5] to overcome the system heterogeneity. Similar to [19], authors in [27] also posed local accuracy for participating devices, based on limited local computation resources, as an index to steer the number of local updates performed. Authors in [28] proposed using local accuracy as the criteria to accommodate the different computational capabilities of local devices. Instead of applying a unified configuration (e.g., local epoch) to all participating devices, the server will assign different local accuracies to participating devices depending on computational capability. Once the criteria are satisfied locally, the updated model will be transmitted to the server to avoid the potential straggler effect caused by hardware constraints. Unlike [19, 27, 28], the work in [21] exhibited an analytical model to dynamically adapt the number of local updates between two consecutive global aggregations in real-time to minimize the learning loss under a fixed resource budget of the edge computing system. Authors in [29] proposed SCAFFOLD to control the difference between the optimization direction of the local and global objectives (a.k.a. device drift). By exploiting the control variate (the estimation of updating the global model and local model), a correction technique is applied to overcome the data heterogeneity. SCAFFOLD has no assumption on data heterogeneity and device sampling and is proven to converge in significantly fewer rounds of communication. In

addition, there is a series of works focusing on sub-net (a subset of neural network) training [30, 31] instead of updating the whole learning model so that the straggler effect due to the limited computation or communication is alleviated. We left the corresponding discussion in Section 2.2.3.

Similar to general machine learning tasks, using an advanced optimizer instead of SGD in a federated network results in faster model convergence and is beneficial in reducing the communication rounds. In particular, rather than using SGD optimizer, authors in [32] adopted the momentum SGD to local training. Compared to local SGD optimizers in FL, employing momentum SGD leads to a linear speedup in the convergence rate w.r.t. the number of participating devices. Additionally, the number of communication rounds needed in FL with momentum SGD optimizer is analyzed for both i.i.d. and non-i.i.d. cases. FedAC [33] also applied momentum at the local device with periodic synchronization, which is proven to obtain similar linear speed-up properties with asymptotically fewer rounds of synchronization. Instead of using SGD with momentum as the local optimizer as in [32, 33], Liu *et al.* [34] proposed to add a momentum term to Deterministic Gradient Descent (DGD), which can realize more accurate training results with improved generalization and fast convergence. With a similar convergence analysis as in [21], authors claim that the convergence rate of momentum-DGD optimizer outperforms SGD-based FL under specific conditions.

Instead of minimizing the data sample-related loss function in (2.2), existing works add a regularization term to form a new local objective and improve the convergence rate of FL designs. With a generic ℓ_2 regularization applied on different models [19], i.e., the Euclidean distance between local models and the global model, each device is able to optimize its model to better align with the direction of minimizing the global objective. The proposed FedProx can control the side effects brought by data heterogeneity. FedDANE [35] adopted a similar regularization term and formulated another term in local objective by following the Distributed Approximate Newton method in distributed optimization. To solve the

discrepancy between minimizing the local and global objectives, FedDyn [36] considered two extra terms in addition to data-related loss as the local objective, including ℓ_2 regularization and a linear term which is formulated to align with the devices' empirical loss surface. In theory, FedDyn ensures that the consensus point of model convergence across devices remains consistent with the global stationary solutions, as long as the local models converge regardless of the data heterogeneity. Authors in [37] proposed to use model contrastive learning as the regularization term. The proposed regularization loss is composed of the representation distance of the local and global models, where a validation dataset is needed on the local side. Recently, Qu *et al.* [38] proposed applying Sharpness Aware Minimization [39] to the local solver and formulating the Federated form (FedSAM), which seeks to find a small perturbation to be added to the learning model. The added perturbation results in a local perturbed loss function, and minimizing the new local objective is beneficial in reducing distribution shift and improving the generalization capability of the global model.

2.2.2 Improving the Convergence via System-level Optimization

Focusing on device-level optimization becomes imperative where the data and hardware conditions are heterogeneous. Meanwhile, the server-side systematic designs responsible for **Model Aggregation**, **Device Selection**, and **Synchronization** are essential to overall system performance, as discussed in the following.

Model Aggregation. A natural way to improve the convergence rate of data-heterogeneous FL is to quantify the difference between on-device data distribution and the population (overall) data distribution over the federated network. Zhao *et al.* [12] quantified the weight divergence by Earth Mover's Distance, which is profiled based on the data distribution difference. However, pushing a small set of uniformly distributed data to participating devices in [12] violates the privacy concern of FL and imposes extra communication costs.

Authors in [40] proposed to adaptively assign different weights for global model aggregation by considering the time difference when the model update is done layerwise asynchronously. To reduce the communication cost, the asynchronous FL updates the shallow and deep layers of the neural network model at different frequencies. So, the model aggregation considers the time difference, and the server assigns the most recently updated layers a higher weight in aggregation. Similarly, authors in [41] suggested using age-aware aggregation in asynchronous FL. Local devices send the model update to the server in different frequencies, making the server aggregate the received model update difficult. On one hand, favoring the older updates potentially balances the participating frequency among devices and reduces the risk of the training model being biased toward the device with stronger computation capability. On the other hand, favoring fresher models makes model convergence faster but at the risk of leading to a biased global model with weak generalization capability. Thus, a variable related to the device selection algorithm is added in model aggregation to mitigate the update asynchrony. Further, Chai *et al.*[42] designed a tier-based FL system by dividing the participating devices into tiers according to their responding time. The server assigns weights to different tiers for model aggregation since varying update frequencies exist across tiers. Both methods in [40–42] focus on the asynchronous FL and aim to weigh the local update along with different communication rounds.

In addition to learning a global model, model aggregation can also be extended to the personalized FL context [43, 44], where the personalized local model can control how to adaptively aggregate model information from other devices. Zhang *et al.* [43] proposed a similar idea to quantify the contribution from other devices (by global model) and combine them with a set of designed weights for personalization. Particularly, the proposed FedFOMO estimates how much the global model performs differently on the local device compared to the same inference result with local models. The weight to update the local model in each global round is measured by the division of performance difference (i.e., the difference of

loss values) and ℓ_2 distance between two models. To achieve the best personalization model, devices weigh the models that most efficiently improve their local performance. Similar to [43], Beaussart *et al.* [44] proposed to use different sets of weights for aggregation, with a specific focus on personalization. In contrast to [43], the proposed WAFFLE has the server to tune the weights for each device. These weights are computed based on the similarity and the current degree of personalization in each global round. The server calculates the Euclidean distance for each device by comparing its model update to the updates from other devices. The farther away the device's distance from others, the smaller weights the device assigns when aggregating other's model updates.

Device Selection. In vanilla FL design [5, 12, 19, 29], a subset of devices is involved to improve the training efficiency, which is chosen randomly from a large number of candidate devices in federated networks. The quality of chosen devices in terms of the on-device training data and hardware capability are important for FL convergence. Hence, a thoughtfully designed device selection process proves advantageous for enhancing performance.

Several works have been carried out focusing on device selection design to improve the FL convergence rate, taking the system heterogeneity and uncertainty of wireless medium into consideration [45–49]. Specifically, Nishio *et al.* [45] proposed to select devices intentionally based on the resource condition on devices. Amiria *et al.* [46] designed a device scheduling algorithm by considering the significance of local update measured by ℓ_2-norm and channel condition separately or jointly. For example, in `BN2` algorithm [46], the server first selects a macro set of devices to participate in local training. Subsequently, a subset of the macro set is chosen for model aggregation by ordering the norm of gradient transmitted from devices of the macro set. In [50], the authors proposed biased client selection strategies, that is, preferentially choosing the device with higher local loss. Though the contribution-related loss measurement leads to a faster convergence, the selection skewness imposes potential error, and the local loss measurement results in additional communication and computation costs.

Differently, references [47–49, 51, 52] focus on probabilistic device selection strategy where each device is eligible to contribute to the global model, making them better balance the exploitation and exploration. Particularly, the contribution and importance-based device selections with different sampling criteria were proposed. Authors in [47] considered the limited bandwidth resource for model transmission where device selection for global model aggregation is of importance. The proposed method measures the device contribution according to the norm of local updates, by which the probability for each device to be selected is calculated so as to execute the device selection procedure. The devices with a higher norm of local updates are chosen with higher probability, thus boosting the convergence rate when limited bandwidth resource is provided. Along with [47], authors in [48] proposed to use Artificial Neural Networks (ANNs) as a predictor to estimate the model updates of devices that are not allocated the bandwidth for transmission, based on the model updates that are successfully transmitted using limited bandwidth resource. The additionally included model updates further accelerate the model convergence. Authors in [49] proposed a probabilistic design by considering the importance of local update and transmission latency, where the importance of local update is evaluated by gradient divergence between local gradients and the ground truth global gradient. The probability for device selection is finally determined by the local gradient norm and transmission latency. Chen *et al.* [51] designed an importance sampling scheme that selects more informative devices. The device sampling procedure minimizes the variance of local gradients for aggregation, while the probability for each device to be chosen is proportional to the norm of local updates. In addition, authors in [52] applied importance sampling for device selection on the server level and data selection on the device level. Similar to [51], the optimal device selection is achieved by minimizing the bound on the variance of gradient noise, i.e., the estimation error of the global gradient because of the partial device participation. The probability for each device to be chosen is proportional to the norm of its local updates.

Similar to probabilistic device selection, Deep Reinforcement Learning (DRL) based methods have been proposed by learning a device selection policy that can fasten the convergence. Wang *et al.* [13] proposed to train the DRL agent by adopting the dimension-reduced model updates and the global update as the state information. The agent is supposed to judiciously select devices that may contribute to global model improvement so that the reward signal is designed in order to improve the global model accuracy per global round aggressively. A similar concept is applied to an energy-constrained FL framework [53]. The optimization target in [53] is to select a subset of participants to guarantee the model quality while maximizing energy efficiency for each individual participating device. One side effect of the DRL-based method is that the RL agent is trained with tens to hundreds of fully observed FL training processes, which might not be easy to realize and justifiable in practice.

Asynchronous FL. As a distributed learning paradigm, the model synchronization protocol is crucial in FL. Traditional FL designs synchronize the model until all local model updates are received, which may cause a long delay due to system heterogeneity. Otherwise, the synchronization frequency can be fixed, and the server drops off the model updates that can not be delivered on time by (staleness) devices, causing model convergence to be unstable. To mitigate the negative effect caused by staleness devices, [54] proposed to use an asynchronous manner to improve the system efficiency. The server updates the global model whenever a local update is received. However, there is an inconsistency in the asynchronous update scheme when devices come to obtaining model parameters from the server. Global feature representation on the server and a dynamic learning step size for local training is proposed to alleviate the inconsistency. The inconsistency in the asynchronous scheme is viewed as the delayed gradient in [55]. To mitigate the error caused by delayed gradient, Zheng *et al.* leveraged Taylor expansion of the gradient function and efficient approximation to the Hessian matrix of the loss function to reconstruct the gap between the real gradient and delayed gradient. The proposed scheme achieves system efficiency because no device

needs to wait for others. In the meanwhile, the delay compensation eases the side effect of asynchronous inconsistency. Xie *et al.* [56] used an asynchronous update and added a regularization term in the local objective. Similarly to [40], the inconsistency problem is alleviated by weighting the device's updates in a time-dependent manner. In this way, the contribution from devices that take a very long time in model aggregation is reduced. Authors in [57] studied the empirical performance of synchronous and asynchronous-based `FedAvg`. Results show that asynchronous-based FL design outperforms the synchronous FL in terms of the task completion time for both i.i.d. and non-i.i.d. distribution. Additionally, the experiment shows that the proposed asynchronous FL is robust to real-world situations where devices join pathways through training or train at different speeds. Huba *et al.* [58] presented `Papaya`, a framework that supports both synchronous and asynchronous model aggregation for a large-scale simulation. In the next-word prediction task with massive candidate devices (100 million Android phones), asynchronous FL is 5 times faster and has nearly 8 times less communication overhead than synchronous FL in high concurrency settings.

It is worth noticing that some server-side optimization methods can be combined with device-level methods to achieve better results or overcome specific obstacles. Instead of achieving a faster convergent model and reducing the communication round, there are many other works directly focusing on the transmission cost of the wireless link, as discussed in Section 2.2.3.

2.2.3 Communication Cost Reduction over Wireless Link

To reduce the communication cost on wireless links, existing works focus on exploring the superposition property of the wireless medium, low-precision quantization technique, and model sparsification.

Over-the-air Aggregation

The superposition nature of wireless channels allows gradients/model updates to be aggregated Over-the-Air (OTA) and favors communication much more efficiently[59]. Communication overheads are significantly decreased through the simultaneous transmission of all local updates with shared communication resources. This approach can be categorized into digital or analog schemes depending on which signal form (e.g., gradient) is transmitted over the channel [60] [61]. In [60], the authors proposed a novel analog communication scheme, where local devices first sparsify their gradient estimates while accumulating error from previous iterations and project the resultant sparse vector into a low-dimensional vector for bandwidth reduction. Authors in [61] designed digital gradient transmission schemes, where gradients at each device are first quantized and then transmitted over a multi-access-channel to be decoded individually at the server.

Zhu *et al.* in [62] proposed a multi-access Broadband Analog Aggregation (BAA) scheme for communication-latency reduction, building on the concept of OTA computation. BAA strategy identifies the trade-off that SNR improvement is at the cost of truncating model parameters from the devices with a longer propagation distance in wireless networks, which needs to be compromised by device scheduling. Simulation results show that the BBA algorithm can reduce latency by 10 to 1000 times compared to FL design with traditional Orthogonal Frequency Division Multiple Access (OFDMA) while maintaining the same model accuracy. Though OTA is effective in reducing the wireless link burden, the superposition signal is sensitive to transmission noise, which will essentially impact the local training, especially for the near-convergent process where the gradient value is small. Thus, authors in [63] proposed to use time-varying precoding that gradually mitigates the contribution of the channel noise over time. To further reduce the communication cost, Yang *et al.* [64] proposed to use second-order information in the local training process, where the second-order (Hessian

matrix) information is combined with first-order (gradient) information by local Newton step [65], and finally transmitted with OTA.

Quantization and Compression

A probabilistic quantization approach was proposed in [66], in which the update matrices are vectorized and quantized for each model parameter. To reduce the error from quantization, a structured random rotation based on the Walsh-Hadamard and binary diagonal matrix can be applied before quantization. An extension [67] was made based on lossy compression on the global model sent server-to-client and the dropout technique in the collaborative learning context. Authors in [68] proposed communication-efficient FedAvg (CE-FedAvg). By adopting *Adam* as optimizer and model compression, CE-FedAvg reduces the number of communication rounds taken to reach a target accuracy and the total data uploaded per round compared to the uncompressed case in `FedAvg`. In addition, for hyper-parameter value in *Adam*, authors proposed to use the Uniform and Exponential Quantization method to reduce transmission cost. These two schemes are used alongside sparsification and Golomb encoding for the compression in CE-FedAvg.

Unlike [60], where the analog scheme was adopted, authors in [61] designed digital gradient transmission schemes, where gradients at each device are first quantized and then transmitted over a Multi-access Channel (MAC) to be decoded individually at the aggregator. A stochastic gradient quantization scheme in [61] was proposed, where quantization parameters are optimized based on the capacity region of MAC. Different from the Uniform and Exponential Quantization method in [68], where the quantization gap is fixed, the stochastic multi-level quantization scheme adopts the dynamic range of the gradient vector. Convergence analysis shows that in order to maximize the convergence speed, devices whose gradients have a higher dynamic range must be assigned a higher quantization budget. Similarly, in [69], the quantization error is analyzed subject to the uplink transmission delay and outage constraint

in each FL global round. By joint allocating the quantization bits and wireless resources, each device can have the same transmission outage when minimizing the quantization error, making the design more suitable for the federated network with transmission latency constraints. Zheng *et al.* [70] analyzed the differences when applying the quantization technique to model weight and gradient. Theoretical results show that transmitting the weight requires increasing the quantitation level with a logarithmic rate, compared to a constant quantization level, which is needed to transmit gradients/model updates. The conclusion provides an in-depth insight into the bandwidth-accuracy tradeoff of FL design.

Sparsification

Under the original `FedAvg` algorithm where either full gradient or nothing (e.g., because of a network outage, transmission delay, etc.) is sent, a more balanced approach is Gradient Sparsification (GS)[71], which sends a sparse vector with partial information of the model and provides a higher degree of freedom to achieve better communication and computation trade-offs. To reduce the transmission cost, sending the most important part of the gradient vector is straightforward, so-called *top* GS, where the parameters with the highest absolute values are selected and transmitted, and the global model parameters are not evenly updated in the meantime. To mitigate this effect, in each global round, *periodic averaging* GS randomly selects and transmits a subset of gradient elements. After a finite number of rounds, all elements of the full gradient can be aggregated at least once. However, the variance raised by *top* and *periodic averaging* GS makes the convergence property insecure. Authors in [72] proposed to use online learning to learn the near-optimal sparsity in FL, namely, FAB-*top*. Instead of choosing several highest absolute values in each round, FAB-*top* utilizes the accumulated gradient, from which the highest absolute values are chosen. Then, the server aggregates the selected index-value pairs from local devices and identifies the sparsity. Further, an *fairness* matrice is adopted to ensure that contribution is from every participating

device.

Authors in [73] unified and identified the popular sparsification methods on SGD-based FL as facets of general sparsification methods that can operate on any possible *atomic decomposition*, for example, on element-wise and single value. With a given gradient, a sparsity budget, and an atomic decomposition, the proposed framework ATOMO is able to give a random unbiased sparsification of the atoms with minimized variance. Further, the sparsification method has also been used in serverless network topology FL design [74]. Particularly, GossipFL was proposed in [74], where each device only needs to exchange a highly compressed model with a single peer at every communication round. Meanwhile, the authors designed a gossip matrix generation algorithm that can better utilize the bandwidth resources while preserving the convergence property.

2.3 Model-Heterogeneous Federated Learning

In FL scenarios, the participating devices are naturally computation-heterogeneous, which may only be capable of training models that align with their on-device resources. It is not trivial to consider model-heterogeneous FL design, where the server assigns different models that align with the devices' capabilities. To aggregate information from heterogeneous models, existing literature can be categorized into two main streams: knowledge distillation and partial model training, and the latter one is generically explored in the general machine learning context before FL, such as slimmable neural network.

Slimmable Neural Network

Yu *et al.* [75] proposed to train a Slimmable Neural Network, i.e., several model variants (with a switch to control the model width) where the parameters on different variants are shared, and their individual information is kept by individual batch normalization layers. Further,

authors in [76] proposed Universal Slimmable Networks (US-Nets), which makes slimmable neural networks more generalized with any model width. Both Slimmable Neural Network and US-Nets aim to train several models simultaneously. In essence, the above methods are so-called model-parallel training, where a portion of the learning model is partitioned across different computing devices. When data is generated with individual features, model-parallel training implies that different parts of the model can only be updated to reflect the present data on any device. A fine-grained synchronization step is needed. Differently, Yuan *et al.* [77] adopted the idea of individual subnet training called Independent Subnet Training (IST), where a large neural network is evenly divided into non-joint subnets and updated individually on different devices. IST focuses on cases where communication/memory is limited on a single device. Since no synchronization is required during local updates, per-step communication volume on multiple devices can be reduced. After synchronization, model parameters are re-distributed based on a new random sampling method, and the IST process repeats. Generally, the learning model in slimmable neural network [75–77] is evenly split into different devices. Though the flexibility to match devices' resources is provided, the applicable scenario is limited, and the convergence guarantee is not observed in FL scenarios. Consider a more general setting where a partial of the neural network model is masked from updating. Authors in [78] analyzed the convergence rate of model training with partial-gradient using a generic template called partial SGD (the combination of parameter perturbation and gradient masking). The theoretical results can be used as a guide to create new training methods.

Knowledge Distillation for Federated Learning

One primary technique to exchange information between differently structured models is Knowledge Distillation (KD), where devices with less computational capability enjoy information from large models [79]. In FL, KD is used to transfer knowledge for both homogeneous models [80, 81] and heterogeneous models [82]. Notably, FedDF proposed by

Lin *et al.* [80] first trains several classifiers on local private data. Local classifiers then go through public unlabeled data and generate logits. Each device's information is treated as a "teacher" whose information is aggregated into the global model ("student") to improve its generalization capability. Similarly, authors in [81] proposed a cluster-based knowledge transfer where each device sends the logits (generated from local public data) back to the server, in which devices can be grouped, and local models are aggregated to different global models based on similarity. With multiple clusters, the proposed method is beneficial to alleviate the data heterogeneity problem.

It is worth mentioning local models in COMET [81] are not necessarily homogeneous. FedGKT [82], proposed by He *et al.*, extends KD to heterogeneous model scenarios. Local devices update the lightweight models in each global round, whose information is sent, aggregated, and integrated into a large model on the server side via KD. The server's model could be larger than any local model. Meanwhile, with soft labels from server-side training, local models' performance is boosted by adopting the KD-based loss. To remove the dependence of public data in KD-based FL, authors in [83] proposed to learn a generative model at the server side, which is solely derived from the output of local devices. Given target labels in the local side, the learned generator yields a feature representation that is consistent with the ensemble of each device's output. Though the generator provides information from other peer devices, transferring the generator is necessary in each round, incurring more communication costs.

Partial Model Training in FL

Authors in [23] introduced the US-Nets to the FL context for the first time and proposed superposition coding and successive decoding to protect different parts of the learning model. In [23], only 0.5× model width (left/right model) is considered, and the learning performance is improved when including the partial model, which the designed superposition

coding/successive decoding strategy can guarantee. Study [23] focuses on the model transmission but requires local devices to train the whole model, which is impractical for devices with heterogeneous computational capabilities. Authors in [30] proposed a computation and communication-efficient FL design for heterogeneous devices by allowing local models to have different architectures from the global model. Different from [23, 75, 76], the proposed *HeteroFL* [30] grants local devices various model architectures (size) according to their computational capabilities and allows weak devices in terms of computation/ communication to contribute to the global model in FL. HeteroFL enables sub-model generation in a static way where sub-models are extracted from a designated part of the global model. Inspired by Dropout [79] in centralized machine learning, it is straightforward to adopt Dropout to FL to alleviate the resource-constrained local computing. As illustrated in [31], the FL server randomly removes a subset of neurons and generates sub-models for the participating device to meet its computation level. Similar to the static sub-model generation as in [30], authors in [84] proposed FjORD, which combines the sub-model training and knowledge distillation to improve the sub-model performance. As stated in [85], both [30, 31, 84] suffer from performance degradation on high data heterogeneity. This is primarily due to the limitation that different sub-models can only be updated on specific devices that match their computation level, forcing different parts of the global model to be updated on samples with different distributions.

However, the performance of Federated Dropout suffers from highly heterogeneous data and the devices cohort when generating a partial model randomly. To overcome the drawback of random [31] and static [30, 84] partial-model generation, authors in [85] proposed using a rolling window to counteract uneven model updates by which all parts of the global model are looped in sequence. This rolling process iterates each round until the global model is evenly trained to converge.

Chapter 3

Fast-Convergent Federated Learning with Adaptive Weighting

The first question we would like to ask in this thesis is: How does statistical heterogeneity affect federated optimization in a simple *full-device participation scenario*, and how can we improve the convergence speed in such circumstances? This chapter presents answers to these questions by exploring the statistical heterogeneity.

3.1 Overview

Federated Learning (FL) has emerged as an attractive paradigm, where local devices collaboratively train a task model under the orchestration of a central server without accessing end-device data [7, 86]. Two major challenges separate FL from traditional distributed optimization: statistical and system heterogeneity, making it difficult to apply to many scenarios. In an attempt to handle the heterogeneity and reduce the communication burden in FL, existing works allow multiple local updates [5] or, even in an adaptive way [21], on a subset of total devices ($\mathcal{S} \subset \mathcal{K}$) in a federated network. In particular, Federated Averaging

(`FedAvg`) [5] is the de facto optimization method in the federated setting, which employs a small number of participating devices to run E epochs of local updates before exchanging model information via the server, where all local models are simply averaged.

Even though the good performance of `FedAvg` is shown empirically, owing to the highly skewed data across local devices and non-i.i.d. distribution in FL, communication cost becomes a critical bottleneck in the FL context since generally several iterations are involved for model convergence [5–7]. The presence of non-i.i.d. data significantly degrades the global model performance, which makes model training take more rounds to converge, and the variance caused by non-i.i.d. data brings instability to the training process [12, 13].

To surmount the slow convergence of vanilla `FedAvg` under the presence of non-i.i.d. dataset, we propose `Fed`erated `Ada`ptive Weighting (`FedAdp`) algorithm that aims to improve the FL performance by assigning distinct weight for participating devices to update the global model. We observe that devices with heterogeneous datasets make different contributions to global objective minimization. Therefore, our main intuition is to measure the devices' contribution based on the gradient information from local devices and then assign different weights accordingly and adaptively at each communication round for global model aggregation. According to device contribution, the proposed adaptive weighting strategy is capable of reducing the expected training loss of FL in each communication round and accelerating the model convergence. Different from existing works in [24, 87], our method does not impose additional communication and computation burden on local devices. Besides, the proposed adaptive weighting calculation is done in each communication round, which is distinct from existing methods [40, 42].

Our main contributions in this work are as follows.

- We identify that the presence of devices with non-i.i.d. data distributions slows down the convergence speed of FL. In addition, we analyze the convergence bound of gradient-

descent-based FL from a theoretical perspective and derive the convergence bound that incorporates the non-i.i.d. data distribution across participating devices and weighting strategy for model aggregation.

- We observe the implicit connection between data distribution on a device and the contribution from that device to the global model aggregation, measured at the central server side by inferring gradient information of participating devices. The convergence bound is lowered, and the convergence speed is accelerated by a carefully designed weighting strategy, `FedAdp`, that assigns different weights to devices for global model aggregation in each round of communication.

- We empirically evaluate the performance of the proposed weighting algorithm via extensive experiments using different real datasets with different learning objectives (i.e., convex and non-convex loss function). Our experimental results have shown that FL training with `FedAdp` can drastically reduce the communication rounds compared with the commonly adopted `FedAvg` algorithm.

Section 3.2 analyzes what factors impact the FL loss minimization in the SGD-based federated optimization method and initiates the contributed-based weighting design. The concrete design of the adaptive weighting algorithm is presented in Section 3.3, where the contribution is measured by gradient similarity and quantified with a non-linear mapping function. Comprehensive experimental results and related complete proof are shown in Section 3.4 and Section 3.5. Finally, Section 3.6 summarizes this chapter.

3.2 Convergence Analysis

Before diving into the algorithm design, we first analyze the convergence property of the SGD-based `FedAvg` algorithm. The theoretical analysis on the expected decrease of FL

loss in each global round reveals that gradient similarity and weighting design impact the convergence, which motivates us to weigh devices' updates proportional to that similarity in global model aggregation.

For theoretical analysis, we employ Assumption 2 and the following Assumption 4.

Assumption 4. ***Bounded Local Dissimilarity***

For any participating device k, the dissimilarity between local objective and global objective at $\mathbf{w}$ is bounded by A and B, i.e., $\mathsf{A}\|\nabla F(\mathbf{w})\| \leq \|\nabla F_k(\mathbf{w})\| \leq \mathsf{B}\|\nabla F(\mathbf{w})\|$.

A similar Assumption has been made in the FL context, for example, in [19, 21, 24]. In [19, 24], the dissimilarity across local gradients is imposed by an upper bound to capture the impact of data heterogeneity on FL convergence, and an analogous definition named gradient divergence is also presented in [21]. By tracking the divergence of gradients on each participating device, we observe that the dissimilarity can be further bounded by a lower bound, as shown in Assumption 4. Here $\nabla F(\mathbf{w})$ is the gradient of the global objective that is defined as $\nabla F(\mathbf{w}) = \sum_{k=1}^{|\mathcal{S}|} a_k \nabla F_k(\mathbf{w})$ in FL context. The local dissimilarity in Assumption 4 can be seen as a metric that reveals the data heterogeneity when the same training configuration (e.g., learning rate, batch size, training epoch, etc.) across participating devices is held. As a sanity check, when all the local data samples are the same, we have $\mathsf{A} = \mathsf{B} = 1$.

Theorem 1. *With loss function $F_k(\mathbf{w})$ satisfying Assumptions 2, 4 and supposing $\mathbf{w}^t$ is not a stationary solution, the expected decrease in the global loss function between two consecutive rounds satisfies,*

$$F(\mathbf{w}^{t+1}) \leq F(\mathbf{w}^t) - \eta \mathbb{E}_{k,t}\left[\left(\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t)\rangle}{\|\nabla F(\mathbf{w}^t)\|\|\nabla F_k(\mathbf{w}^t)\|} - \frac{\mathsf{B}L\eta}{2}\right) \cdot \frac{\mathsf{A}^2}{\mathsf{B}}\|\nabla F(\mathbf{w}^t)\|^2\right], \tag{3.1}$$

where the expectation $\mathbb{E}_{k,t}$ refers to the t-th global round weighting strategy of the participating

device $k \in \mathcal{S}$ for global model aggregation.

The proof of Theorem 1 is given in Section 3.5.1. Theorem 1 provides a bound on how rapid the decrease of the global FL loss can be expected. Based on Theorem 1, we have the following corollary and remarks.

Corollary 1. The convergence upper bound of FL after T global rounds is given by,

$$F(\mathbf{w}^T) \leq F(\mathbf{w}^1) - \eta \sum_{t=1}^{T} \mathbb{E}_{k,t} \left[\left(\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle}{\|\nabla F(\mathbf{w}^t)\| \|\nabla F_k(\mathbf{w}^t)\|} - \frac{\mathsf{B} L \eta}{2} \right) \cdot \frac{\mathsf{A}^2}{\mathsf{B}} \|\nabla F(\mathbf{w}^t)\|^2 \right]. \tag{3.2}$$

Remark 1. The decrease of FL loss between two consecutive global rounds shows a dependency on learning rate η, the bounded local dissimilarity of participating devices, the correlation between the local gradient and the global gradient $\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle}{\|\nabla F(\mathbf{w}^t)\| \|\nabla F_k(\mathbf{w}^t)\|}$, and the weight strategy $\mathbb{E}_{k,t}$ that weighs participating devices for the global model aggregation in each global round.

Remark 2. The local gradient, which is correlated with minimizing the local objective, may not align with the direction of approaching the optimal of the global objective. The correlation $\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle}{\|\nabla F(\mathbf{w}^t)\| \|\nabla F_k(\mathbf{w}^t)\|}$ between the local gradient and the global gradient is a metric to measure their alignment level. From Theorem 1, we can see this metric also indicates how much each device contributes to reducing FL loss in each round.

Remark 3. The FL loss $F(\mathbf{w}^{t+1})$ is negatively associated with the bound gap in Assumption 4, meaning that as bound gap $[\mathsf{A}, \mathsf{B}]$ grows larger, the bound weakens, and the convergence exacerbates. Intuitively, the root cause of dissimilarity is the divergence of local gradients across participating devices with heterogeneous datasets, which can be intentionally regularized by a properly designed weighting strategy.

An immediate suggestion from Theorem 1 that to improve the convergence of FL, one can reduce the FL loss by increasing $\mathbb{E}_{k,t}[\cdot]$ in each global round. This motivates us to

measure device contribution quantitatively through the correlation $\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle}{\|\nabla F(\mathbf{w}^t)\| \|\nabla F_k(\mathbf{w}^t)\|}$ between the local gradient and the global gradient and assign larger weights to the devices with higher contribution to enlarge the expected decrease of FL loss in each global round.

3.3 Adaptive Weighting

3.3.1 Measurement of device Contribution

In FL, the direction of minimizing local objective $F_k(\mathbf{w})$ might not align with the direction of minimizing $F(\mathbf{w})$. In particular, with a gradient aiming to minimize the local objective associated with data distribution q_k, the gradient on different devices may be tremendously diverse, especially for participating devices with heterogeneous datasets. As such, the contribution from participating devices for global aggregation is different. Empirically, we note that if the data distribution on a device is highly skewed, the gradient may highly deviate from or even in the opposite direction to the global gradient, causing a negative effect on the global aggregation.

Instead of assigning weight for participating devices based on the size of datasets as in `FedAvg` [5], we measure the contribution of participating devices based on the correlation between the local gradient and global gradient. Particularly, we quantify the contribution of each device at each global round based on *angle* θ_k^t, which is defined as

$$\theta_k^t = \arccos \frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle}{\|\nabla F(\mathbf{w}^t)\| \|\nabla F_k(\mathbf{w}^t)\|}. \tag{3.3}$$

where the function "arccos" represents the inverse cosine function of the cosine similarity between local gradient $\nabla F_k(\mathbf{w}^t)$ and the global gradient $\nabla F(\mathbf{w}^t)$ in each round t. From (3.3), we can see that when the angle θ_k^t is small, it means the local gradient $\nabla F_k(\mathbf{w}^t)$ has a similar direction to the global gradient, thereby positively contributing to the global aggregation. In

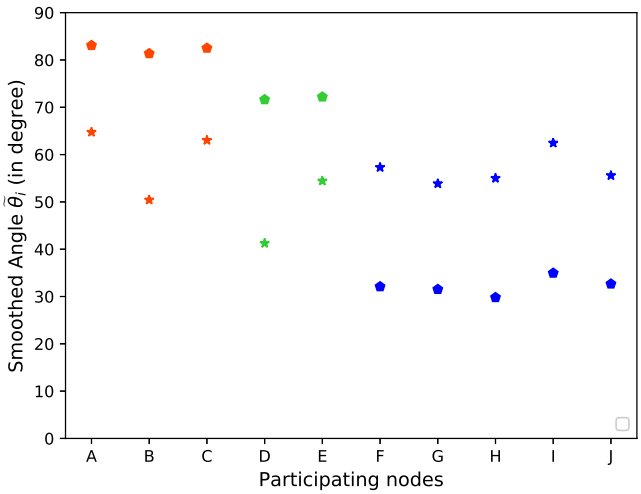

Figure 3.1: The smoothed angle $\tilde{\theta}_k$ of participating device at different training rounds, where star and pentagon sign denote the angle at communication round 1 and communication round 15, respectively. Devices with different data distributions are marked with different colors.

contrast, when θ_k^t is large, e.g., larger than $\pi/2$, the local gradient $\nabla F_k(\mathbf{w}^t)$ has an opposite direction to the global gradient, thereby negatively contributing to the global aggregation.

To restrain the instability caused by randomness presented in instantaneous angle θ_k^t at each round, we use so-called *smoothed angle* $\tilde{\theta}_k^t$ as a substitution, which is the averaged angle over previous training rounds and is defined as

$$\tilde{\theta}_k^t = \begin{cases} \theta_k^t & t = 1 \\ \frac{t-1}{t}\tilde{\theta}_k^{t-1} + \frac{1}{t}\theta_k^t & t > 1 \end{cases}. \tag{3.4}$$

By using *smoothed angle* $\tilde{\theta}_k^t$, the angle difference across devices uniquely depends on the data distribution. Intuitively, the angle $\tilde{\theta}_k^t$ will be larger as the dissimilarity between data distribution on device k and population distribution grows. Also, the smoothed angle is capable of quantifying the degree of data dissimilarity among the local devices.

We conduct an experiment to illustrate how data distribution can be reflected by angle.

Under the same training model in 3.3, we randomly assign i) 3 devices with 1-*class non-i.i.d. setting* (i.e., device "A", "B", "C"), ii) 2 devices with 2-*class non-i.i.d. setting* (i.e., devices "D" and "E"), and iii) the rest of 5 devices with *i.i.d. setting*.

As shown in Fig. 3.1, the smoothed angle between the local gradient and the global gradient is full of randomness at the beginning of FL training (labeled with the star sign). Along with the training, smoothed angle $\tilde{\theta}_k$ indicated by the pentagon sign shows diversity across the participating devices due to the impact of data heterogeneity on local training. To be more specific, for those devices with 1-class non-i.i.d. setting, the data samples are highly skewed since the label space is extremely limited. Due to the limited richness of data samples on device k, the direction for minimizing its local objective $F_k(\mathbf{w})$, which is reflected by $\nabla F_k(\mathbf{w})$, will be far away from the direction for minimizing the overall objective $F(\mathbf{w})$, which is reflected by $\nabla F(\mathbf{w}) = \sum_{k=1}^{|\mathcal{S}|} a_k \nabla F_k(\mathbf{w})$, resulting a greater θ_k as defined by (3.3). As shown in Fig. 3.1, the gradient from the device with extremely skewed data (e.g., device "A", "B", "C") is nearly orthogonal with the global gradient after 15 communication rounds, which barely brings a contribution to the global model. If we ignore the discrepancy of device contribution and average local update according to the size of datasets, as in `FedAvg`, it slows model convergence.

3.3.2 `Federated Adaptive Weighting` (`FedAdp`)

Provided the diverse device contribution from participating devices, the weighting strategy affects Theorem 1 through the expectation $\mathbb{E}_{k,t}\left[\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t)\rangle}{\|\nabla F(\mathbf{w}^t)\|\|\nabla F_k(\mathbf{w}^t)\|}\right]$ consequently. To accelerate the convergence rate, we seek to lower the upper bound of the expected loss in each communication round, which reveals to assign different weights $\tilde{a}_k$ to different devices for the global model aggregation. As such, the corresponding objective is formally stated as enlarging $\mathbb{E}_{k,t}\left[\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t)\rangle}{\|\nabla F(\mathbf{w}^t)\|\|\nabla F_k(\mathbf{w}^t)\|}\right] = \sum_{k=1}^{|\mathcal{S}|} \frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t)\rangle}{\|\nabla F(\mathbf{w}^t)\|\|\nabla F_k(\mathbf{w}^t)\|} \cdot \tilde{a}_k^t$ via designing $\tilde{a}_k$ under the

inherent constrain $\sum_{k=1}^{|\mathcal{S}|} \tilde{a}_k^t = 1, \quad \tilde{a}_k^t \geq 0 \ \ \forall k, t$.

Considering the device contribution is measured by (3.3), a natural weighting design aiming to enlarge the expectation should follow the criterion that devices with higher contribution deserve higher weights for aggregation in each global round. We characterize the contribution-regulated weighting strategy for the global aggregation in each global round adaptively as `Fed`erated `Ad`aptive Weighting (`FedAdp`).

Assigning adaptive weight for updating the global model in the proposed `FedAdp` algorithm includes two steps.

Non-linear mapping function

We design a non-linear mapping function to first quantify the contribution of each device based on angle information. Inspired by the sigmoid function, we use a variant of *Gompertz function*[88], which is a non-linear decreasing function defined as

$$\tilde{f}(\tilde{\theta}_k^t) = \varsigma(1 - e^{-e^{-\varsigma(\tilde{\theta}_k^t - 1)}}), \tag{3.5}$$

where $\tilde{\theta}_k^t$ is the smoothed angle in *radian*, e denotes the exponential constant and ς is a constant as explained in the following.

The designed mapping function has several properties that are important for the subsequent weight calculation:

- $\lim_{\tilde{\theta}_k^t \to \pi/2} \tilde{f}(\tilde{\theta}_k^t) = \iota$, where $\iota \propto \frac{1}{\varsigma}$ is constant;
- $\lim_{0 \to \tilde{\theta}_k^t \to \nu} \tilde{f}(\tilde{\theta}_k^t) = \varsigma$, where $\nu \propto \varsigma$ is a constant;

ς controls the decreasing rate of $\tilde{f}(\tilde{\theta}_k^t)$ from ς to ι as $\tilde{\theta}_k^t$ increases from ν to $\pi/2$. For example, a small $\varsigma \in \mathbb{Z}^+$ indicates a lower decreasing rate of $\tilde{f}(\tilde{\theta}_k^t)$ that decreases from ς to $\iota \propto \frac{1}{\varsigma}$ as $\tilde{\theta}_k^t$ increases from $\nu \propto \varsigma$ to $\pi/2$. As ς increases, the gap between small angle and

large angle is amplified (e.g., $\tilde{f}(\tilde{\theta}_k^t)$ changes within a relatively large range $[\varsigma, \iota]$ as $\tilde{\theta}_k^t$ increases within range $[\varsigma, \pi/2]$), so is the difference of contribution from those devices. However, keeping increasing ς is not consistently effective to distinguish the difference of contributions from devices. Since ν is proportional to ς, a large ς narrows the boundary $[\nu, \frac{\pi}{2}]$ where the device contribution should be considered, making the contribution of devices whose angle lays between $[0, \nu]$ indistinguishable.

Weighting

After getting the contribution mapped using the smoothed angle from each device, we use *Softmax function* to finally calculate the weight of participating devices for global model aggregation as follows,

$$\tilde{a}_k^t = \begin{cases} \frac{e^{\tilde{f}(\tilde{\theta}_k^t)}}{\sum_{k'=1}^{|\mathcal{S}|} e^{\tilde{f}(\tilde{\theta}_{k'}^t)}} & |\mathcal{D}_m| = |\mathcal{D}_n|, \forall m, n \in \mathcal{S} \\ \frac{|\mathcal{D}_k| e^{\tilde{f}(\tilde{\theta}_k^t)}}{\sum_{k'=1}^{|\mathcal{S}|} |\mathcal{D}_{k'}| e^{\tilde{f}(\tilde{\theta}_{k'}^t)}} & |\mathcal{D}_m| \neq |\mathcal{D}_n|, \exists m, n \in \mathcal{S} \end{cases}. \tag{3.6}$$

From the first line of (3.6), if all the participating devices have the same size of data samples, the proposed `FedAdp` algorithm will assign weight solely based on their contribution quantified by $e^{\tilde{f}(\tilde{\theta}_k^t)}$. From the 2nd line of (3.6), `FedAdp` will assign weight based on both the contribution and the data size.

Remark 4. Different from `FedAvg`, where the weight for aggregation is solely proportional to the size of local datasets (e.g., $a_k = |\mathcal{D}_k| / \sum_{k'=1}^{|\mathcal{S}|} |\mathcal{D}_k|$), `FedAdp` takes both the data size and the node contribution into consideration when assigning weights for model aggregation.

The reason for adopting the Softmax function is twofold: i) The output of the Softmax function is a normalized value with a larger angle corresponding to a smaller weight. ii) Using

Algorithm 1 `Fed`erated `Ada`ptive Weighting (`FedAdp`)

procedure FEDERATED OPTIMIZATION

Input: device set $\mathcal{S}, E, B, T, \eta$,

Server initializes global model $\mathbf{w}^1$, global update Δ^1, smoothed angle $\tilde{\theta}_k^1, k \in \mathcal{S}$

  for $t = 1, \cdots, T-1$ **do**

    for device $k \in \mathcal{S}$ in parallel **do**

      $\Delta_k^{t+1} \leftarrow$ LOCAL UPDATE $(k, \mathbf{w}_k^t)$

    $\mathbf{w}^{t+1} \leftarrow$ GLOBAL UPDATE $(\Delta_1^{t+1}, \Delta_2^{t+1}, \cdots, \Delta_{|\mathcal{S}|}^{t+1})$

procedure LOCAL UPDATE

Input: device index k, model $\mathbf{w}_k^t$

Calculate local updates for τ times of SGD with step-size η on $F_k(\mathbf{w})$ and obtain $\mathbf{w}_k^{t+1}$

Calculate the model difference $\Delta_k^{t+1} = \mathbf{w}_k^{t+1} - \mathbf{w}^t$

return Δ_k^{t+1}

procedure GLOBAL UPDATE

Input: local update $\Delta_1^{t+1}, \Delta_2^{t+1}, \cdots, \Delta_{|\mathcal{S}|}^{t+1}$

Calculate the global gradient $\nabla F(\mathbf{w}^{t+1}) = \sum_{k=1}^{|\mathcal{S}|}(|\mathcal{D}_k| / \sum_{k'=1}^{|\mathcal{S}|} |\mathcal{D}_{k'}|)\nabla F_k(\mathbf{w}^{t+1})$, where $\nabla F_k(\mathbf{w}^{t+1}) = -\Delta_k^{t+1}/\eta$

Calculate instantaneous angle θ_k^t by (3.3)

Update smoothed angle $\tilde{\theta}_k^t$ by (3.4)

Calculate weight for model aggregation by (3.5), (3.6)

Update global model $\mathbb{E}_{k,t}\left[\tilde{a}_k^t \mathbf{w}_k^t\right]$

return $\mathbf{w}^t$

the Softmax function, each device's contribution can be reinforced or suppressed, depending on the smoothed angle between its gradient and the global gradient.

The complete procedures of the proposed `FedAdp` algorithm are presented in Algorithm 1 and `FedAdp` with adaptive weighting strategy leads to the following theorem.

Theorem 2. *`FedAdp` with weight design $\tilde{a}_k$ achieves a tighter bound on FL loss decrease in Theorem 1 than `FedAvg` with weight a_k.*

The proof of Theorem 2 is provided in Section 3.5.2.

Compared to `FedAvg`, `FedAdp` adopts a simple yet effective strategy that measures the device contribution by quantifying the correlation between the local gradient and the global gradient. Weight for the global model updates can be adaptively assigned based on device

contribution rather than evenly averaging, which results in greater FL loss reduction in each global round and consequently accelerates model convergence, as confirmed by our experimental results.

3.4 Numerical Results

To evaluate the performance of our proposed adaptive weighting algorithm, we implemented `FedAdp` with PyTorch framework and PySyft library and studied the image classification task. We evaluated `FedAdp` by training typical convex and non-convex learning models on two datasets, MNIST and FashionMNIST, and when the different degree of skewness of the non-i.i.d. dataset is presented, we investigated how `FedAdp` outperforms `FedAvg`[5] by assigning adaptive weight for model aggregation.

We consider Multinomial Logistic Regression[4] (MLR) model and CNN model[5] to represent convex and non-convex learning objectives, respectively. We use the number of communication rounds for the FL model to reach a target testing accuracy as a performance metric. Unless otherwise specified, the target accuracy is set to 95% for training on MNIST and 80% for training on FashionMNIST[6]. The number of participating devices $|\mathcal{S}| = 10$, $|\mathcal{D}_k| = 600$, SGD batch size 50 for MLR and 32 for CNN, $E = 1$, $T = 300$, $\eta = 0.01$, decay rate $= 0.995$, the constant in non-linear mapping function $\varsigma = 5$. The skewness of the dataset is measured by *x-class non-i.i.d.*. The dataset for devices is generated in the same way as in Chapter 2.1.3. Particularly, X i.i.d. + Y non-i.i.d. (1) (or (2)) represents X devices are at *i.i.d. setting* and

[4]For the MLR model, the input is a flattened 784-dimension (28×28) image, and the output is a class label between 0 and 9. Note that the MLR model can be extended to the strongly convex setting by adding regularization term [89].

[5]The CNN has 7 layers with the following structure: $5 \times 5 \times 32$ Convolutional $\rightarrow 2 \times 2$ MaxPool $\rightarrow 5 \times 5 \times 64$ Convolutional $\rightarrow 2 \times 2$ MaxPool $\rightarrow 1024 \times 512$ Fully connected $\rightarrow 512 \times 10$ Fully connected $\rightarrow$ Softmax (1,663,370 total parameters). All Convolutional and Fully connected layers are mapped by ReLu activation. The configuration is similar to [5].

[6]FashionMNIST dataset contains 70,000 28×28 grayscale images of fashion products from 10 categories from a dataset of Zalando article images, with 7,000 images per class.

Y devices are at *1-class (or 2-class) non-i.i.d. setting.*

3.4.1 Data Heterogeneity

We investigate the different number of non-i.i.d. devices with different skewness levels of non-i.i.d. data to testify the efficiency of `FedAdp`. For non-i.i.d. data, two skewness cases that $x = 1, 2$ are considered. We plot the test accuracy vs. the communication rounds of federated learning in Fig. 3.2 and Fig. 3.3 when MLR and CNN models are adopted, respectively.

MLR Model

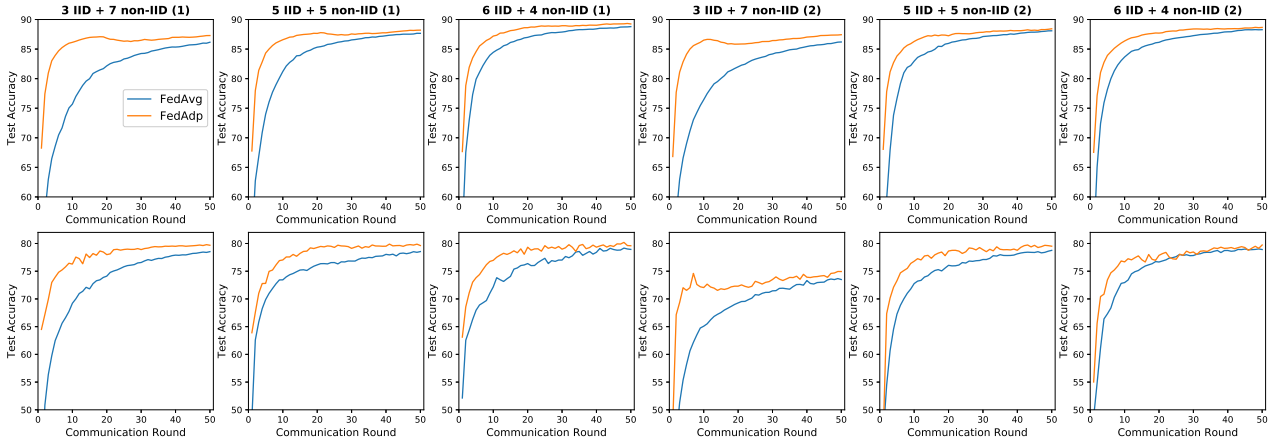

Figure 3.2: Test accuracy over communication rounds of `FedAdp` and `FedAvg` with heterogeneous data distribution over participating devices using MLR model. Upper and lower subplots correspond to training performance on MNIST and FashionMNIST datasets.

Given that the learning capability of MLR is limited, instead of setting a target accuracy, we simply train a model over 50 global rounds. We plot the test accuracy vs. the communication rounds of federated learning algorithms in Fig. 3.2. From Fig. 3.2, we can tell `FedAdp` always outperforms `FedAvg` when the devices with non-i.i.d. dataset are present. In addition, `FedAdp` converges very fast in the early training stage, and the superiority of `FedAdp` is more

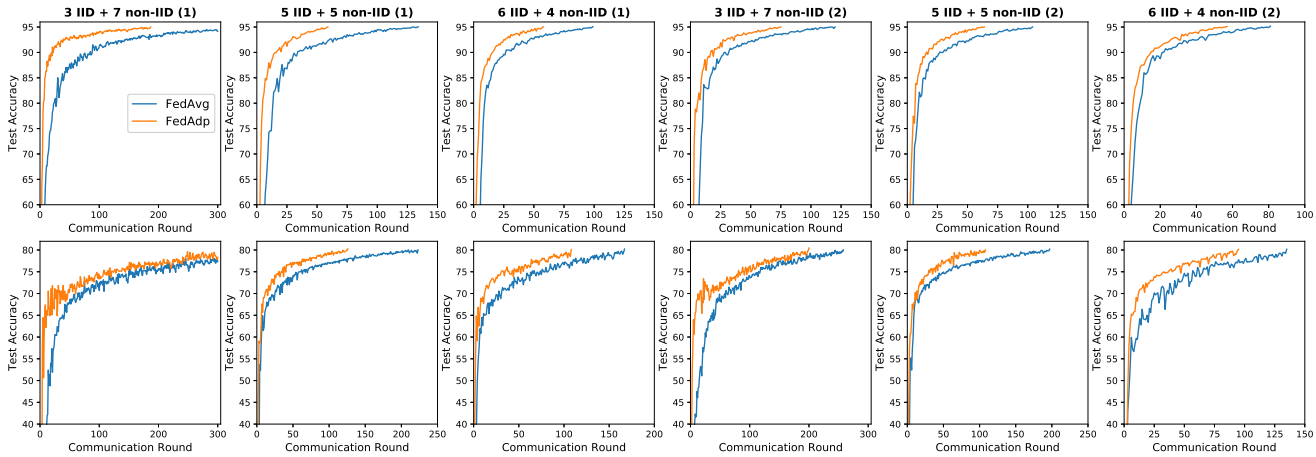

Figure 3.3: Test accuracy over communication rounds of `FedAdp` and `FedAvg` with heterogeneous data distribution over participating devices using CNN model. Upper and lower subplots correspond to training performance on MNIST and FashionMNIST datasets, respectively.

prominent when the proportion of devices with non-i.i.d. datasets is larger. It is noted that the gap between `FedAdp` and `FedAvg` over 50 global rounds is not conspicuous because of the simplicity of the MLR model. Different weighting strategies will not make much difference when the model is reaching its learning capability. In contrast, the weighting strategy will consistently impact the FL training process when a more complex neural network model is applied, as shown in the following experiment.

CNN Model

We plot the test accuracy vs. the communication rounds of federated learning in Fig. 3.3. From Fig. 3.3, we can tell `FedAdp` always outperforms `FedAvg` when the devices with non-i.i.d. dataset are present. In particular, `FedAdp` converges very fast in the early training stage since the gradient divergence is more obvious in the initial rounds, which makes the effect of assigning adaptive weight for updating the global model even more significant.

To measure the effectiveness of `FedAdp`, we count the number of communication rounds

Table 3.1: Number of communication rounds to reach a target accuracy for `FedAdp`, versus `FedAvg`, within 300 rounds. N/A refers that algorithms can not reach target accuracy before termination where the highest test accuracy is shown

	3 i.i.d. + 7 non-i.i.d.	5 i.i.d. + 5 non-i.i.d.	6 i.i.d. + 4 non-i.i.d.
MNIST 95% Accuracy			
1-class non-i.i.d.			
`FedAvg`	N/A (94.48%)	133	99
`FedAdp`	**187**	**61**	**58**
2-class non-i.i.d.			
`FedAvg`	120	104	81
`FedAdp`	**75**	**59**	**52**
Fashion MNIST 80% Accuracy			
1-class non-i.i.d.			
`FedAvg`	N/A (77.31%)	222	167
`FedAdp`	N/A (**79.5**%)	**125**	**107**
2-class non-i.i.d.			
`FedAvg`	258	196	134
`FedAdp`	**207**	**107**	**94**

needed to reach a target accuracy when `FedAdp` is adopted. Each entry in Table 3.1 shows the number of communication rounds necessary to achieve a test accuracy of 95% for CNN on MNIST and 80% for FashionMNIST. The bold number indicates the better result achieved by `FedAdp`, as compared to `FedAvg`. `FedAdp` decreases the number of communication rounds by up to 54.1% and 43.2% for the MNIST task when non-i.i.d. devices are at 1-class and 2-class non-i.i.d. setting, respectively. For the FashionMNIST task, the corresponding decreases are up to 43.7% and 45.4%, respectively. In the cases when the target accuracy is not reachable before 300 rounds, `FedAdp` always terminates with higher testing accuracy.

Previously, two extremely skewness cases that $x = 1, 2$ are considered, while the superiority of the proposed weighting strategy is not limited to extreme cases. To verify the proposed weighting strategy in a more general data heterogeneity case, we consider the CNN model for the MNIST dataset in the following two cases.

- Case 1: The number of classes of data samples owned by device k, denoted by x_k, is randomly selected from the set $\{1, 2, \cdots, 10\}$ without overlapping. Whereafter, the

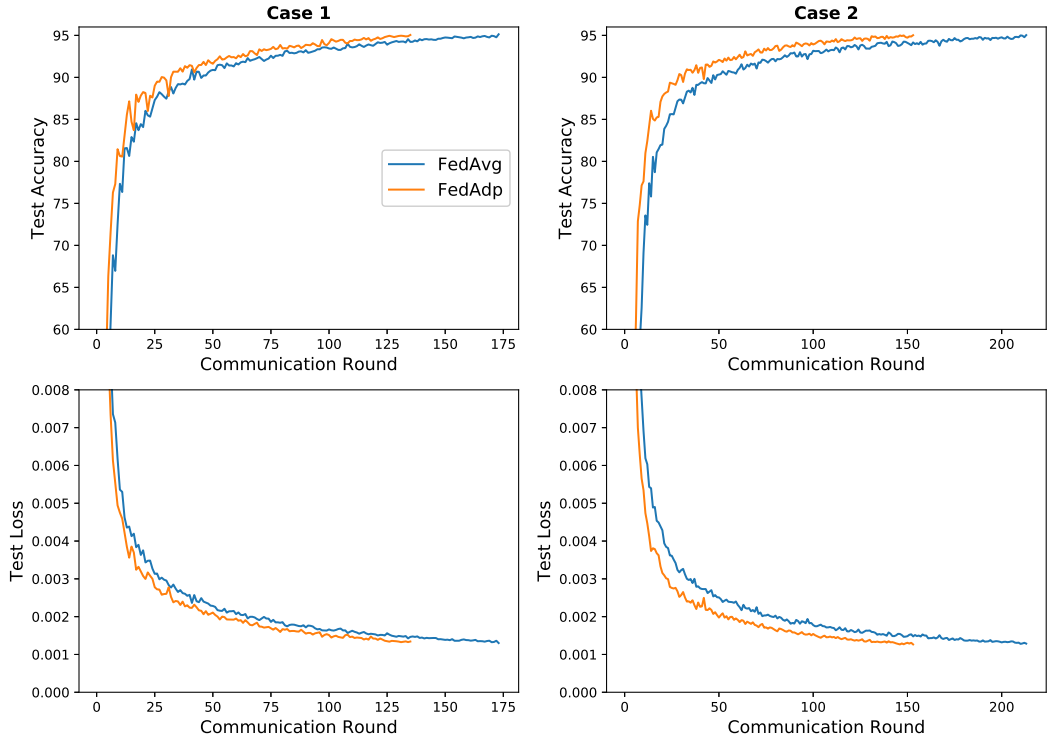

Figure 3.4: FL training performance over communication rounds when `FedAdp` is adopted considering general heterogeneous data distribution over participating nodes. The top row and bottom row represent the test accuracy and training loss over the communication round, respectively.

data samples on each device are randomly selected from the x_k-subset of the training dataset.

- Case 2: For half of the devices, their x_k (i.e., the number of classes of data samples) is selected following the uniform distribution $\mathcal{U}(1, 5)$, whereas for the other half, x_k follows the uniform distribution $\mathcal{U}(6, 10)$. The data samples on each device are randomly selected from the x_k-subset of the training dataset.

From Fig. 3.4, we can see `FedAdp` outperforms `FedAvg` in both cases. In both cases, the convergence performance is worse than the result in Fig. 3.3 because the number of i.i.d.

devices is small and the local dissimilarity is greater in these two cases. However, it is clear by measuring device contribution, `FedAdp` is more rapid in reducing FL loss in each global round thus accelerating model convergence, even without the participation of i.i.d. devices.

3.4.2 Choosing ς

One natural question is how to determine ς for non-linear function. A large ς may increase the convergence by emphasizing the difference of contribution from participating devices, which hastens model convergence in the initial training stage. Meanwhile, since ν is proportional to ς, a large ς also narrows the boundary $[\nu, \frac{\pi}{2}]$ where the device contribution should be considered, making the contribution of devices whose angle lays between $[0, \nu]$ indistinguishable.

We heuristically choose $\varsigma \in \mathbb{Z}^+$ in the ascending order. From Fig. 3.5, increasing ς leads to faster convergence since the gap between small angle and large angle is amplified, so is the difference of contribution from those devices. However, a larger ς is not always effective, especially after the initial training stage. Empirically, the best ς is 5 for our experimental setting.

3.4.3 Divergence Measurement

Finally, in Fig. 3.6, we take one experimental case as an example to demonstrate the divergence of local gradients, which captures the overall data heterogeneity of participating devices. In particular, we track the divergence of gradients over all participating devices, which is measured by $\sum_{k\in\mathcal{S}} \frac{1}{|\mathcal{S}|}\|\nabla F(\mathbf{w}) - \nabla F_k(\mathbf{w})\|$. Empirically, we observe that our proposed weighting strategy leads to smaller divergence among participating devices, and the smaller the divergence, the smaller the FL loss. As $\mathbf{w}^t$ is not a stationary solution along with the training, aggregation by `FedAdp` is seen as a regularization process that restrains the local weight $\mathbf{w}_k^{t+1}$ trained by skewed datasets from being deviatory, which lowers the model

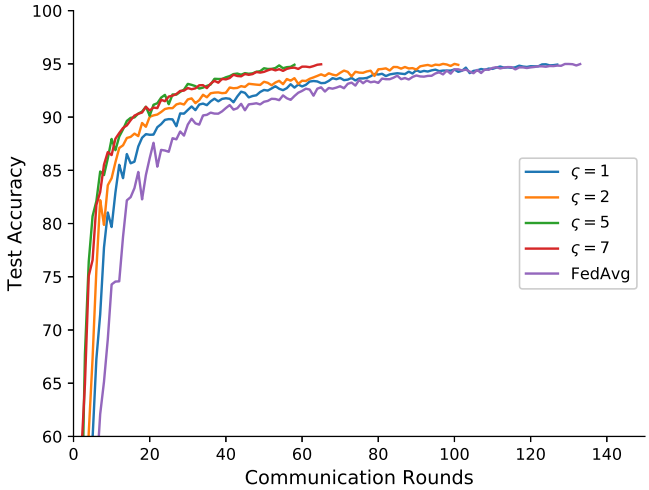

Figure 3.5: Effect of setting ς on federated learning performance. Data heterogeneity setting is 5 i.i.d. + 5 non-i.i.d. (1) and CNN model is adopted.

divergence and consequently accelerates the convergence.

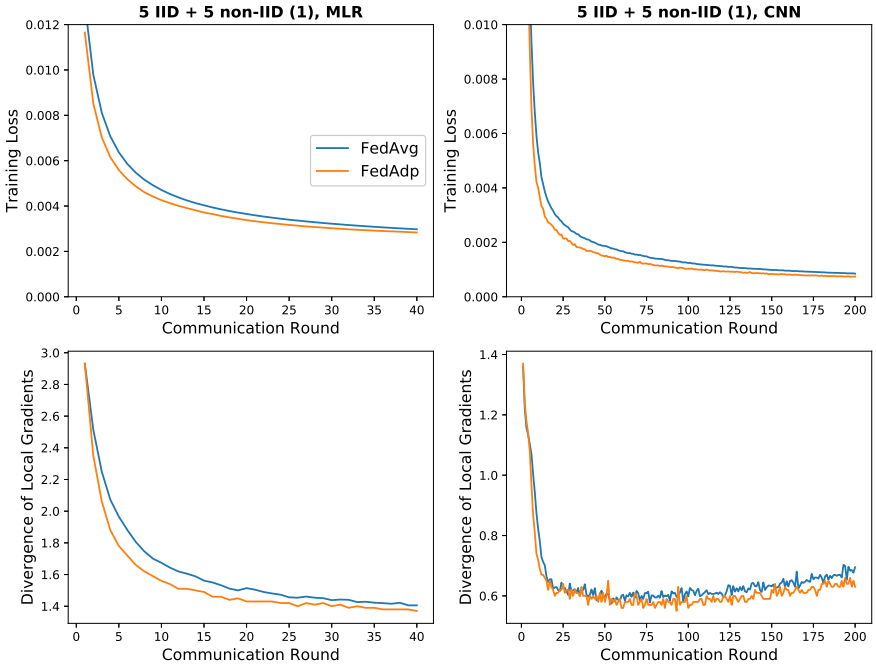

Figure 3.6: The connection between the model test loss and the divergence across local gradients. The proposed weighting strategy `FedAdp` gives an impact on alleviating the divergence brought by devices with skewed datasets. (1) Top row: the training loss on the MNIST dataset under one data heterogeneity setting (5 i.i.d. + 5 non-i.i.d. (1)). (2) Bottom row: the corresponding divergence measurement.

3.5 Complete Proof

3.5.1 Proof of Theorem 1

From the L-smoothness of $F(\mathbf{w})$, we have

$$F(\mathbf{w}^{t+1}) \leq F(\mathbf{w}^t) + \langle \nabla F(\mathbf{w}^t), \mathbf{w}^{t+1} - \mathbf{w}^t \rangle + \frac{L}{2} \|\mathbf{w}^{t+1} - \mathbf{w}^t\|^2. \tag{3.7}$$

The last two terms on the right-hand side of the above inequality are bounded as

- *Bounding* $\|\mathbf{w}^{t+1} - \mathbf{w}^t\|^2$

By the definition of the global aggregation for $\mathbf{w}^{t+1}$, we have

$$\|\mathbf{w}^{t+1} - \mathbf{w}^t\| = \mathbb{E}_{k,t}\left[\|\mathbf{w}_k^{t+1} - \mathbf{w}^t\|\right]. \tag{3.8}$$

By following SGD optimization, for each term within the expectation in the right-hand side of (3.8), we have

$$\mathbf{w}_k^{t+1} = \mathbf{w}^t - \eta \nabla F_k(\mathbf{w}^t). \tag{3.9}$$

$$\|\mathbf{w}^{t+1} - \mathbf{w}^t\|^2 = (\mathbb{E}_{k,t}\left[\|\mathbf{w}_i^{t+1} - \mathbf{w}^t\|\right])^2 = \eta^2(\mathbb{E}_{k,t}\left[\|\nabla F_k(\mathbf{w}^t)\|\right])^2 \overset{1}{\leq} \eta^2 \mathbb{E}_{k,t}\left[\|\nabla F_k(\mathbf{w}^t)\|^2\right], \tag{3.10}$$

where inequality 1 holds by Cauchy-Schwarz inequality.

- *Bounding* $\langle \nabla F(\mathbf{w}^t), \mathbf{w}^{t+1} - \mathbf{w}^t \rangle$

Again, by the definition of the global aggregation for $\mathbf{w}^{t+1}$ and (3.9) we have

$$\langle \nabla F(\mathbf{w}^t), \mathbf{w}^{t+1} - \mathbf{w}^t \rangle = -\eta \mathbb{E}_{k,t}\left[\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle\right]. \tag{3.11}$$

The expectation term in (3.11) can be further rewritten as

$$\begin{aligned}\mathbb{E}_{k,t}\left[\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle\right] &= \mathbb{E}_{k,t}\left[\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle}{\|\nabla F(\mathbf{w}^t)\|\|\nabla F_k(\mathbf{w}^t)\|} \cdot \|\nabla F(\mathbf{w}^t)\|\|\nabla F_k(\mathbf{w}^t)\|\right] \\ &\overset{2}{\geq} \mathbb{E}_{k,t}\left[\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle}{\|\nabla F(\mathbf{w}^t)\|\|\nabla F_k(\mathbf{w}^t)\|} \cdot \frac{\|F_k(\mathbf{w}^t)\|^2}{\mathsf{B}}\right],\end{aligned} \tag{3.12}$$

where inequality 2 comes from Assumptions 4 that local dissimilarity is upper bounded by B.

Plugging (3.12) into (3.11), then the last two terms on the right-hand side of (3.7) are

expressed as

$$
\begin{aligned}
&\langle \nabla F(\mathbf{w}^t), \mathbf{w}^{t+1} - \mathbf{w}^t \rangle + \frac{L}{2}\|\mathbf{w}^{t+1} - \mathbf{w}^t\|^2 \\
\leq &- \eta \mathbb{E}_{k,t}\left[\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t)\rangle}{\|\nabla F(\mathbf{w}^t)\|\|\nabla F_k(\mathbf{w}^t)\|} \cdot \frac{\|F_k(\mathbf{w}^t)\|^2}{B}\right] + \frac{L\eta^2}{2}\mathbb{E}_{k,t}\left[\|\nabla F_k(\mathbf{w}^t)\|^2\right] \\
\overset{3}{\leq} &- \eta \mathbb{E}_{k,t}\left[\left(\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t)\rangle}{\|\nabla F(\mathbf{w}^t)\|\|\nabla F_k(\mathbf{w}^t)\|} - \frac{\mathsf{B}L\eta}{2}\right) \cdot \frac{\mathsf{A}^2}{\mathsf{B}}\|\nabla F(\mathbf{w}^t)\|^2\right],
\end{aligned} \tag{3.13}
$$

where inequality 3 holds because of Assumptions 4 that local dissimilarity is lower bounded by A.

Finally, Theorem 1 is proved by substituting (3.13) into (3.7).

3.5.2 Proof of Theorem 2

We consider the general case that participating devices have a different number of data samples. For device k with data size $|\mathcal{D}_k|$, we create $|\mathcal{D}_k|$ virtual devices, each with a unit sample size. Hereinafter, we use index $(k, j), j \in \{1, \cdots, |\mathcal{D}_k|\}$ to denote the j-th virtual device split from the participating device $k, k \in \mathcal{S}$, where the gradient information is kept on virtual devices as on the participating device (e.g., $\nabla F_{k,j}(\mathbf{w}^t) = \nabla F_k(\mathbf{w}^t), \theta_{k,j} = \theta_k$). As such, all virtual devices split by device k share the same weight (i.e., $\tilde{a}^t_{k,j} = \tilde{a}^t_{k,j\prime}, \forall j, j\prime \in \{1, \cdots, |\mathcal{D}_k|\}$), where $\tilde{a}^t_{k,j}$ denotes the weight for virtual device (k, j). The weight of device k is $\tilde{a}^t_k = \sum_{j=1}^{|\mathcal{D}_k|} \tilde{a}^t_{k,j} = |\mathcal{D}_k|\tilde{a}^t_{k,j}$.

From (3.3), $\theta_{k,j} = \theta_k$ monotonically decreases with $\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_i(\mathbf{w}^t)\rangle}{\|\nabla F(\mathbf{w}^t)\|\|\nabla F_i(\mathbf{w}^t)\|}$. From (3.5), $\tilde{f}(\cdot)$ is a decreasing function of θ. Thus, by that $\tilde{a}^t_{k,j} = \frac{e^{\tilde{f}(\tilde{\theta}^t_{k,j})}}{\sum_{i'=1}^{|\mathcal{S}|}|\mathcal{D}_{k\prime}|e^{\tilde{f}(\tilde{\theta}^t_{k\prime})}}$, we can see $\tilde{a}^t_{k,j}$ monotonically

increases with $\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t)\rangle}{\|\nabla F(\mathbf{w}^t)\|\|\nabla F_k(\mathbf{w}^t)\|}$. Therefore, generic $\tilde{a}_{k,j}^t$ satisfies the following criterion

$$\begin{aligned}
&\tilde{a}_{k,j}^t \propto \frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t)\rangle}{\|\nabla F(\mathbf{w}^t)\|\|\nabla F_k(\mathbf{w}^t)\|} \\
&\tilde{a}_{k,j}^t \geq 0 \ \ \forall k, j, t \\
&\sum_{k=1}^{|\mathcal{S}|}\sum_{j=1}^{|\mathcal{D}_k|} \tilde{a}_{k,j}^t = \sum_{k=1}^{|\mathcal{S}|} \tilde{a}_k^t = 1,
\end{aligned} \tag{3.14}$$

with the corresponding bound of the expected loss being

$$F(\mathbf{w}^{t+1}) \leq F(\mathbf{w}^t) - \eta \sum_{k=1}^{|\mathcal{S}|} \left(\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t)\rangle}{\|\nabla F(\mathbf{w}^t)\|\|\nabla F_k(\mathbf{w}^t)\|} \tilde{a}_k^t - \frac{BL\eta}{2} \right) \cdot \frac{A^2}{B} \|\nabla F(\mathbf{w})\|^2. \tag{3.15}$$

where $\tilde{a}_k^t$ is defined as in (3.6).

In order to compare the expected loss achieved by `FedAdp` and `FedAvg`, one can simply measure the expectation term in (3.1). We use $u_{k,j}$ to denote the contribution from virtual device j of participating device k for model aggregation. In each global round, we sort the contribution from all the virtual devices that is measured by the correlation $\frac{\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t)\rangle}{\|\nabla F(\mathbf{w}^t)\|\|\nabla F_k(\mathbf{w}^t)\|}$ between the local gradient and the global gradient in descending order, that is $u_{1,1} = u_{1,2} = \cdots = u_{1,|\mathcal{D}_1|} \geq u_{2,1} = u_{2,2} = \cdots = u_{2,|\mathcal{D}_2|} \geq \cdots \geq u_{|\mathcal{S}|,1} = u_{|\mathcal{S}|,2} = \cdots = u_{|\mathcal{S}|,|\mathcal{D}_{|\mathcal{S}|}|}$. Apparently, the weight assigned to virtual device in `FedAdp` should follow the same order $\tilde{a}_{1,1} = \tilde{a}_{1,2} = \cdots = \tilde{a}_{1,|\mathcal{D}_1|} \geq \tilde{a}_{2,1} = \tilde{a}_{2,2} = \cdots = \tilde{a}_{2,|\mathcal{D}_2|} \geq \cdots \geq \tilde{a}_{|\mathcal{S}|,1} = \tilde{a}_{|\mathcal{S}|,2} = \cdots = \tilde{a}_{|\mathcal{S}|,|\mathcal{D}_{|\mathcal{S}|}|}$, with $\sum_k \sum_j \tilde{a}_{k,j} = 1$. As such, by Chebyshev's inequality [90], we have the following hold for any $u_{m,j}$, $u_{n,j'}$

$$\begin{aligned}
&\bar{a}(u_{m,j} - u_{n,j'})(\frac{\tilde{a}_{m,j}}{\bar{a}_{m,j}} - \frac{\tilde{a}_{n,j'}}{\bar{a}_{n,j'}}) \geq 0 \\
&\bar{a}[u_{m,j}\tilde{a}_{m,j}\bar{a}_{n,j'} + u_{n,j'}\tilde{a}_{n,j'}\bar{a}_{m,j}] \geq \bar{a}[u_{m,j}\tilde{a}_{n,j'}\bar{a}_{m,j} + u_{n,j'}\tilde{a}_{m,j}\bar{a}_{n,j'}],
\end{aligned} \tag{3.16}$$

where $\bar{a} = \bar{a}_{m,j} = \bar{a}_{n,j'} = \frac{1}{|\mathcal{D}|}$ denotes the weight of `FedAvg` for all virtual devices with

$|\mathcal{D}| = \sum_k^{|\mathcal{S}|} |\mathcal{D}_k|$.

Adding all the $|\mathcal{D}|^2$ inequalities, we have

$$
\begin{aligned}
&\bar{a}\left[\sum_{m=1}^{|\mathcal{S}|}\sum_{j=1}^{|\mathcal{D}_m|}\sum_{n=1}^{|\mathcal{S}|}\sum_{j'=1}^{|\mathcal{D}_n|} u_{m,j}\tilde{a}_{m,j}\bar{a}_{n,j'} + u_{n,j'}\tilde{a}_{n,j'}\bar{a}_{m,j}\right] \\
\geq &\bar{a}\left[\sum_{m=1}^{|\mathcal{S}|}\sum_{j=1}^{|\mathcal{D}_m|}\sum_{n=1}^{|\mathcal{S}|}\sum_{j'=1}^{|\mathcal{D}_n|} u_{m,j}\tilde{a}_{n,j'}\bar{a}_{m,j} + u_{n,j'}\tilde{a}_{m,j}\bar{a}_{n,j'}\right] \\
&\sum_{m=1}^{|\mathcal{S}|}\sum_{j=1}^{|\mathcal{D}_m|} u_{m,j}\tilde{a}_{m,j}\underbrace{\sum_{n=1}^{|\mathcal{S}|}\sum_{j'=1}^{|\mathcal{D}_n|}\bar{a}_{n,j'}}_{=1} + \sum_{n=1}^{|\mathcal{S}|}\sum_{j'=1}^{|\mathcal{D}_n|} u_{n,j'}\tilde{a}_{n,j'}\underbrace{\sum_{m=1}^{|\mathcal{S}|}\sum_{j=1}^{|\mathcal{D}_m|}\bar{a}_{m,j}}_{=1} \\
\geq &\sum_{m=1}^{|\mathcal{S}|}\sum_{j=1}^{|\mathcal{D}_m|} u_{m,j}\bar{a}_{m,j}\underbrace{\sum_{n=1}^{|\mathcal{S}|}\sum_{j'=1}^{|\mathcal{D}_n|}\tilde{a}_{n,j'}}_{=1} + \sum_{n=1}^{|\mathcal{S}|}\sum_{j'=1}^{|\mathcal{D}_n|} u_{n,j'}\bar{a}_{n,j'}\underbrace{\sum_{m=1}^{|\mathcal{S}|}\sum_{j=1}^{|\mathcal{D}_m|}\tilde{a}_{m,j}}_{=1} \\
&2\cdot\sum_{m=1}^{|\mathcal{S}|}\sum_{j=1}^{|\mathcal{D}_m|} u_{m,j}\tilde{a}_{m,j} \geq 2\cdot\sum_{m=1}^{|\mathcal{S}|}\sum_{j=1}^{|\mathcal{D}_m|} u_{m,j}\bar{a}_{m,j} \\
&\underbrace{\sum_m u_m\tilde{a}_m}_{\text{FedAdp}} \overset{4}{\geq} \underbrace{\sum_m u_m a_m}_{\text{FedAvg}}.
\end{aligned}
\tag{3.17}
$$

where $u_m = u_{m,1} = \cdots = u_{m,|\mathcal{D}_m|}$. Inequality 4 holds because $\tilde{a}_m = \tilde{a}_{m,j} \cdot |\mathcal{D}_m|$ and $a_m = \bar{a}_{m,j} \cdot |\mathcal{D}_m|$ with $\tilde{a}_m$ and a_m denoting the weight for model aggregation in `FedAdp` and `FedAvg`, respectively. The equality 4 holds when $u_i = u_j, \forall k, j \in \mathcal{S}$.

Due to the greater expectation term in (3.1). `FedAdp` results in a greater decrease of FL loss in each global round, as compared to `FedAvg`. This completes the proof.

3.6 Summary

In this chapter, we have presented our design of `FedAdp` algorithm that assigns devices with different weights for updating the global model in each round adaptively to reduce

the communication rounds of FL training in the presence of non-i.i.d. data. We argue that non-i.i.d. data exacerbates the model divergence and observe that the devices with non-i.i.d. data make a smaller (or even negative) contribution to the global model aggregation than the devices with i.i.d. data. We have proposed to measure the device contribution based on the angle between the local gradient and global gradient and designed a non-linear mapping function to quantify device contribution. We have designed an adaptive weighting strategy that assigns weight proportional to device contribution instead of according to the size of local datasets. The simple yet effective strategy is able to reinforce positive (suppress negative) device contribution dynamically, leading to a significant communication round reduction. Its performance superiority over `FedAvg` is verified both theoretically and experimentally. We have shown that FL training with `FedAdp` has reduced the communication rounds by up to 54.1% on the MNIST dataset and up to 45.4% on the FashionMNIST dataset compared to `FedAvg`.

Chapter 4

Device Selection Toward Faster Convergence for Federated Learning on Non-IID Data

4.1 Overview

To improve the system efficiency, it is crucial to choose a subset of devices among a large number of candidate devices. Statistical heterogeneity makes random device selection [5] inefficient in FL training because the directions of minimizing local objectives may contradict each other [24]. Several works focus on device selection strategies based on different criteria to handle different scenarios. e.g., wireless channel [47, 49], computation capability [45], and data heterogeneity [49, 50, 52, 60]. Prominently, to handle the data heterogeneity problem, authors in [49, 52, 60] proposed to use the magnitude of the norm of local gradient as the indicator to measure the device's contribution and select the devices, which is inaccurate and ineffective, as analyzed theoretically and empirically in this chapter.

To balance the exploration and exploitation of the contribution from all candidate devices,

we adopt the probabilistic framework for device selection based on the correlation between local updates and the global update, which is better for profiling the device contribution, compared to methods in [46–49, 51, 52]. Particularly, for FL with the heterogeneous dataset, we analyze the convergence property of `FedAvg` theoretically and challenge the necessity of global model aggregation over all participating devices. Then, the `Optimal Aggregation` algorithm is proposed, which can identify and exclude the adverse local updates to make greater progress on reducing the expected decrement of global loss in each round. The FL with probabilistic device selection (`FedPNS`) is proposed based on the result of `Optimal Aggregation`. `FedPNS` adjusts the probability for each device to be selected, and the server is able to preferentially select devices that propel a faster model convergence. Note that our probabilistic device selection is conducted on the server side, which does not impose additional communication costs. Our main contributions in this work are as follows.

- We analyze the convergence bound of the commonly adopted (`FedAvg`) algorithm [5] from a theoretical perspective and derive the expected decrease of FL global loss, considering the data heterogeneity and the way to aggregate local updates.

- We challenge the necessity of global model aggregation over local updates of all participating nodes and propose `Optimal Aggregation` to identify and exclude the potential adverse local updates, which enlarges the expected decrease of global loss in each round.

- We design `FedPNS`, a probabilistic device selection scheme that enables the server to dynamically adjust the probability for each node to be selected in each round, based on the result of `Optimal Aggregation`. `FedPNS` tendentiously selects nodes that boost model convergence. The convergence rate improvement of `FedPNS` over `FedAvg` is illustrated theoretically, and the imposed computational complexity of `FedPNS` is discussed.

- We empirically evaluate the performance of `FedPNS` via extensive experiments using the synthetic dataset and real datasets with different learning objectives. The experimental results show the effectiveness of `FedPNS` in improving the convergence rate of the FL model compared with the commonly adopted `FedAvg` algorithm.

Section 4.2 provides a sanity check of `FedAvg`, which indicates the necessity of excluding the potential adverse local updates, laying the basis of probability adjustment of the proposed framework. Section 4.3 discusses how probabilistic device selection outperforms random device selection from the perspective of convergence property. Comprehensive experimental results and related complete proof are shown in Section 4.4 and Section 4.5. Finally, Section 4.6 summarizes this chapter.

4.2 Contribution-based Device Selection

4.2.1 Sanity Check of `FedAvg`

For theoretical analysis purposes, we employ Assumption 1 - 3, and the following Assumption 5 to the loss function, which have also been commonly made in the literature [19, 20, 22, 24].

Assumption 5. ***ϕ-local dissimilarity***. *Local loss functions $F_k(\mathbf{w}^t)$ are ϕ-local dissimilar at $\mathbf{w}^t$, i.e., $\mathbb{E}_{\mathcal{S}}\left[\|\nabla F_k(\mathbf{w}^t)\|^2\right] \leq \|\nabla F(\mathbf{w}^t)\|^2\phi^2$ for $k \in \mathcal{S}$ and $t = 1, \cdots, T$, where T is the number of global rounds. $\mathbb{E}_{\mathcal{S}}[\cdot]$ denotes the expectation over participating devices $\mathcal{S}$ with weight a_k (as in equation 2.1). $\nabla F(\mathbf{w}^t)$ is the global gradient at the t-th global round defined as $\nabla F(\mathbf{w}^t) = \sum_{k\in\mathcal{S}} a_k \nabla F_k(\mathbf{w}^t)$.*

The discrepancy between the local objective and global objective caused by the data heterogeneity is captured by Assumption 5, which has been made in previous work [20, 24]. As the data distribution across participating devices becomes more heterogeneous, the local

updates (i.e., gradient) will diverge from each other, and ϕ will increase. On the other hand, if the data samples on participating devices follow the same data distribution, the local gradients become more similar, and ϕ goes to 1.

Lemma 1. *Let assumptions 1 and 5 hold. Suppose that* $\mathbf{w}^t$ *is not a stationary solution; the expected decrement on the global loss of* `FedAvg` *between two consecutive rounds satisfies*

$$F(\mathbf{w}^{t+1}) \leq F(\mathbf{w}^t) - \eta \mathbb{E}_{\mathcal{S}} \left[\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle \right] + \frac{L\eta^2}{2} \|\nabla F(\mathbf{w}^t)\|^2 \phi^2, \tag{4.1}$$

where η *is the learning rate of SGD.*

The proof of Lemma 1 is presented in Section 4.5. Lemma 1 provides a bound on how rapid the decrease of the global FL loss can be expected. The decrease of global FL loss between two consecutive rounds shows a dependency on ϕ, which represents the variance between local data distributions, and the aggregation strategy $\mathbb{E}_{\mathcal{S}}[\cdot]$, where $\nabla F(\mathbf{w}^t)$ is obtained by aggregating over local updates from all participating devices, i.e., $\nabla F_k(\mathbf{w}^t), k \in \mathcal{S}$ with weight $a_k = |\mathcal{D}_k| / \sum_{k=1}^{|\mathcal{S}|} |\mathcal{D}_k|$.

4.2.2 Aggregation with Gradient Information

In vanilla `FedAvg` [5] and the subsequent work [12, 13, 19, 21], the averaging technique is used for global update aggregation due to its simplicity. One can challenge the inherent rule that the global update is aggregated over local updates of all participating devices since the local updates may contribute global model in an adverse way. As a sanity check, at any communication round t, the local update from the participating devices whose inner product between their gradients and global gradient is negative i.e., $\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle < 0$, will slow the model convergence because of the reduced expected loss decrement (i.e., a lower expectation value as in (4.1)) in this round. As such, it is not trivial to exclude the *adverse*

local updates, which is realized by examining the value of expectation term in Lemma 1, as illustrated later. Excluding adverse local updates gives an impact on the reduction of overall data heterogeneity, thus changing the relationship between the local gradient and the global gradient $\langle \nabla \hat{F}(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle$, where $\nabla \hat{F}(\mathbf{w}^t) = \frac{1}{|\tilde{\mathcal{S}}|} \sum_{k \in \mathcal{S}^*} \nabla F_k(\mathbf{w}^t)$ is defined over $\mathcal{S}^*$, i.e., the subset of participating devices $\mathcal{S}$ after successfully excluding the devices with adverse local updates.

To find the optimal subset of local updates to aggregate, we first check the expectation term $\mathbb{E}_{\mathcal{S}} \left[\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle \right]$ in Lemma 1 and exclude the local updates from participating devices k, i.e., $k \in \mathcal{S} - \tilde{\mathcal{S}}$ if $\mathbb{E}_{\tilde{\mathcal{S}}} \left[\langle \nabla \hat{F}(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle \right] > \mathbb{E}_{\mathcal{S}} \left[\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle \right]$ is satisfied. However, excluding local updates gives an impact on the global update and overall data heterogeneity, i.e., $\|\nabla F(\mathbf{w}^t)\|^2 \phi^2$, the last term on the right-hand side of (4.1), which makes the expected decrement of global loss, i.e., $\Delta F(\mathbf{w}^t) = \frac{L\eta^2}{2} \|\nabla F(\mathbf{w}^t)\|^2 \phi^2 - \eta \mathbb{E}_{\mathcal{S}} \left[\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle \right]$, difficult to be analyzed quantitatively given L and ϕ. Therefore, in the second step, test loss is adopted to ensure that excluding local updates makes global updates better in terms of model convergence, as in [50]. In particular, the global model $\mathbf{w}^{t+1}$ and $\tilde{\mathbf{w}}^{t+1}$ generated by $\nabla F_k(\mathbf{w}^t), k \in \mathcal{S}$ and $\nabla F_k(\mathbf{w}^t), k \in \tilde{\mathcal{S}}$, respectively, are evaluated using mini-batch of samples uniformly chosen at random from $\mathcal{D}_{test}$ (e.g., test dataset in MNIST).

An iterative algorithm called `Optimal Aggregation` is proposed for a better local update aggregation in each round, which finds the *optimal* subset of local update $\Delta_k, k \in \mathcal{S}^* \subseteq \mathcal{S}$ by excluding the adverse local updates Δ_k, $k \in \mathcal{S} \setminus \mathcal{S}^*$, as in Algorithms 2. Specifically, for a given set of participating devices $\mathcal{S}$ in each global round t, the server iteratively removes one of the local updates $\nabla F_k(\mathbf{w}^t), k \in \mathcal{S}$, generates the potential global gradient, and calculates the expectation term in (4.1) (i.e., Check Expectation, line 18-21). If excluding one local update gives a higher expectation value, compared with the case that includes all local updates retained in $\mathcal{S}$, that local update will be labeled, and loss comparison will be performed to check the loss criterion (Check Loss, line 22-25), otherwise the server

keeps all local updates (line 6). If the loss criterion is satisfied (line 13), the labeled local update is eventually removed from set $\mathcal{S}$ (line 14). Otherwise, the server keeps that local update retained in $\mathcal{S}$ (line 12). The process repeats until no adverse local update can be found or the number of remaining local updates is below a threshold v (line 4). In Algorithm 2, the function `pop` is defined as removing an element (line 14). The introduced "temp" is a dictionary with key-value pairs (line 5) and the function `max` returns the maximum value (line 6) or the key (i.e., the device index k) corresponding to that value (line 9), respectively.

Given a set of participating devices $\mathcal{S}$, the benefits of finding optimal local updates are twofold: (i) Excluding the potential local updates that contribute to the global model adversely results in a larger decrement of the expected loss in each round. (ii) By CHECK EXPECTATION, the potential adversarial devices $k, k \in \mathcal{S} \setminus \tilde{\mathcal{S}}$ (devices with non-i.i.d. dataset normally) are identified. This identification can be used for consequent probabilistic device selection, as illustrated in Section 4.2.3.

4.2.3 FL with Probabilistic Device Selection

Providing the variety of different devices on contributing global model, to improve the convergence rate, one can seek to preferentially select the devices with higher contribution (i.e., the devices with i.i.d. dataset, as observed in [5]). As such, we propose a probabilistic device selection design that dynamically changes the probability for each device to be selected in each communication round, based on their data distribution-related contribution, which can be distinguished by the procedure CHECK EXPECTATION in `Optimal Aggregation`.

As we know in each round of FL, a number of devices are selected to participate in the local update and global aggregation. It is natural to lower the device selection probabilities for those devices whose local updates slow model convergence. Therefore, on the server-side, we propose to dynamically change the probability for each device to be selected via using the

Algorithm 2 Optimal Local Updates for Aggregation

Procedure OPTIMAL AGGREGATION
Input: $\mathcal{S}$, Δ_k^t, v, temp $= \{\}$
$\nabla F(\mathbf{w}_k^t) = -\Delta_k^t/\eta$
$\nabla F(\mathbf{w}^t) = \frac{1}{|\tilde{\mathcal{S}}|}\sum_{k\in\mathcal{S}} \nabla F_k(\mathbf{w}^t)$
max $= \mathbb{E}_{\mathcal{S}}\left[\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t)\rangle\right]$
while $|\mathcal{S}| \geq v$ **do**
  temp $\leftarrow$ CHECK EXPECTATION ($\nabla F_k(\mathbf{w}^t)$, $\mathcal{S}$, temp)
  if `max`(temp).value $<$ max **do**
    break with $\mathcal{S}^* = \mathcal{S}$
  else
    key = `max`(temp).key
    ls($\mathbf{w}$), ls($\tilde{\mathbf{w}}$), $\tilde{\mathcal{S}} \leftarrow$ CHECK LOSS ($\nabla F_k(\mathbf{w}^t)$, $\mathcal{S}$, key)
    if ls($\mathbf{w}$) $>$ ls($\tilde{\mathbf{w}}$) **do**
      break with $\tilde{\mathcal{S}}$, $\mathcal{S}^* = \mathcal{S}$
    else
      $\mathcal{S}, \mathcal{S}^* \leftarrow \mathcal{S}$.`pop`(key)
      max $\leftarrow$ temp(key).value
return $\mathcal{S}^*$, $\tilde{\mathcal{S}}$
$\mathbf{w}^{t+1} \leftarrow$ GLOBAL UPDATE ($\nabla F_k(\mathbf{w}^t)$, $\mathcal{S}^*$)

Procedure CHECK EXPECTATION
Input: $\nabla F_k(\mathbf{w}^t)$, $\mathcal{S}$, temp
for $k = 1, \cdots, |\mathcal{S}|$ **do**
  $\tilde{\mathcal{S}} \leftarrow \mathcal{S}$.`pop`($\mathcal{S}[k]$)
  $\nabla \hat{F}(\mathbf{w}^t) = \frac{1}{|\tilde{\mathcal{S}}|}\sum_{k\in\tilde{\mathcal{S}}} \nabla F_k(\mathbf{w}^t)$
  temp($\mathcal{S}[k]$) $= \mathbb{E}_{\tilde{\mathcal{S}}}\left[\langle \nabla \tilde{F}(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t)\rangle\right]$

Procedure CHECK LOSS
Input: $\nabla F_k(\mathbf{w}^t)$, $\mathcal{S}$, key
$\tilde{\mathcal{S}} \leftarrow \mathcal{S}$.`pop`(key)
Generate global model $\mathbf{w}^{t+1}$ by $\nabla F_k(\mathbf{w}^t), k \in \mathcal{S}$ and $\hat{\mathbf{w}}^{t+1}$ by $\nabla F_k(\mathbf{w}^t), k \in \tilde{\mathcal{S}}$, respectively
Evaluate $\mathbf{w}^{t+1}$, $\hat{\mathbf{w}}^{t+1}$ by using mini-batch samples from $\mathcal{D}_{test}$ and get the loss ls($\mathbf{w}$) and ls($\hat{\mathbf{w}}$), respectively
return ls($\mathbf{w}$), ls($\hat{\mathbf{w}}$), $\tilde{\mathcal{S}}$

Procedure GLOBAL UPDATE
Input: $\nabla F_k(\mathbf{w}^t)$, $\mathcal{S}^*$
Generate $\mathbf{w}^{t+1}$ by $\nabla F_k(\mathbf{w}^t), k \in \mathcal{S}^*$ via weighted summation with weight a_k
return $\mathbf{w}^{t+1}$

output of `Optimal Aggregation` (i.e., $\tilde{\mathcal{S}}$). In particular, the probabilities for those devices that are labeled by the procedure CHECK EXPECTATION (i.e., $k \in \mathcal{S} - \tilde{\mathcal{S}}$) are decreased

according to the parameter x in (4.2), and the probabilities for all the rest devices will be increased.

$$\Delta p_k^t = p_k^t \cdot \texttt{min}[(x+\beta)^\alpha,\ 1], \quad k \in \mathcal{S} \setminus \tilde{\mathcal{S}}, \tag{4.2}$$

where p_k^t and Δp_k^t denote the probability for device k to be selected in the t-th global round and its probability decrement in the next round, respectively. `min` function returns the minimum value among all arguments, $x \in (0, 1]$ is defined as the ratio between the accumulated times that a device is labeled by the procedure Check Expectation and the accumulated times that the device is selected, $\alpha \in \mathbb{Z}^+, \beta \in [0, 1]$ are coefficients as explained in the following.

- $\lim_{x \to \epsilon}(x+\beta)^\alpha \approx 1$, where $\epsilon \propto \alpha$ is constant.
- $\lim_{0 \to x \to \upsilon}(x+\beta)^\alpha \approx \beta$, where $\upsilon \propto \alpha$ is a constant.

α controls how big the probability decrement is achieved by $(x+\beta)^\alpha$ given a ratio x. For example, a large value of α brings an aggressive decrement since the probability decrement happens in a wide range $(\beta, 1)$ as x increases within a small range (υ, ϵ), making the device selection probability drop very quickly when x grows. Meanwhile, the large α makes device selection sensitive to the identification mistake, which may prevent i.i.d. devices from being selected in the subsequent rounds. However, setting a small value of α is not consistently effective in differentiating the devices since the probability change is marginal. β is adopted to keep the rate of probability change in a visible range $[\beta, 1]$. From experiments, we find out $\alpha = 2, \beta = 0.7$ is a good choice that balances the tradeoff. The choice of α and β is empirically investigated in Section 4.4.

After getting the probability change for the labeled devices (i.e., $k \in \mathcal{S} \setminus \tilde{\mathcal{S}}$), we equally

Algorithm 3 FL with Probabilistic Device Selection

Procedure FEDERATED OPTIMIZATION
Input: $E, B, \eta, \mathcal{K}, T, p_k^t\ i = 1, \cdots, |\mathcal{K}|$
1: Server initializes $\mathbf{w}^1, p_k^1 = 1/|\mathcal{K}|$
2: **for** $t = 1, \cdots, T$ **do**
3: Server samples a subset $\mathcal{S}$ of devices according to p_k^{t-1}
4: Server sends $\mathbf{w}^t$ to devices $k \in \mathcal{S}$
5: Each device $k \in \mathcal{S}$ optimizes $F_k(\mathbf{w}^t)$ using SGD and sends back Δ_k^t to the server
6: $\mathbf{w}^{t+1}$, $\tilde{\mathcal{S}} \leftarrow$ OPTIMAL AGGREGATION
7: Server updates $p_k^t, k = 1, \cdots, |\mathcal{K}|$ by (4.2) and (4.3) for next round's usage
8: **return** $\mathbf{w}^T$
Procedure OPTIMAL AGGREGATION
Input: $\mathcal{S}$, Δ_k^t, v, temp $= \{\}$
9: Direct to Algorithm 2
10: **return** $\mathbf{w}^{t+1}$, $\tilde{\mathcal{S}}$

increase the probability for all the rest devices $k \in \mathcal{K} \setminus (\mathcal{S} \setminus \tilde{\mathcal{S}})$, as shown in (4.3).

$$p_k^{t+1} = \begin{cases} p_k^t - \Delta p_k^t & k \in \mathcal{S} \setminus \tilde{\mathcal{S}} \\ p_k^t + \frac{\sum_{k \in \mathcal{S} \setminus \tilde{\mathcal{S}}} \Delta p_k^t}{|\mathcal{K} \setminus (\mathcal{S} \setminus \tilde{\mathcal{S}})|} & k \in \mathcal{K} \setminus (\mathcal{S} \setminus \tilde{\mathcal{S}}) \end{cases}, \tag{4.3}$$

where $p_k^{t+1}, k \in \mathcal{K}$ are used for the $(t+1)$-th round.

We summarize the proposed FL design with probabilistic device selection and optimal aggregation in Algorithm 3. Particularly, in each commutation round t, after the server receives the local update from participating devices $k \in \mathcal{S}$, the server identifies the devices that are labeled by the procedure CHECK EXPECTATION (i.e., $\mathcal{S} - \tilde{\mathcal{S}}$) and the remaining devices for aggregation $\mathcal{S}^*$, which are used to regulate the probability for subsequent rounds (line 7) and aggregate the global model for this round (line 6).

4.3 Convergence Analysis

To facilitate theoretical analysis, we introduce the auxiliary parameter $\mathbf{v}^t$, which is optimized w.r.t. the global loss function $F(\mathbf{v})$ in the centralized setting. $\mathbf{v}^t$ is a virtual sequence since $F(\mathbf{v})$ is only observable when all data samples are available at a central place. We use $\tilde{\mathbf{w}}^t$ to denote model weight with full node participation, i.e., $\tilde{\mathbf{w}}^t = \sum_{k=1}^{|\mathcal{K}|} \frac{1}{|\mathcal{K}|}\mathbf{w}_k^t$. We define that $\mathbf{v}^t$ is "synchronized" with $\tilde{\mathbf{w}}^t$ at the beginning of each global round, i.e., at the beginning of the t-th global round, the initial value of $\mathbf{v}^t$ is set as $\mathbf{v}^{t-1} = \tilde{\mathbf{w}}^{t-1}$. At the end of the t-th global round, the update rule of the centralized SGD is as follows.

$$\mathbf{v}^t = \mathbf{v}^{t-1} - \eta \left[\frac{1}{|\mathcal{D}|} \sum_{z_{\cup,s} \in \mathcal{D}} - \left(\sum_{c=1}^{C} q(y_{\cup,s} = c) \mathbb{E}_{\mathbf{x}_{\cup,s}|y_{\cup,s}=c} \left[\log l_c(\mathbf{w}, \mathbf{x}_{\cup,s}, y_{\cup,s}) \right] \right) \right], \tag{4.4}$$

where $z_{\cup,s} = \{\mathbf{x}_{\cup,s}, y_{\cup,s}\}$ denotes the s-th sample of the centralized dataset $\mathcal{D}$ and $q(y_{\cup,s} = c)$ is the population distribution of the s-th sample over class c. For simplicity of the proof, we omit the sample index s and use $z_k = \{\mathbf{x}_k, y_k\}$ to represent the training sample from $\mathcal{D}_k$. Similarly, $z_\cup = \{\mathbf{x}_\cup, y_\cup\}$ represent the samples from $\mathcal{D}$.

We first quantify the weight divergence $\mathbb{E}_{\mathcal{S}}\|\mathbf{w}^t - \mathbf{v}^t\|$ between $\mathbf{w}^t$ and $\mathbf{v}^t$, for any global round $t, t = 1, \cdots, T$. Then, by combing the result in [21], we obtain the convergence rate of `FedPNS`.

Theorem 3. *Consider $\mathcal{K}$ local devices with equal data size, and the data samples on device $k \in \mathcal{K}$ follow the data distribution q_k. Let assumptions 3 and 5 hold. Assume a fixed number of local updates τ exists between two consecutive global rounds. Then, the weight divergence in* ***`FedPNS`*** *after the $(t-1)$-th synchronization satisfies,*

$$\mathbb{E}_{\mathcal{S}}\|\mathbf{w}^t - \mathbf{v}^t\| \leq \eta \sum_{k=1}^{|\mathcal{K}|} (p_i G_i + \frac{1}{|\mathcal{K}|} q_k^{dif} (\sum_{r=1}^{\tau-1} \varpi^r g_{max}(\mathbf{v}^{t\tau-1-r}))), \tag{4.5}$$

where $g_{max}(\mathbf{v})$ *is defined as* $max_{c=1}^{C}\|\nabla\mathbb{E}_{\mathbf{x}_\cup|y_\cup=c}\left[\log l_c(\mathbf{w},\mathbf{x}_\cup,y_\cup)\right]\|$*;* $\varpi = 1+\eta L$*.* q_k^{dif} *is defined as* $\sum_{c=1}^{C}\|(q_k(y_k=c)-q(y_\cup=c)\|$*.*

Remark 5. The weight divergence between $\mathbf{w}^t$ and $\mathbf{v}^t$ mainly comes from two parts, the bound of the norm of local gradient from each participating device, i.e., $\sum_{k=1}^{|\mathcal{K}|} p_k G_k$, and the weight divergence introduced by the difference between the data distribution on device and population distribution, i.e., q_k^{dif}. `FedPNS` preferentially selects devices with a smaller bounded gradient, which results in a smaller weight divergence, compared with device selection with equal probability in `FedAvg`, i.e., $\sum_{k=1}^{|\mathcal{K}|} p_k G_k \leq \sum_{k=1}^{|\mathcal{K}|} \frac{1}{|\mathcal{K}|} G_k$.

Theorem 4. *When* $\eta \leq \frac{1}{L}$*, compared with* ***`FedAvg`****,* ***`FedPNS`*** *with a smaller weight divergence achieves tighter upper bound after* T *global rounds, i.e.,* $F(\mathbf{w}^T)-F(\mathbf{w}^*)$*, where* $F(\mathbf{w}^*)$ *denotes the optimal model parameter that minimizes* $F(\mathbf{w})$*.*

Proof. Theorem 4 is proven by combing the weight divergence $\mathbb{E}_{\mathcal{S}}\|\mathbf{w}^t-\mathbf{v}^t\|$ in Theorem 3 with the result in [21, Theorem 2]. From Theorem 3, it is straightforward to see that the weight divergence $\mathbb{E}_{\mathcal{S}}\|\mathbf{w}^t-\mathbf{v}^t\|$ in `FedPNS` is smaller than that in `FedAvg`. From [21, Theorem 2], we have $F(\mathbf{w}^T)-F(\mathbf{w}^*) \propto \mathbb{E}_{\mathcal{S}}\|\mathbf{w}^t-\mathbf{v}^t\|$, i.e., a smaller weight divergence in each global round $t, t=1,\cdots,T$ results in a smaller gap between the global loss after T global round and the global loss with optimal model, $F(\mathbf{w}^T)-F(\mathbf{w}^*)$, which completes the proof.

4.4 Numerical Results

We now present empirical results for the proposed probabilistic device selection strategy. We implement `FedPNS` on different tasks, models, datasets, and compare with commonly used benchmark `FedAvg`. We first demonstrate the effectiveness of the proposed `Optimal Aggregation` in enlarging the expected decrement of FL global loss and in identifying the potential adversarial devices (Section 4.4.1). Then, the superiority of the proposed

`FedPNS` in the presence of different data heterogeneity is illustrated in Section 4.4.2. All code, data, and experiments are publicly available as an open-source GitHub repository at: github.com/HongdaWu1226/FedPNS.

We briefly describe our adopted datasets, learning model, and experiment setting as follows.

Synthetic data. To better characterize the data heterogeneity and study its impact on model convergence, we generate synthetic data by following a similar setup as in [19, 91]. In particular, the data samples $\{\mathbf{x}_k, y_k\}$ on local device k are generated according to the model $y_k = argmax(\text{softmax}(\mathbf{w}\mathbf{x}_k + b_k)), \mathbf{x}_k \in \mathbb{R}^{60}, \mathbf{w}_k \in \mathbb{R}^{10\times 60}, b_k \in \mathbb{R}^{10}$. We set $\mathbf{w}, b \backsim \mathcal{N}(0, 1)$. For the data on i.i.d. devices, $\mathbf{x}$ follows the same distribution $\mathcal{N}(0, \Sigma)$, where Σ is diagonal with $\Sigma_{h,h} = h^{-1.2}$. For the data samples on non-i.i.d. device k, $\mathbf{x}_k \backsim \mathcal{N}(o_k, \Sigma)$, each element in the mean vector o_k is drawn from $\mathcal{N}(B_k, 1), B_k \backsim N(0, \varrho)$. As such, a big value of ϱ denotes a more heterogeneous data scenario. The training set and testing set are randomly split with $80\% - 20\%$ proportion on each device. A Multinomial Logistic Regression (MLR) model is applied to the synthetic data.

Real data. We explore different learning objectives on different real datasets, which are considered in prior works [5, 19]. In Section 4.4.2, we start with a convex classification problem with MNIST [92] using MLR model. Then, for the non-convex setting, we consider two CNN models for MNIST and CIFAR-10[7] [93], which are referred as CNN-M[8] and CNN-C[9] hereinafter.

[7]CIFAR-10 dataset contains 60,000 32×32 color images in 10 different classes. The 10 different classes represent airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. There are 6,000 images of each class.

[8]The CNN-M model has 7 layers with the following structure: $5 \times 5 \times 10$ Convolutional $\rightarrow 2 \times 2$ MaxPool $\rightarrow 5 \times 5 \times 20$ Convolutional $\rightarrow 2 \times 2$ MaxPool $\rightarrow 320 \times 50$ Fully connected $\rightarrow 50 \times 10$ Fully connected $\rightarrow$ Softmax. The second convolutional layer is with 50% dropout. All Convolutional and Fully connected layers are mapped by ReLu activation.

[9]The CNN-C model has 8 layers as structured follows: $5 \times 5 \times 6$ Convolutional $\rightarrow 2 \times 2$ MaxPool $\rightarrow 5 \times 5 \times 16$ Convolutional $\rightarrow 2 \times 2$ MaxPool $\rightarrow 400 \times 120$ Fully connected $\rightarrow 120 \times 84$ Fully connected $\rightarrow 84 \times 10$ Fully connected $\rightarrow$ Softmax. ReLu activation is applied to all layers.

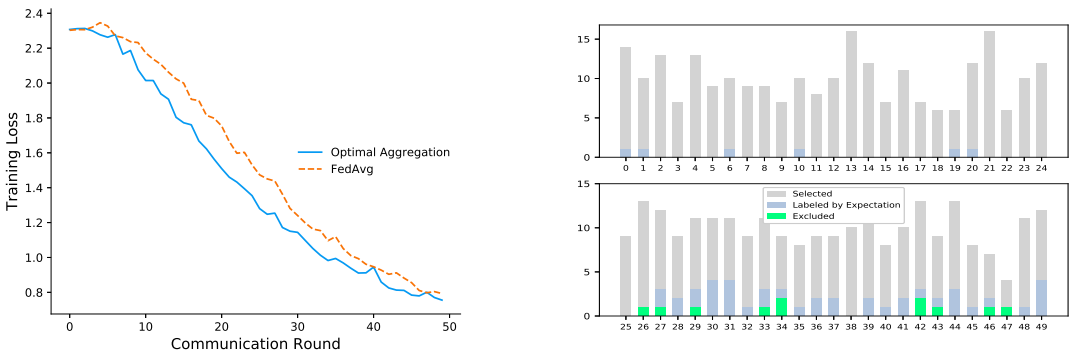

Figure 4.1: Performance of the proposed `Optimal Aggregation`. (1) Left: The training loss on the MNIST dataset when different aggregation strategies are adopted. `Optimal Aggregation` and `FedAvg` aggregate local updates over $\mathcal{S}^*$ and $\mathcal{S}$, respectively. (2) Right: We use a triple to observe the result of `Optimal Aggregation`, which includes the accumulated times that each device is selected, labeled by CHECK EXPECTATION, and excluded eventually by CHECK LOSS during FL model training. The upper and bottom row refer to the results for i.i.d. devices and non-i.i.d. devices, respectively.

Through the experimental result, unless otherwise specified, we evaluate the accuracy of the trained models using the testing set from each dataset. The fraction for selecting devices is set to be 0.2, $|\mathcal{S}| = 10$, $|\mathcal{D}_k| = 200$, $E = 1$, $T = 200$, $\eta = 0.01$, decay rate $= 0.995$, $v = 0.7$, batch size in local training and CHECK LOSS are 20 and 128, respectively. For real datasets, the overall data heterogeneity is measured by σ and the skewness of dataset on non-i.i.d. devices is represented by ρ. For example, $\sigma = 0.2, \rho = 2$ means that $\sigma|\mathcal{K}| = 10$ devices are equipped with i.i.d. dataset, where non-i.i.d. dataset lay on the rest $(1-\sigma)|\mathcal{K}| = 40$ devices, and the data samples on which are evenly belong to 2 labels. As such, a small σ, ρ indicates a higher data heterogeneity.

4.4.1 Performance of `Optimal Aggregation`

In this part, we conduct an experiment to illustrate the performance of the proposed `Optimal Aggregation` algorithm. Particularly, we adopt the CNN-M model on the MNIST dataset where the data heterogeneity is set to be $\sigma = 0.5, \rho = 1$. In each global round,

we randomly select $|\mathcal{S}| = 10$ devices while guaranteeing the participating devices include half i.i.d. devices and half non-i.i.d. devices. To avoid the randomness of device selection, the participating devices in each round are kept the same for `FedAvg` [5] and the proposed `Optimal Aggregation` algorithm.

As shown in the upper part of Fig. 4.1, the proposed `Optimal Aggregation` algorithm can achieve lower training loss than `FedAvg`. When the global model is not robust in the several initial rounds, the local updates are more diverse due to the data heterogeneity; thus, excluding adverse local updates is more effective. We count the accumulated times that each device is selected, labeled by the procedure CHECK EXPECTATION (line 7 in Algorithm 2), and finally excluded by the procedure CHECK LOSS (line 14 in Algorithm 2). As we can see from the bottom part of Fig. 4.1, i) the i.i.d devices (i.e., with index "0", $\cdots$, "24") are never been excluded, yet some of the non-i.i.d devices (e.g.,"26", "27", "34", etc.) have been excluded for many times. ii) Almost all non-i.i.d. devices were labeled at least one time, which illustrates the effectiveness of `Optimal Aggregation` in identifying the devices with the skewed dataset.

4.4.2 Data Heterogeneity

In this part, we use different combinations of σ and ρ to investigate the performance of the proposed `FedPNS` scheme in the presence of different data heterogeneity. Through all experiments, α and β are chosen to be 2 and 0.7, respectively.

MLR Model with Synthetic Data We follow the description in Section 4.4 to generate synthetic data samples. The ratio of i.i.d. devices is set to be $\sigma = 0.2, 0.3$, and 0.5 with $\varrho = 0.5$ and 1. For each device k, the number of data samples $|\mathcal{D}_k| = 1000$, and the number of epochs for local training is $E = 20$. In Fig. 4.2, we study how data heterogeneity affects model convergence using MLR model and synthetic dataset. As we can see from Fig. 4.2, as

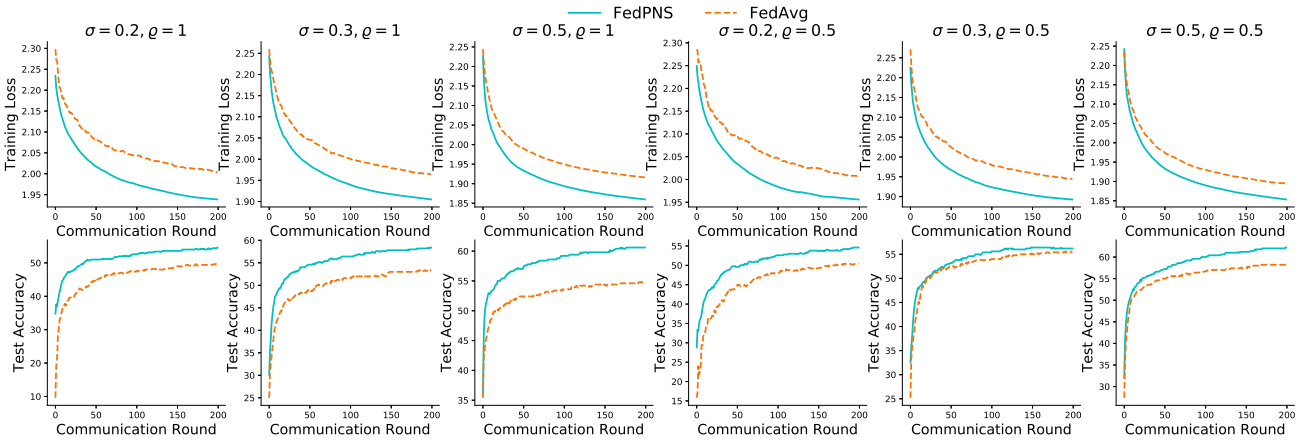

Figure 4.2: Effect of data heterogeneity on convergence. (1) Top row: we show the training loss on the synthetic dataset whose data heterogeneity decreases from left to right (with a fixed σ or ϱ). (1) Bottom row: we show the corresponding test accuracy.

the data heterogeneity increases, i.e., $\sigma = 0.5, 0.3$ and 0.2 with fixed $\varrho = 1$ or 0.5, `FedAvg` slows to converge (i.e., higher training loss) with a decreasing test accuracy in the meantime. `FedPNS` achieves a lower training loss and higher test accuracy, compared with `FedAvg` in all data setting.

MLR, CNN-M Model for MNIST

As we can tell from Fig. 4.3, `FedPNS` converges faster and achieves a higher test accuracy, compared with `FedAvg` for both MLR and CNN model regardless of different data heterogeneity. `FedPNS` achieves better improvement when the CNN model is adopted, compared with the scenario when the MLR model is utilized, which is attributed to the limited learning capability of MLR. In addition, it is observable that as the data becomes more heterogeneous, the performance enhancement is enlarged (i.e., α decreases from 0.5 to 0.2 for a given β, or β changes from 2 to 1 for a given α). When the number of i.i.d. devices is limited and the non-i.i.d devices are equipped with highly skewed dataset (e.g., $\sigma = 0.2, \rho = 1$ and $\sigma = 0.3, \rho = 1$), `FedPNS` gains remarkable performance improvement, which verifies the effectiveness of `FedPNS`

in identifying and selecting the devices that contribute global model better. For the scenario with the lowest data heterogeneity (i.e., $\sigma = 0.5, \rho = 2$), the performance gap between `FedPNS` and `FedAvg` is not obvious. This is because the impact of the non-i.i.d. devices on the convergence is reduced when a large number of i.i.d. devices can be selected.

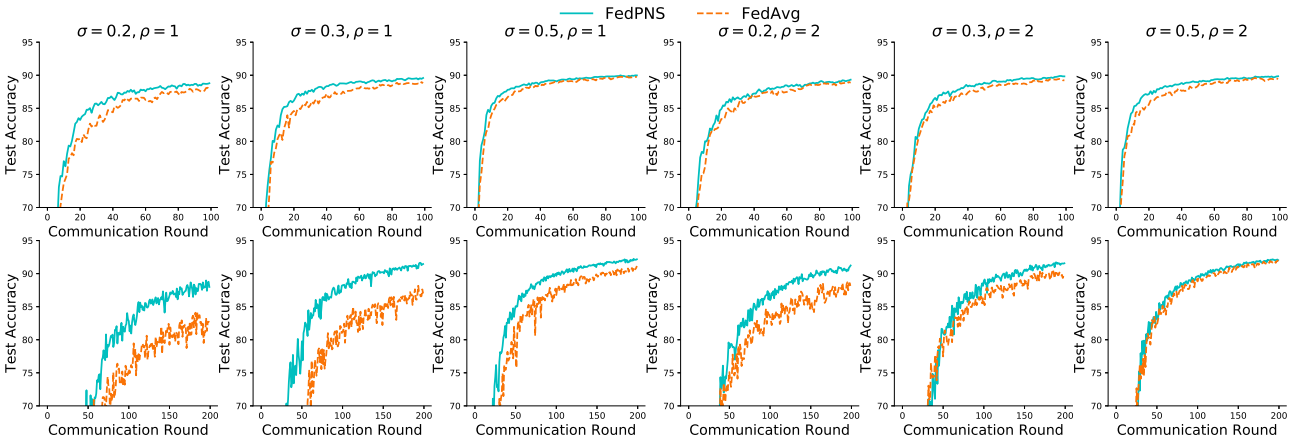

Figure 4.3: Test accuracy over communcation rounds of `FedPNS` and `FedAvg` with different data heterogeneity. Upper and lower subplots correspond to training performance when the MLR model and CNN-M model are adopted for MNIST, respectively. A smaller σ, ρ indicates a higher data heterogeneity.

CNN-C Model for CIFAR-10

For the more complex three-channel image classification task, the number of local epochs is set to be $E = 5$. As we can see from Fig. 4.4, compared with `FedAvg`, `FedPNS` converges faster and leads to a higher test accuracy, especially for the high data heterogeneity scenario (i.e., $\sigma = 0.2$ and $0.3, \rho = 1$). The performance improvement of `FedPNS` is not obvious when $\sigma = 0.2, \rho = 2$; this is because the small number of i.i.d. devices with less heterogeneous data samples on non-i.i.d. devices makes `FedPNS` hard to distinguish the device contribution.

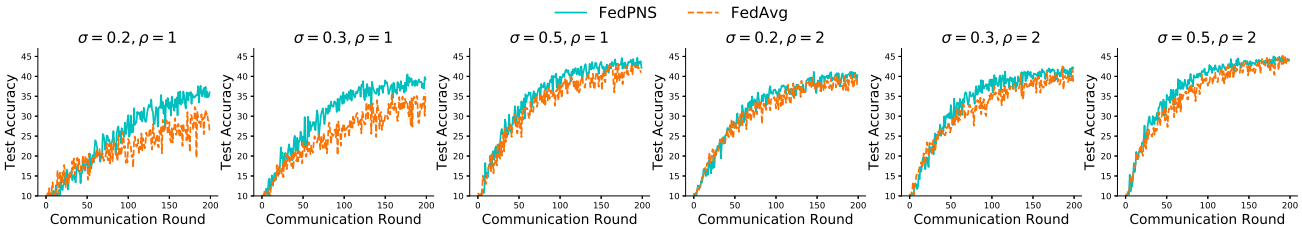

Figure 4.4: Test accuracy over communcation rounds of `FedPNS` and `FedAvg` with different data heterogeneity. CNN-C model is adopted for CIFAR-10.

4.4.3 Choosing α and β

The choice of α and β gives an impact on `FedPNS`. As discussed in Chapter 4.2.3, a large value of α can help increase the model convergence rate by aggressively adjusting the device probability. On the other hand, a large value of α also makes device selection sensitive to the identification mistake, which may negatively impact the convergence. A similar effect is achieved by β, which keeps the rate of probability change in a range $[\beta, 1]$. We studied the effect of different α and β via heuristically choosing $\alpha \in \mathbb{Z}^+, \beta \in [0, 1]$ in ascending order. From the top row of Fig. 4.5, for a fixed $\beta = 0.7$, increasing α from 1 to 2 boosts performance. However, keep increasing α does not consistently embrace performance gain, this is because `FedPNS` becomes more sensitive to identification mistakes, which may prevent i.i.d devices from being selected in the subsequent rounds. Similarly, from the bottom plot of Fig. 4.5, for a fixed $\alpha = 2$, increasing β from 0.5 to 0.7 promotes model performance. However, further increasing β to 0.8 leads to a degraded performance. Empirically, we find $\alpha = 2, \beta = 0.7$ that balances the tradeoff and leads to the best performance.

4.4.4 Other Comparison

In this section, we take one experimental case as an example to demonstrate the bounded norm of local gradient $\|\nabla F_k(\mathbf{w}^t)\|$, which is related to the data distribution on each device. Besides, we compare the proposed `FedPNS` with another device selection scheme `BN2` [46],

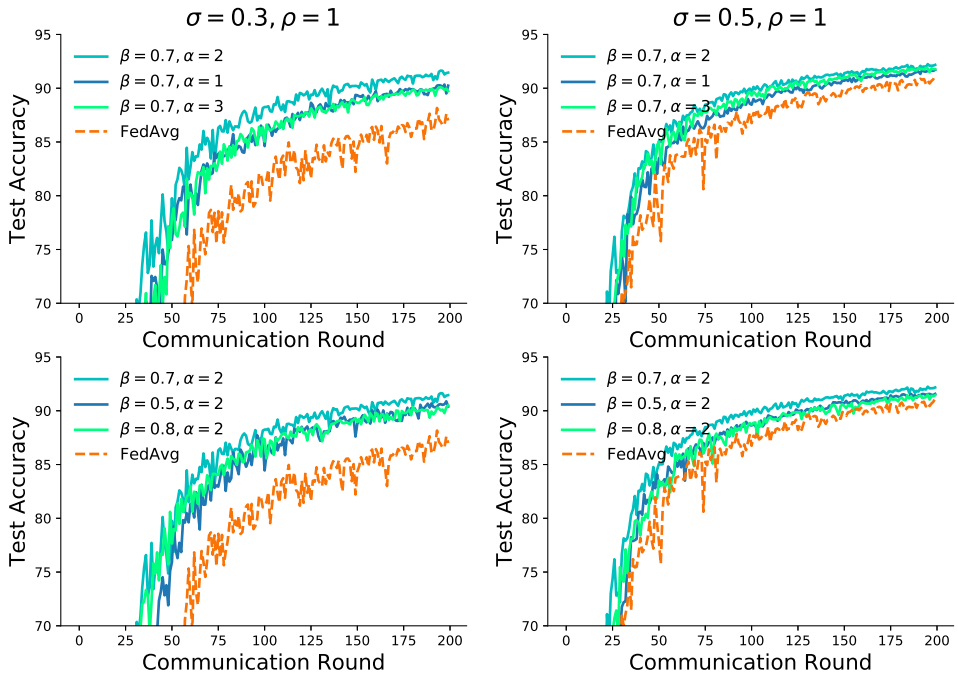

Figure 4.5: Effect of adopting different α and β. We heuristically choosing $\alpha \in \mathbb{Z}^+, \beta \in [0, 1]$ in ascending order. The top row and bottom row correspond to the performance with varied α and β, respectively. CNN-M on MNIST is adopted.

which chooses the devices with higher $\|\nabla F_k(\mathbf{w}^t)\|$ for aggregation. Specifically, in each global round, `BN2` first randomly selects $|\mathcal{M}|$ devices for local training. After that, the participating devices send their gradient norm $\|\nabla F_k(\mathbf{w}^t)\|, k \in \mathcal{M}$ to the server. The server chooses the first $|\mathcal{S}_t|$ local updates for model aggregation by sorting $\|\nabla F_k(\mathbf{w}^t)\|, k \in \mathcal{M}$ in descending order.

In this experiment, $|\mathcal{M}|$ is set to be 20. We track the norm of the gradient for each participating device $k \in \mathcal{M}$ statistically in each global round. As we can see from Fig. 4.6, the averaged gradient norm from i.i.d. devices is smaller than that from non-i.i.d. devices. This is because the data distribution on i.i.d. devices is more similar to the population distribution that is defined over all devices. As such, preferentially scheduling the devices

with a higher norm of gradient would slow the convergence, as shown in the bottom of Fig. 4.6.

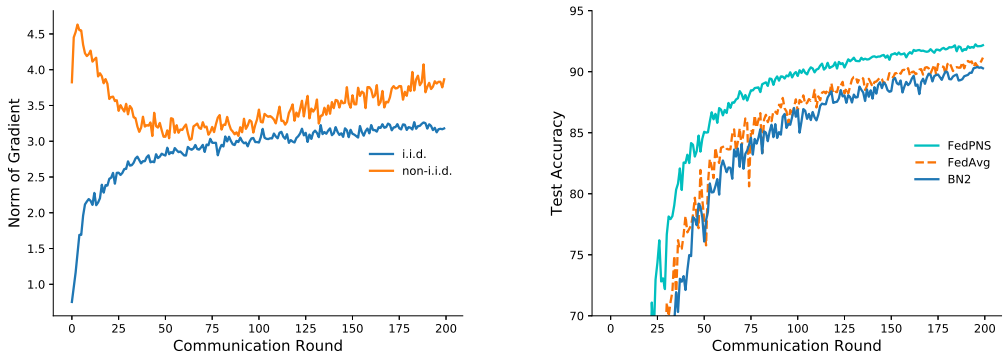

Figure 4.6: Device selection design with different importance indicators. `FedPNS` chooses devices by measuring the data distribution on local devices, while `BN2` selects devices according to the norm of gradient. (1) Top plot: we track the averaged gradient norm of device $k \in \mathcal{M}$ with different data distribution, where each device is selected from $\mathcal{K}$ randomly. (2) Bottom plot: we compare the test accuracy for different device selection designs. CNN-M on MNIST is adopted with $\sigma = 0.5, \rho = 1$.

4.5 Complete Proof

4.5.1 Proof of Lemma 1

From the L-smooth of $F(\mathbf{w})$ and applying Taylor expansion, we have

$$F(\mathbf{w}^{t+1}) \leq F(\mathbf{w}^t) + \langle \nabla F(\mathbf{w}^t), \mathbf{w}^{t+1} - \mathbf{w}^t \rangle + \frac{L}{2} \|\mathbf{w}^{t+1} - \mathbf{w}^t\|^2. \tag{4.6}$$

- Bounding $\|\mathbf{w}^{t+1} - \mathbf{w}^t\|^2$

By the definition of the global aggregation and local SGD, we have

$$\|\mathbf{w}^{t+1} - \mathbf{w}^t\|^2 = (\mathbb{E}_{\mathcal{S}}\left[\|\mathbf{w}^{t+1} - \mathbf{w}^t\|\right])^2 = \eta^2(\mathbb{E}_{\mathcal{S}}\left[\|\nabla F_k(\mathbf{w}^t)\|\right])^2 \overset{1}{\leq} \eta^2\mathbb{E}_{\mathcal{S}}\left[\|\nabla F_k(\mathbf{w}^t)\|^2\right]$$
$$\leq \eta^2\|\nabla F(\mathbf{w}^t)\|^2\phi^2, \quad (4.7)$$

where inequality 1 holds because of Cauchy-Schwarz inequality, and the last inequality is due to the bounded dissimilarity assumption.

- Bounding $\langle \nabla F(\mathbf{w}^t), \mathbf{w}^{t+1} - \mathbf{w}^t \rangle$

Again, by the definition of the global aggregation and SGD optimization, we have

$$\langle \nabla F(\mathbf{w}^t)), \mathbf{w}^{t+1} - \mathbf{w}^t) \rangle = -\eta\mathbb{E}_{\mathcal{S}}\left[\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle\right]. \quad (4.8)$$

Plugging (4.7) and (4.8) into (4.6), we obtain

$$F(\mathbf{w}^{t+1}) - F(\mathbf{w}^t) \leq -\eta\mathbb{E}_{\mathcal{S}}\left[\langle \nabla F(\mathbf{w}^t), \nabla F_k(\mathbf{w}^t) \rangle\right] + \frac{L\eta^2}{2}\|\nabla F(\mathbf{w}^t)\|^2\phi^2. \quad (4.9)$$

4.5.2 Proof of Theorem 4

At any global round t, the weight divergence between the model $\mathbf{w}^t$ with partial device participation and centralized model $\mathbf{v}^t$ is bounded as follows

$$\mathbb{E}_{\mathcal{S}}\|\mathbf{w}^t - \mathbf{v}^t\| = \mathbb{E}_{\mathcal{S}}\|\mathbf{w}^t - \tilde{\mathbf{w}}^t + \tilde{\mathbf{w}}^t - \mathbf{v}^t\| \leq \mathbb{E}_{\mathcal{S}}\|\mathbf{w}^t - \tilde{\mathbf{w}}^t\| + \|\tilde{\mathbf{w}}^t - \mathbf{v}^t\|. \quad (4.10)$$

- Bounding $\|\tilde{\mathbf{w}}^t - \mathbf{v}^t\|$

In this part, to facilitate analysis, we introduce the index of local update, e.g., the models $\tilde{\mathbf{w}}^t$ and $\mathbf{v}^t$ are represented by $\tilde{\mathbf{w}}^{t\tau}$ and $\mathbf{v}^{t\tau}$ since τ times of local SGD are applied in each global round.

Based on the definition of $\tilde{\mathbf{w}}^t$ and $\mathbf{v}^t$, we have $\|\tilde{\mathbf{w}}^t - \mathbf{v}^t\| = \|\tilde{\mathbf{w}}^{t\tau} - \mathbf{v}^{t\tau}\|$ and

$$
\begin{aligned}
\|\tilde{\mathbf{w}}^{t\tau} - \mathbf{v}^{t\tau}\| =& \|\sum_{k=1}^{|\mathcal{K}|} \frac{|\mathcal{D}_i|}{\sum_{k=1}^{|\mathcal{K}|} |\mathcal{D}_i|} \mathbf{w}_k^{t\tau} - \mathbf{v}^{t\tau}\| \\
\overset{1}{=}& \|\sum_{k=1}^{|\mathcal{K}|} \frac{1}{|\mathcal{K}|} (\mathbf{w}_k^{t\tau-1} - \eta \nabla F_k(\mathbf{w}_k^{t\tau-1})) - \mathbf{v}^{t\tau-1} + \eta \nabla F(\mathbf{v}^{t\tau-1})\| \\
\overset{2}{\leq}& \|\sum_{k=1}^{|\mathcal{K}|} \frac{1}{|\mathcal{K}|} \mathbf{w}_k^{t\tau-1} - \mathbf{v}^{t\tau-1}\| + \eta \|\sum_{k=1}^{|\mathcal{K}|} \frac{1}{|\mathcal{K}|} \sum_{c=1}^{C} q_k(y=c) \\
& (\nabla \mathbb{E}_{\mathbf{x}_k | y_k = c} \left[\log l_c(\mathbf{w}_k^{t\tau-1}, \mathbf{x}_k, y_k)\right] - \nabla \mathbb{E}_{\mathbf{x}_\cup | y_\cup = c} \left[\log l_c(\mathbf{v}^{t\tau-1}, \mathbf{x}_\cup, y_\cup)\right])\| \\
\overset{3}{=}& \|\sum_{k=1}^{|\mathcal{K}|} \frac{1}{|\mathcal{K}|} \mathbf{w}_k^{t\tau-1} - \mathbf{v}^{t\tau-1}\| + \eta \|\sum_{k=1}^{|\mathcal{K}|} \frac{1}{|\mathcal{K}|} (\nabla F_k(\mathbf{w}_k^{t\tau-1}) - \nabla F_k(\mathbf{v}^{t\tau-1}))\| \\
\overset{4}{\leq}& \sum_{k=1}^{|\mathcal{K}|} \frac{1}{|\mathcal{K}|} (1 + \eta L) \|\mathbf{w}_k^{t\tau-1} - \mathbf{v}^{t\tau-1}\|,
\end{aligned} \tag{4.11}
$$

where equality 1 holds by the updating rule of SGD, and by that, all devices have equal data size. Inequality 2 holds by applying triangle inequality and by the observation that for each class, the data distribution over all devices is the same as the distribution over the whole data samples, i.e., $j \in [C], q(y_\cup = c) = \sum_{k=1}^{|\mathcal{K}|} \frac{1}{|\mathcal{K}|} q_k(y_k = c)$. Equality 3 holds by (2.1), (2.3) and (4.4). and inequality 4 holds by Assumption 2 that the local loss function is L-smooth.

For device $k \in \mathcal{K}$, $\|\mathbf{w}_k^{t\tau-1} - \mathbf{v}^{t\tau-1}\|$ is bounded as

$$
\begin{aligned}
&\|\mathbf{w}_k^{t\tau-1} - \mathbf{v}^{t\tau-1}\| \\
=&\|\mathbf{w}_k^{t\tau-2} - \eta\nabla F_k(\mathbf{w}_k^{t\tau-2}) - \mathbf{v}^{t\tau-2} + \eta\nabla F(\mathbf{v}^{t\tau-2})\| \\
\leq&\|\mathbf{w}_k^{t\tau-2} - \mathbf{v}^{t\tau-2}\| + \eta\|\sum_{c=1}^{C} q_k(y_k = c)\nabla\mathbb{E}_{\mathbf{x}_k|y_k=c}\left[\log l_c(\mathbf{w}_k^{t\tau-2}, \mathbf{x}_k, y_k)\right] \\
&- \sum_{c=1}^{C} q(y_\cup = c)\nabla\mathbb{E}_{\mathbf{x}_\cup|y_\cup=c}\left[\log l_c(\mathbf{v}^{t\tau-2}, \mathbf{x}_\cup, y_\cup)\right]\| \\
\overset{5}{\leq}&\|\mathbf{w}_k^{t\tau-2} - \mathbf{v}^{t\tau-2}\| + \eta\|\sum_{c=1}^{C} q_k(y_k = c) \\
&(\nabla\mathbb{E}_{\mathbf{x}_k|y_k=c}\left[\log l_c(\mathbf{w}_k^{t\tau-2}, \mathbf{x}_k, y_k)\right] - \nabla\mathbb{E}_{\mathbf{x}_\cup|y_\cup=c}\left[\log l_c(\mathbf{v}^{t\tau-2}, \mathbf{x}_\cup, y_\cup)\right])\| \\
&+ \eta\|\sum_{c=1}^{C}(q_k(y_k = c) - q(y_\cup = c))\nabla\mathbb{E}_{\mathbf{x}_\cup|y_\cup=c}\left[\log l_c(\mathbf{v}^{t\tau-2}, \mathbf{x}_\cup, y_\cup)\right]\| \\
\overset{6}{=}&\|\mathbf{w}_k^{t\tau-2} - \mathbf{v}^{t\tau-2}\| + \eta\|\nabla F_k(\mathbf{w}_k^{t\tau-2}) - \nabla F_k(\mathbf{v}^{t\tau-2})\| \\
&+ \eta\|\sum_{c=1}^{C}(q_k(y_k = c) - q(y_\cup = c))\nabla\mathbb{E}_{\mathbf{x}_\cup|y_\cup=c}\left[\log l_c(\mathbf{v}^{t\tau-2}, \mathbf{x}_\cup, y_\cup)\right]\| \\
\overset{7}{\leq}&(1 + \eta L)\|\mathbf{w}_k^{t\tau-2} - \mathbf{v}^{t\tau-2}\| + \eta g_{max}(\mathbf{v}^{t\tau-2})\sum_{c=1}^{C}\|(q_k(y_k = c) - q(y_\cup = c))\|,
\end{aligned} \tag{4.12}
$$

where inequality 5 holds by introducing a term $\sum_{c=1}^{C} q_k(y_k = c)\nabla\mathbb{E}_{\mathbf{x}_\cup|y_\cup=c}\left[\log l_c(\mathbf{v}^{t\tau-2}, \mathbf{x}_\cup, y_\cup)\right]$ (with add and minus) and applying triangle inequality. Equality 6 holds by (2.1), (2.3) and (4.4). Inequality 7 holds by the L-smoothness in Assumption 2 and by defining $g_{max}(\mathbf{v}^{t\tau-2}) = max_{c=1}^{C}\|\nabla\mathbb{E}_{\mathbf{x}_\cup|y_\cup=c}\left[\log l_c(\mathbf{v}^{t\tau-2}, \mathbf{x}_\cup, y_\cup)\right]\|$.

Based on (4.12), by mathematical induction and setting $\varpi = 1 + \eta L$, we have

$$
\begin{aligned}
&\|\mathbf{w}_k^{t\tau-1} - \mathbf{v}^{t\tau-1}\| \\
\leq\quad& \varpi\|\mathbf{w}_k^{t\tau-2} - \mathbf{v}^{t\tau-2}\| + \eta\sum_{c=1}^{C}\|(q_k(y_k = c) - q(y_\cup = c))\|g_{max}(\mathbf{v}^{t\tau-2}) \\
\leq\quad& \varpi^2\|\mathbf{w}_k^{t\tau-3} - \mathbf{v}^{t\tau-3}\| + \eta\sum_{c=1}^{C}\|(q_k(y_k = c) - q(y_\cup = c))\|(g_{max}(\mathbf{v}^{t\tau-2}) + \varpi g_{max}(\mathbf{v}^{t\tau-3}))
\end{aligned}
$$

$$\vdots$$

$$\leq \quad \varpi^{\tau-1}\|\mathbf{w}_k^{(t-1)\tau} - \mathbf{v}^{(t-1)\tau}\| + \eta\sum_{c=1}^{C}\|(q_k(y_k=c) - q(y_\cup = c))\|(\sum_{r=0}^{\tau-2}\varpi^k g_{max}(\mathbf{v}^{t\tau-2-r})). \quad (4.13)$$

Substituting (4.13) to (4.11), we obtain

$$\begin{aligned}\|\tilde{\mathbf{w}}^t - \mathbf{v}^t\| \leq \sum_{K=1}^{|\mathcal{K}|}\frac{1}{|\mathcal{K}|}(\varpi^\tau\|\mathbf{w}_k^{(t-1)\tau} - \mathbf{v}^{(t-1)\tau}\| \\ + \eta\sum_{c=1}^{C}\|(q_k(y_k=c) - q(y_\cup = c))\|(\sum_{r=1}^{\tau-1}\varpi^r g_{max}(\mathbf{v}^{t\tau-1-r}))). \quad (4.14)\end{aligned}$$

Since $\mathbf{v}^t$ is "synchronized" with $\tilde{\mathbf{w}}^t$ at the beginning of each global round, we ignore the first item of the right hand side of (4.14), which is the weight divergence accumulated from the previous round. Thus, the weight divergence $\|\tilde{\mathbf{w}}^t - \mathbf{v}^t\|$ between two consecutive global round is represented as

$$\|\tilde{\mathbf{w}}^t - \mathbf{v}^t\| \leq \eta\sum_{k=1}^{|\mathcal{K}|}\frac{1}{|\mathcal{K}|}q_k^{dif}(\sum_{r=1}^{\tau-1}a^k g_{max}(\mathbf{v}^{t\tau-1-r}), \quad (4.15)$$

where $q_k^{dif} = \sum_{c=1}^{C}\|(q_k(y_k=c) - q(y_\cup = c))\|$.

• Bounding $\|\mathbf{w}^t - \tilde{\mathbf{w}}^t\|$: We follow the identical sampling distribution (i.e., $\{p_1, p_2, \cdots, p_{|\mathcal{K}|}\}$) to select $|\mathcal{S}|$ devices from $|\mathcal{K}|$ devices and let $\mathcal{S} = \{\mathrm{k}_1, \cdots, \mathrm{k}_{|\mathcal{S}|}\}$ denote the set of indices of chosen devices. The global model in FL with partial device participation is represented as $\mathbf{w}^t = \frac{1}{|\mathcal{S}|}\sum_{k=1}^{|\mathcal{S}|}\mathbf{w}_{\mathrm{k}_k}^t$. Taking expectation over $\mathcal{S}$, we have

$$\mathbb{E}_{\mathcal{S}}\|\mathbf{w}^t - \tilde{\mathbf{w}}^t\| = \mathbb{E}_{\mathcal{S}}\frac{1}{|\mathcal{S}|}\sum_{k=1}^{|\mathcal{S}|}\|\mathbf{w}_{\mathrm{k}_k}^t - \tilde{\mathbf{w}}^t\| = \sum_{k=1}^{|\mathcal{K}|}p_k\|\mathbf{w}_k^t - \tilde{\mathbf{w}}^t\|, \quad (4.16)$$

where the last equality in (4.16) is obtained by the following the observation $\mathbb{E}_{\mathcal{S}}\sum_{k\in\mathcal{S}}x_k = \mathbb{E}_{\mathcal{S}}\sum_{k=1}^{|\mathcal{S}|}x_{\mathrm{k}_k} = |\mathcal{S}|\mathbb{E}_{\mathcal{S}}x_{\mathrm{k}_k} = |\mathcal{S}|\sum_{k=1}^{|\mathcal{K}|}p_k x_k$ given $\mathcal{S} = \{x_{\mathrm{k}_1}, \cdots, x_{\mathrm{k}_{|\mathcal{S}|}}\} \subset \mathcal{K}$, and by replacing x_k with $\mathbf{w}_k^t$ in the above observation.

We consider the model parameter in the previous global round $\mathbf{w}_k^{t-1}$, which is identical for any $k \in \mathcal{K}$. As such, we have $\sum_{k=1}^{|\mathcal{K}|} p_k(\mathbf{w}_k^t - \mathbf{w}^{t-1}) = \tilde{\mathbf{w}}^t - \tilde{\mathbf{w}}^{t-1}$. Thus, the above equation can be bounded as

$$\begin{aligned}\sum_{k=1}^{|\mathcal{K}|} p_k \|\mathbf{w}_k^t - \tilde{\mathbf{w}}^t\| &= \sum_{k=1}^{|\mathcal{K}|} p_k \|(\underbrace{\mathbf{w}_k^t - \tilde{\mathbf{w}}^{t-1}}_{X}) - (\tilde{\mathbf{w}}^t - \tilde{\mathbf{w}}^{t-1})\| \\ &\leq \sum_{k=1}^{|\mathcal{K}|} p_k \|\mathbf{w}_k^t - \tilde{\mathbf{w}}^{t-1}\|, \end{aligned} \tag{4.17}$$

where the last equality holds because $\mathbb{E}\|X - \mathbb{E}[X]\| \leq \mathbb{E}\|X\|$.

Substituting (4.17) into (4.16), we have,

$$\begin{aligned}\mathbb{E}_{\mathcal{S}}\|\mathbf{w}^t - \tilde{\mathbf{w}}^t\| &\leq \sum_{k=1}^{|\mathcal{K}|} p_k \|\mathbf{w}_k^t - \tilde{\mathbf{w}}^{t-1}\| \\ &\leq \sum_{k=1}^{|\mathcal{K}|} p_k \|\mathbf{w}_k^t - \mathbf{w}_k^{t-1}\| \\ &\leq \sum_{k=1}^{|\mathcal{K}|} p_k \|\eta \nabla F_k(\mathbf{w}^{t-1})\| \\ &\leq \eta \sum_{k=1}^{|\mathcal{K}|} p_k G_k, \end{aligned} \tag{4.18}$$

where the last inequality results from Assumption 3.

Finally, Theorem 3 is proved by substituting (4.18) and (4.15) into (4.10).

4.6 Summary

In this chapter, we have presented our design of `FedPNS` algorithm, a probabilistic node selection strategy that can preferentially select nodes to boost model convergence of FL with non-i.i.d. datasets. `FedPNS` adjusts the probability for each node to be selected in each round based on the result of the proposed `Optimal Aggregation` algorithm, which is able to find

out the optimal subset of local updates from participating nodes and excludes the adverse local updates for a better model aggregation, by measuring the relationship between the local gradient and the global gradient from participating nodes. The convergence rate improvement of the `FedPNS` design over `FedAvg` is analyzed theoretically. Finally, experimental results on different tasks, models, and datasets have shown that FL training with `FedPNS` accelerates model convergence and leads to higher test accuracy, as compared to `FedAvg`.

Chapter 5

Mitigating Heterogeneous Computation with Partial Model Training

5.1 Overview

In this chapter, we try to answer the following questions: *given limited computation on local devices, which part of the training model should be updated or protected in FL? Moreover, how does the server generate and assign sub-models to computation-heterogeneous devices?*

Existing partial model training designs [23, 30, 75–77] encounter covariant shift problems because the partial model is constructed by statically downsampling in each layer. As a result, different sub-models can only be trained on specific devices that match the resource constraint, updating different parts of the global model with different data distributions. This drawback would degrade the training performance, especially in data heterogeneous FL scenarios. Since the expectation of the output feature in the partial model differs from that in the full model, one must add batch normalization layers or manually scale the output

feature. Authors in [85] proposed to handle the problem with a rolling window where different sub-models (in each layer) can be updated more evenly. Even though sub-model generation is updated in each global round, multiple local updates during consecutive rounds might cause skewness in model learning. More importantly, all the works mentioned above discard a specific ratio of *weight* (the connection between neurons in layers) in every layer to generate a sub-model, which might not be necessary for model training in FL since features in shallow layers are less important and can be shared among devices while unique features of devices are revealed by keeping (at least) the classifier updated.

As such, we propose a new method, `Fed`erated `P`artial `M`odel `T`raining (`FedPMT`), to generate sub-models in a layerwise way for computationally heterogeneous devices to reduce the FL completion time. Different from existing PMT-based methods, which generate sub-models by preserving a subset of neurons in each layer [30, 31, 75–77, 84, 85, 94, 95], `FedPMT` constructs sub-models from the back-propagation (BP) perspective. For resource-constrained devices, the computation burden is reduced by restricting gradient information from back-propagating to the shallow layers. Meanwhile, the most important layers (deep layers) are updated by back-propagation, and the local information (from each participant with unique data samples) is preserved in the partial model training process. We list out related works and a detailed comparison of the proposed design and existing model-homogeneous and model-heterogeneous FL methods in Table 5.1.

To the best of our knowledge, this is the first work considering layerwise model update to handle the system heterogeneity problem in FL[10]. The main advantage of our proposed `FedPMT` is that all participants prioritize the most crucial parts of the global model (i.e., deep layers) and ensure that local training achieves the purpose of data augmentation, as pursued in FL. Meanwhile, avoiding removing neurons in deep layers guarantees a relatively large

[10]It is worth mentioning that a similar concept, sparsified BP [96], where only a small amount of parameters are updated in BP, is applied to reduce the over-fitting problem. Our work is orthogonal to [96] from both objective and implementation perspectives.

Table 5.1: Comparisons of model-homogeneous and model-heterogeneous FL design in the existing literature and the proposed `FedPMT`

FL Methods		Model Heterogeneity	Aggregation Scheme	Comp. / Comm. Heterogeneity	Sub-model Generation / Auxiliary Data
Convergence Optimization	`FedAvg` [5]	No	-	- / No	- / No
	SCAFFOLD [29]				
Knowledge Transfer	FedDF [80]	Yes	Knowledge Distillation	-	- / Unlabeled
	COMET [81]				- / Unlabeled
	FedGKT [82]				- / No
	FedGen [83]				- / No (Generator)
Model Prunning	FedMP [94]	Yes	-	✓ / ✓	Random / No
	PruneFL [95]			✓ / ✓	Static / No
Partial Model Training	Federated Dropout [31]	Yes	Sub-model Training	✓ / ✓	Random / No
	HeteroFL [30]			✓ / -	Fixed / No
	SlimFL [23]			- / ✓	Fixed / No
	FjORD [84]			✓ / ✓	Ordered / No
	FedRolex [85]			✓ / ✓	Rotated / No
	`FedPMT` (Ours Approach)			✓ / ✓	**Layerwise** / No

model capacity. Our main contributions in this work are as follows.

- We identify the prospect of model-heterogeneous FL and propose a layerwise partial model training strategy, `FedPMT`, for resource-constrained FL systems. The proposed `FedPMT` accommodates heterogeneous computation over the FL system by counteracting back-propagation cost when updating the model, a.k.a. layerwise. Without invoking further local computation overhead, `FedPMT` is an easy-to-implement framework fully compatible with existing FL systems or secure aggregation protocols for privacy enhancement.

- We analyze the convergence property of the proposed design. `FedPMT` converges to the global optimum at a rate of $\mathcal{O}(1/T)$ for strongly convex and smooth function in data heterogeneous scenarios, which is similar to the `FedAvg` cases with no resource

constraints. However, given heterogeneous computational capabilities on devices, `FedPMT` has a shorter task completion time.

- We empirically evaluate the performance of `FedPMT` via extensive experiments using the synthetic dataset and real datasets with different learning objectives. By analyzing the computation for various heterogeneous settings, our results demonstrate that the proposed design outperforms model-homogeneous (`FedAvg`) and model-heterogeneous (FedDrop) benchmarks regarding task completion time and training accuracy, respectively.

Our proposed layerwise partial model training strategy inaugurates a new direction of handling computation heterogeneity in FL due to its effectiveness, simplicity, and scalability.

The rest of this chapter is organized as follows. Section 5.2 presents a computation model in FL. Section 5.3 explains how to split the learning model and achieve the layer-wise partial model training in FL. Section 5.4 analyzes the convergence property of the proposed partial model training design and discusses the insights gained from the theoretical result. Experimental results and related complete proof are presented in Section 5.5 and Section 5.6. Finally, Section 5.7 summarizes this chapter.

5.2 System Heterogeneity and Computation Model

Several works have demonstrated the effectiveness of `FedAvg` from both empirical and theoretical perspectives in various settings [20, 29, 97]. One needs to notice that in the system heterogeneous FL, the assumption that every participating device can timely train the designated model and/or transmit the updated model back to the server may not always hold true. For example, the network identities in heterogeneous networks can be Internet of Thing (IoT) devices, PCs, and mobile devices, which have different computational and/or

communication capabilities. Devices may not be able to train a large model due to their energy consumption on this task, or their CPU cycles are too small to finish the task on time, causing a long delay or straggler effect [84, 98]. Therefore, it is not trivial to design an FL system from the time consumption perspective and consider the system heterogeneity. In what follows, we introduce a computation model in general FL.

We denote the number of CPU cycles for device k to execute one sample of data by c_k, which is considered a priori information and can be measured offline. Suppose that all samples $z_{k,s} \in \mathcal{D}_k$ have the same size (e.g., the number of pixels in images), the number of CPU cycles required by device k for each time of local training (i.e., one global round) is $c_k \cdot |\mathcal{D}_k| \cdot E$, where E is the number of local training epoch. Furthermore, the computation time for each global round is derived as $T^t_{cmp} = \frac{c_k \cdot |\mathcal{D}_k| \cdot E}{\kappa_k}$, where κ_k is the CPU cycle frequency of device k, which is fixed for one device and varies for different devices. In this work, the system heterogeneity is reflected by κ_k. This is because c_k is a constant given a training model $\mathbf{w}$, so devices with a higher value of κ_k signify a larger computational capacity, enabling them to complete the local training process faster. In a typical FL design, all devices are rehearsed with the same number of SGD steps (i.e., τ). Therefore, devices with small computational capability would spend a long time to finish the local training, resulting in the straggler effect [84, 98]. In this work, we do not consider the convergence improvement by assigning adaptive τ, which is determined by $|\mathcal{D}_k|$, E, or SGD batch-size as in [21, 99], but focus on delivering different partial models to different participants to mitigate the impact of system heterogeneity. The time consumption for participating devices in each global round t is bounded as $\max\{\frac{c_k \cdot |\mathcal{D}_k| \cdot E}{\kappa_k}\}, k \in \mathcal{S}$.

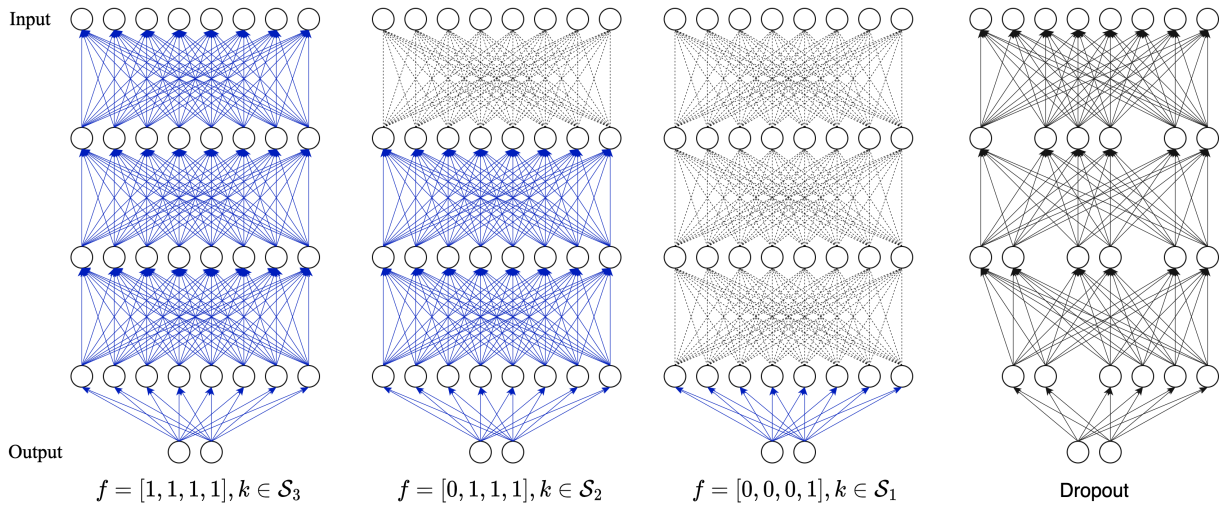

Figure 5.1: Illustration of different local training models of the proposed `FedPMT` for four layers of fully connected neural networks (i.e., $|\mathcal{L}| = 4$) with model width $\mathcal{I} = \{1, 2, 3\}$, where $|\mathcal{I}| = 3 < |\mathcal{L}| = 4$. The leftmost plot represents the model with full model width $|\mathcal{I}| = 3$. The partial model training process with mask $\Xi_i, i \in \mathcal{I}$ is shown by dotted lines and arrow lines in blue, i.e., weights with arrow lines in blue are updated using BP. The weights with dotted lines are not updated by BP, where only the forward process is involved. Function $f_k, k \in \mathcal{S}_i, i \in \mathcal{I}$ is given to represent Ξ_i using $\Upsilon_l, l \in \mathcal{L}$. In comparison, FedDrop removes neurons in hidden layers with probabilities (to accommodate different computational capabilities on devices) at random. For example, the model after dropout (with a dropout rate of 0.25) is shown in the rightmost plot.

5.3 Layerwise-based Partial Model Training

We consider an FL scenario where participating devices have heterogeneous computing capabilities. We adopt the concept of partial model training to accommodate the contribution of devices with heterogeneous computing capabilities. To better compromise the system heterogeneity, the server provides a variety of computing options reflected by *model width* $\mathcal{I} = \{1, 2, \cdots, |\mathcal{I}|\}$ to those devices for motivating participation and improving the global model convergence. Specifically, at any global round $t = 1, 2, \cdots, T$, each participating device can choose one of the model widths provided in $\mathcal{I}$ for its local model training process according to its computing capability.

Without loss of generality, in the t-th global round, we use $|\mathcal{I}|$ to indicate the full model width, use $\Xi_i, i \in \{1, 2, \cdots, |\mathcal{I}|\}$ as the *mask* to generate the local model in order to do partial training and use non-joint sets $\mathcal{S}_i, i \in \{1, 2, \cdots, |\mathcal{I}|\}$ to represent the corresponding sets (with the same model width) that devices belong to. As such, device $k \in \cup\mathcal{S}_i, i \in \mathcal{I} \setminus \{|\mathcal{I}|\}$ can generate a partial model (based on its computing capability) as $\mathbf{w}_k^t = \mathbf{w}^t \odot \Xi_i$ for further processing, and it is clear that $\mathbf{w}_k^t = \mathbf{w}^t \odot \Xi_{|\mathcal{I}|} = \mathbf{w}^t$ holds for devices with full model width, i.e., $k \in \mathcal{S}_{|\mathcal{I}|}$. We define $\odot$ operated on model $\mathbf{w}^t$ to represent the partial model generation process, which is illustrated as

$$\mathbf{w}_k^t = \begin{cases} \mathbf{w}^t \odot \Xi_i & k \in \mathcal{S}_i, i \in \mathcal{I} \setminus \{|\mathcal{I}|\} \\ \mathbf{w}^t \odot \Xi_i = \mathbf{w}^t & k \in \mathcal{S}_i, i = |\mathcal{I}| \end{cases}. \tag{5.1}$$

In our proposed `FedPMT`, we achieve partial model training from the perspective of Back-Propagation (BP)[11]. Particularly, all participating devices $k \in \cup\mathcal{S}_i, i \in \mathcal{I}$ share the same forward process, which calculates the loss function given current model $\mathbf{w}^t$ and its data samples $z_k \in \mathcal{D}_k$. Differently, devices without full model width, i.e., $k \in \cup\mathcal{S}_i, i \in \mathcal{I} \setminus |\mathcal{I}|$ *will not* update all the parameters by BP, instead only the parts where BP is involved. This is achieved by restricting gradient information from back-propagating to the shallow layers. We introduce $\Upsilon_k^l, l \in \mathcal{L} = \{1, 2, \cdots, |\mathcal{L}|\}$ to indicate whether the l-th layer of the learning model on device k is involved in the BP process, where $|\mathcal{L}|$ is the total number of layers in the model. Therefore, for each device $k \in \mathcal{S}_i$ with mask Ξ_i, a relationship between Ξ_i and Υ_k^l is generated $\Xi_i = f_k(\sum_{l \in \mathcal{L}} \Upsilon_k^l), i \in \mathcal{I}$ to represent the involved layers in BP, where Υ_k^l is

[11] Our proposed scheme is different from existing works[75–77], which split the training model into different sub-models with overlap [75, 76] or without overlap [77]. However, these works mentioned above split the network from the neurons' perspective (by only including partial parameters of each layer of the training model, as seen in Fig. 5.1.). Our work splits the training model from the layers' perspective. Partial model training means that devices with model width $i \in \mathcal{I} \setminus \{|\mathcal{I}|\}$ exclusively update part of layers of this model, from the back to the front. As with the traditional FL design, all the participating devices update the classifier, i.e., the last layer of the model, which is helpful to alleviate the classification bias that is identified as the culprit of FL with heterogeneous data [100].

a vector with binary values. The l-th element of Υ_k^l is 1, indicating that the l-th layer of device k is involved in BP; otherwise, all elements in Υ_k^l are 0. $f_k(\sum_{l\in\mathcal{L}} \Upsilon_k^l)$ is regarded as a mapping function with binary coefficients that shows which layer's gradient is updated[12].

In what follows, notation $f_k(\sum_{l\in\mathcal{L}} \Upsilon_k^l)$ is simplified as f_k. Similar to the vanilla federated optimization [5], each device minimizes its empirical risk as shown in (2.2) by running τ steps of (mini-batch) SGD to update local parameters initialized as $\mathbf{w}^t$. For device k, the local model training is formally expressed as

$$\mathbf{w}_k^{t+1} = \mathbf{w}^t - \eta_t \underbrace{\nabla F_k(\mathbf{w}^t, \xi_k) \circ f_k}_{\nabla \tilde{F}_k(\mathbf{w}_k^t, \xi_k)}, \tag{5.3}$$

where η_t is the learning rate, ξ_k is the mini-batch samples, and $\nabla \tilde{F}_k(\mathbf{w}_k^t, \xi_k)$ is the actual gradient for model update in device k, which might be the partial or full gradient depending on the binary values of f_k. We use $\circ$ to represent the *layer-wise multiplication* between a vector a of length $|\mathcal{L}|$ and gradient vector b with $|\mathcal{L}|$ blocks/layers, Below shows a general example for layer-wise multiplication: $\mathrm{a} = [0, 1, 2], \mathrm{b} = [2, 2, 2; 1, 1; 3, 3, 3, 3]$. In the vector b, the semicolon ';' serves as a delimiter to distinguish between model parameters across different layers. We have $\mathrm{a} \circ \mathrm{b} = [0,\ 0,\ 0;\ 1,\ 1;\ 6,\ 6,\ 6,\ 6]$. Note that the length of vector a equals the number of layers in gradient vector b. As such, by introducing the mask, devices with model width $i \in \mathcal{I} \setminus \{|\mathcal{I}|\}$ will not update the model weight of front layers, thus alleviating the computational burden (e.g., for partial derivative and matrix multiplication).

After the local training, the server collects the model updates $\nabla \tilde{F}_k(\mathbf{w}_k^t, \xi_k), k \in \mathcal{S}$ to

[12]Suppose that a three-layer model $\mathbf{w}$ is divided into three different widths, i.e., $\mathcal{I} = \{1, 2, 3\}$. With $i = 3$ being the full model, i.e., $\mathbf{w} \odot \Xi_3 = \mathbf{w}$, devices within set $\mathcal{S}_3$ will update all layers using the BP process. In this case, f_k is written as $[1, 1, 1]$ and $\Upsilon_k^1 = [1, 0, 0], \Upsilon_k^2 = [0, 1, 0], \Upsilon_k^3 = [0, 0, 1]$ (we remove the subscription k for generalization). Similarly, for those devices $k \in \mathcal{S}_1$ with model width Ξ_1, f_k is viewed as $[0, 0, 1]$ ($\Upsilon_k^1 = [0, 0, 0], \Upsilon_k^2 = [0, 0, 0], \Upsilon_k^3 = [0, 0, 1]$), which means that only the last layer is involved in the BP process, and the first two layers will not be updated by BP. In more general cases where $|\mathcal{I}|$ is less than the total number of model layers (e.g., $|\mathcal{I}| = 3 \leq |\mathcal{L}| = 4$), f_k can be generated similarly, e.g., $f_k = [0, 1, 1, 1]$ and $f_k = [1, 1, 1, 1]$. Refer to Fig. 5.1 for a detailed illustration.

$$
\begin{aligned}
\mathbf{w}^{t+1} &\overset{(a)}{=} \mathbf{w}^t - \eta_t \sum_{k\in\mathcal{S}} A_k \circ \nabla\tilde{F}_k(\mathbf{w}_k^t, \xi_k) \\
&\overset{(b)}{=} \mathbf{w}^t - \eta_t \left(\frac{1}{|\mathcal{S}_{|\mathcal{I}|}|} \sum_{k\in\mathcal{S}_{|\mathcal{I}|}} \nabla F_k(\mathbf{w}^t, \xi_k) \circ \Upsilon_k^1 + \frac{1}{|\mathcal{S}_{|\mathcal{I}|} \cup \mathcal{S}_{|\mathcal{I}|-1}|} \sum_{k\in\mathcal{S}_{i|\mathcal{I}|}\cup\mathcal{S}_{|\mathcal{I}|-1}} \nabla F_k(\mathbf{w}^t, \xi_k) \circ \Upsilon_k^2 \right. \\
&\quad \left. + \cdots + \frac{1}{|\cup \mathcal{S}_{i,i\in\mathcal{I}}|} \sum_{k\in\cup\mathcal{S}_i, i\in\mathcal{I}} \nabla F_k(\mathbf{w}^t, \xi_k) \circ \Upsilon_k^{|\mathcal{I}|} \right) \\
&= \mathbf{w}^t - \eta_t \underbrace{\sum_{i\in\mathcal{I}} \frac{1}{|\cup \mathcal{S}_{j,j=i,\cdots,|\mathcal{I}|}|} \sum_{k\in\cup\mathcal{S}_j} \nabla F_k(\mathbf{w}^t, \xi_k) \circ \Upsilon_k^{|\mathcal{I}|-i+1}}_{\nabla F(\mathbf{w}^t)}, \qquad (5.4)
\end{aligned}
$$

cast the global model, as shown in (5.4), where $A_k = [a_1, a_2, \cdots, a_{|\mathcal{L}|}]$ and $a_l = \frac{f_k[l]}{\sum_{k\in\mathcal{S}} f_k[l]}$ for $l = 1, 2, \cdots, |\mathcal{L}|$ and $k \in \mathcal{S}$, where $f_k[l]$ is the l-th element of f_k. For simplicity of representation, we consider the size of local datasets on local devices to be the same. Equation (5.4) gives two different ways to represent global model aggregation, i.e., from the device's perspective (equation (a)) or the layer-wise perspective (equation (b)). As shown in (a), different from weighing local models with a single scalar [5, 14], a weighting vector A_k whose values correspond to the specific layer-wise weight in aggregation is allocated to local models since devices may provide a partially updated model[13]. For example, $A_k[2]$ indicates the weight for aggregating the model parameters of the 2nd layer of device k. $\circ$ represents the layer-wise multiplication calculation between weighting vector A_k and gradient $\nabla\tilde{F}_k(\mathbf{w}_k^t, \xi_k)$.

Procedures of the proposed `FedPMT` algorithm are summarized in Algorithm 4. Two options are provided, as seen in Algorithm 4, where the mask for partial model training is generated either by the server or by local devices. If we choose option I, where the server generates the mask, then each device should report its computation level to the server, similar to [45]. Otherwise, option II can be adopted, in which each device determines mask f that

[13]The values in A_k indicate that devices can provide partial gradient/model for aggregation, and parameters of the rest parts of the partial model are not counted in aggregation because no gradient update is done for those parameters.

Algorithm 4 Federated Learning with Partial Model Training

Input: device set $\mathcal{K}$, step size η_t, model initialization $\mathbf{w}^1$, number of global round T, number of local steps τ, κ if Option I is chosen
for $t = 1, \ldots, T$ **do**
  Server: $\mathcal{S}_t \leftarrow$ random subset of $\mathcal{K}$
  Option I (server initiates model splitting):
    send $\mathbf{w}^t$ and f_k to device $k \in \mathcal{S}_i$
  Option II (device initiates model splitting):
    send $\mathbf{w}^t$ and different masks $\Xi_i, i \in \mathcal{I}$ to device $k \in \mathcal{S}$
  for local device $k \in \mathcal{S}_t$ in parallel **do**
    Option I (server initiates model splitting):
      $\mathsf{LocalUpdate}(\mathbf{w}^t, f_k, \eta_t, \tau)$
    Option II (device generates partial model):
      choosing appropriate $f_k = \Xi_i$ based on computational capability
  end for
  Server: model aggregated by (5.4)
end for
return
$\mathsf{LocalUpdate}(\mathbf{w}^t, f_k, \eta_t, \tau)$ at the k-th device
for $step = 1, \ldots, \tau$ **do**
  (mini-batch) stochastic gradient descent by (5.3)
end for
return $\nabla \tilde{F}_k(\mathbf{w}_k^t, \xi_k)$

matches its computation level. In this case, only marginal extra information f is added to the uplink model transmission.

5.4 Convergence Analysis

In this section, we analyze the convergence property of the proposed partial model training with the local objective satisfying the strongly convex and smooth assumptions and compare it to the convergence rate of `FedAvg`. For the ease of theoretical analysis, we consider $|\mathcal{I}| = |\mathcal{L}|$ and scenarios with $|\mathcal{I}| \leq |\mathcal{L}|$ are verified in Section 5.5.3. To facilitate the convergence analysis, we introduce assumption 6 on SGD optimization, which is commonly adopted in the literature [20, 22, 23]. Then, we give the derived lemmas and theorem for the convergence

rate of `FedPMT`.

Assumption 6. ***Bounded local gradient variance.*** *The variance of local gradient* $\nabla F_k(\mathbf{w}_k^t, \xi_k)$ *is bounded,*
i.e., $\mathbb{E}\left[\|\nabla F_k(\mathbf{w}_k^t, \xi_k) - \nabla F_k(\mathbf{w}_k^t)\|^2\right] \leq \delta_k^2$*, with* $\nabla F_k(\mathbf{w}_k^t)$ *denoting the ground-truth gradient over device* k *given* $\mathcal{D}_k$*.*

Similar to [21], we define $\delta^2 = \frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}} \delta_k^2$ to measure the overall data heterogeneity of all devices in federated optimization.

Proposition 1. *Given local loss satisfying* μ*-strong convexity, the following inequality can be derived, i.e.,* $\langle \mathbf{w}_k^t - \mathbf{w}^*, A_k \circ F_k(\mathbf{w}_k^t)\rangle \geq \frac{\varepsilon}{|\mathcal{S}|} \cdot \left(F_k(\mathbf{w}_k^t) - F_k(\mathbf{w}^*) + \frac{\mu}{2}\|\mathbf{w}_k^t - \mathbf{w}^*\|^2\right)$ *for devices* $k \in \mathcal{S}$ *in* ***`FedPMT`****, where* $\varepsilon \in [0, 1]$ *indicates the information loss due to the partial model update.*

Proof. Given the loss function satisfying Assumption 1, $\langle \mathbf{w}_k^t - \mathbf{w}^*, \nabla F_k(\mathbf{w}_k^t)\rangle \geq F_k(\mathbf{w}_k^t) - F_k(\mathbf{w}^*) + \frac{\mu}{2}\|\mathbf{w}_k^t - \mathbf{w}^*\|^2$, Proposition 1 is derived based on the fact that all parts (reflected by each layer of gradient $\nabla F_k(\mathbf{w}_k^t)$) of the model contribute to the local objective minimization (i.e., the right-hand side of above inequality), and removing part of the model information results in the slowness of the minimization process.

We assume the *most* information decrement on devices with partial model update is measured by $1-\varepsilon$, and whose bound, i.e., $\left(F_k(\mathbf{w}_k^t) - F_k(\mathbf{w}^*) + \frac{\mu}{2}\|\mathbf{w}_k^t - \mathbf{w}^*\|^2\right)$ is lowered by a constant factor ε in such cases. The rationality behind the reduced bound related to the constant factor ε reveals that these devices can retain at least the following amount of information $\varepsilon \cdot \left(F_k(\mathbf{w}_k^t) - F_k(\mathbf{w}^*) + \frac{\mu}{2}\|\mathbf{w}_k^t - \mathbf{w}^*\|^2\right)$, though they update the model with the least effort due to the computation constraint. Since devices $k \in \mathcal{S}_1$ only update the last layer of the model, thus lose the most information regarding its local objective minimization process. Consequently, with weighting vector $A_k, k \in \mathcal{S}_1$ for aggregation being $[0, \cdots, \frac{1}{|\mathcal{S}|}]$, the following inequality is

achieved, i.e., $\langle \mathbf{w}_k^t - \mathbf{w}^*, A_k \circ \nabla F_k(\mathbf{w}_k^t) \rangle \geq \frac{\varepsilon}{|\mathcal{S}|} \left(F_k(\mathbf{w}_k^t) - F_k(\mathbf{w}^*) + \frac{\mu}{2} \|\mathbf{w}_k^t - \mathbf{w}^*\|^2 \right), \forall k \in \mathcal{S}_1$. For all the other devices $k \in \cup \mathcal{S}_{i,i=2,\cdots,|\mathcal{I}|}$ that update more layers in partial model training and thus can retain more information, the above inequality is fulfilled. Therefore, the inequality is achieved for all devices in the partial model training scheme, i.e., $\langle \mathbf{w}_k^t - \mathbf{w}^*, A_k \circ \nabla F_k(\mathbf{w}_k^t) \rangle \geq \frac{\varepsilon}{|\mathcal{S}|} \left(F_k(\mathbf{w}_k^t) - F_k(\mathbf{w}^*) + \frac{\mu}{2} \|\mathbf{w}_k^t - \mathbf{w}^*\|^2 \right), \forall k \in \mathcal{S}$. □

Lemma 2. ***(Bounded variance for global gradient)***. *From Assumption 6, the variance of global gradient is bounded as* $\mathbb{E}\left[\|\nabla F(\mathbf{w}^t) - \nabla \bar{F}(\mathbf{w}^t)\|^2\right] \leq 2|\mathcal{I}|\delta^2\psi$, *where* $\nabla F(\mathbf{w}^t)$ *and* $\nabla \bar{F}(\mathbf{w}^t)$ *represent the global gradient surrogated by* $\nabla F_k(\mathbf{w}_k^t, \xi_k)$ *and* $\nabla F_k(\mathbf{w}_k^t)$, *respectively, according to the aggregation method (5.4),* $\psi = \sum_{i \in \mathcal{I}} \frac{1}{\sum p_{j,j=i,\cdots,|\mathcal{I}|}}$ *and* p_j *in the denominator is defined as the ratio between the number of devices in* $\mathcal{S}_j$ *and the number of participated devices in a global round, i.e.,* $\frac{|\mathcal{S}_j|}{|\mathcal{S}|}$.

Lemma 3. ***(One round convergence)***. *Under Assumptions 1, 2, 3, 6 and Proposition 1, the divergence between the global model at the* $(t+1)$*-th global round and the optimal model satisfies* $\mathbb{E}\left[\|\mathbf{w}^{t+1} - \mathbf{w}^*\|^2\right] \leq (1 - \eta_t \mu \varepsilon) \mathbb{E}\left[\|\mathbf{w}^t - \mathbf{w}^*\|^2\right] + \eta_t^2 (8(\tau-1)^2 G^2 + 2L\eta_t^2 (|\mathcal{I}|\psi + |\mathcal{S}| + \varepsilon)\Lambda + 2\delta^2\psi)$, *where* L, μ, δ^2, τ, G, ε *are defined earlier and* $\Lambda = \frac{1}{|\mathcal{S}|} \sum_{k \in \mathcal{S}} (F^* - F_k^*)$ *measures the degree of non-i.i.d. in federated optimization.* F^* *and* F_k^* *denote the optima of the global loss and local loss of device* k, *respectively.*

We direct readers to Section 5.6.1 and 5.6.2 for the detailed proof Lemmas 2 and 3, respectively. Based on Lemmas 2 and 3, the convergence rate of the proposed `FedPMT` is shown in the following Theorem 5, which is proven in Section 5.6.3.

Theorem 5. *Let Assumptions 1 - 3 and 6 hold and let* $L, \mu, \delta_k, \tau, G$ *be as defined above. Choose the step size* $\eta_t = \frac{2}{\mu\varepsilon(t+\lambda)}$, *the convergence of federated learning with partial model training satisfies*

$$\mathbb{E}\left[F(\mathbf{w}^T) - F(\mathbf{w}^*)\right] \leq \frac{1}{T+\lambda} \left(\frac{(\lambda+1)\Gamma_1}{2} + \frac{2\tilde{\Delta}}{\mu^2} \right), \tag{5.5}$$

where $\lambda > 0$, $\tilde{\Delta} = (8(\tau-1)^2G^2 + 2L(|\mathcal{I}|\psi + |\mathcal{S}| + \varepsilon)\Lambda + 2\delta^2\psi)/\varepsilon^2$, *and* $\Gamma_1 = \mathbb{E}\left[\|\mathbf{w}^1 - \mathbf{w}^*\|^2\right]$ *denotes the distance between the initial and optimal global models.*

From Theorem 1, we observe that `FedPMT` has a convergence rate of $\mathcal{O}(1/T)$, which aligns with the convergence rate of `FedAvg` in [19, 101](refer to Section 5.5.3 for empirical verification). The difference between `FedPMT` and `FedAvg` lies in problem-related constant $\tilde{\Delta}$, essentially caused by information loss in partial model training. In addition, the bound in the right-hand side of (5.5) is related to model splitting (i.e., ψ), as analyzed in the following.

1) Given the initial global model $\mathbf{w}^1$, we have $\Gamma_1 = \|\mathbf{w}^1 - \mathbf{w}^*\|^2 \leq \frac{4}{\mu^2}G^2$ derived for a μ-strongly convex global objective F [101]. Therefore, as shown in (5.5), the dominating term is $\mathcal{O}\left(\frac{(L(|\mathcal{I}|\psi+|\mathcal{S}|+\varepsilon)\Lambda+\delta^2\psi+\lambda G^2+\tau^2G^2)/\varepsilon^2}{T\mu^2}\right)$, compared to the term $\mathcal{O}\left(\frac{L\Lambda+\delta^2+\lambda G^2+\tau^2G^2}{T\mu^2}\right)$ in `FedAvg`. The results reveal that the loss gap between the global model $\mathbf{w}^T$ and optimal model $\mathbf{w}^*$ in `FedPMT` is more significant. This is because only a subset of participating devices update the whole model in local computation. Devices that update the partial model will lose information and contribute less to the global objective minimization.

2) The loss gap in (5.5) is also related to the way to split the model, which determines how much the devices with partial model training can contribute to the global objective minimization. Notably, in order to shrink the loss gap between `FedPMT` and `FedAvg`, one needs to reduce ψ, i.e., $\sum_{i\in\mathcal{I}} \frac{|\mathcal{I}||\mathcal{S}|}{\sum p_{j,j=i,\cdots,|\mathcal{I}|}}$, by enlarging the denominator of ψ. This demonstrates that the gap can be reduced with more devices updating more layers (i.e., a larger $\sum p_{j,j=i,\cdots,|\mathcal{I}|}$). On the contrary, if we assume devices $k \in \mathcal{S} \setminus \mathcal{S}_{|\mathcal{I}|}$ have the computational capability to do large computational tasks but they choose to do small tasks (e.g., updating the last layer of the model), this type of model splitting results in a smaller $\sum p_{j,j=i,\cdots,|\mathcal{I}|}$ and hence a larger gap. This analysis indicates that the partial model design should fully excavate the computation of the local devices in order to expedite the FL process.

Even though FedPMT ends with a larger loss gap, with proper partial models being allocated to resource-constrained devices, FedPMT achieves a better trade-off in terms of completion time in FL.

5.5 Numerical Results

In this section, we implement FedPMT across various tasks with different learning models and compare it with existing benchmarks FedAvg [5] and a Dropout-based partial model training design, FedDrop [31]. In particular, we use a Fully Connected Neural Network (FCNN)[14] and Convolutional Neural Network[15] (CNN) for MNIST and CIFAR-10 tasks, respectively. In the following Section 5.5.1, we briefly describe the computational complexity analysis of the model learning process, including Forward Propagation (FP) and BP. Section 5.5.2 describes the experiment setup. In Section 5.5.3, under the same computation setup, we first compare FedPMT with FedDrop [31] on MNIST dataset in terms of learning accuracy. Then, we compare FedPMT with FedAvg on the CIFAR10 task regarding the training time for given target accuracies.

5.5.1 Computational Complexity Analysis

We consider the model in floating-point format (i.e., 32 bits for each parameter), and the operations in algorithms are floating-point operations. Following the similar analysis in [15, Section IV-E], and supposing n^x training samples in the calculation, we present the following

[14]FCNN model for MNIST task: 784×400 Fully connected (Fc1) $\rightarrow$ 400×300 Fully connected (Fc2) $\rightarrow$ 300×200 Fully connected (Fc3) $\rightarrow$ 200×100 Fully connected (Fc4) $\rightarrow$ 100×10 Fully connected $\rightarrow$ Softmax. All Fully connected layers are mapped by ReLu activation.

[15]CNN for MNIST task is constructed as below: $5 \times 5 \times 8$ Convolutional $\rightarrow$ 2×2 MaxPool $\rightarrow$ $5 \times 5 \times 16$ Convolutional $\rightarrow$ 2×2 MaxPool $\rightarrow$ 256×128 Fully connected $\rightarrow$ 128×10 Fully connected $\rightarrow$ Softmax.

CNN model for CIFAR10 task: $5 \times 5 \times 16$ Convolutional (Conv1) $\rightarrow$ 2×2 MaxPool $\rightarrow$ $5 \times 5 \times 32$ Convolutional (Conv2) $\rightarrow$ 2×2 MaxPool $\rightarrow$ 800×500 Fully connected (Fc1) $\rightarrow$ 500×300 Fully connected (Fc2) $\rightarrow$ 300×10 Fully connected $\rightarrow$ Softmax.

All Fully connected layers are mapped by ReLu activation.

complexity analysis.

FP for FCNN:

- The complexity of propagating from the input layer to the 2nd layer is represented as $O_{2,x} = W_{2,1}Z_{1,x}$, which has a complexity of $\mathcal{O}(n_2 \times n_1 \times n^x)$, where Z, W, O represent input, weight parameter, and output of one layer, respectively. The subscript $\{2, 1\}$ denotes the transition process between layers hereinafter, and n_j is the number of neurons of the $j-$th layer.
- The activation function $Z_{2,x} = \bar{f}_{ac}(O_{2,x})$ has a complexity of $\mathcal{O}(n_2 \times n^x)$.
- The rest of the layers follow a similar analysis of the above steps.

BP for FCNN:

For output layer (i.e., o) to the 4th hidden layer (Fc4), we

- Compute the error signal $e_{\{o,x\}}$ at the output layer as $e_{o,x} = \bar{f}'_{ac}(S_{o,x}) \circledast (Z_{o,x} - y_{o,x})$, where $Z_{o,x}$ is the raw output signal of the last layer, $\bar{f}'_{ac}$ is the inverse activation function, $y_{o,x}$ is the data label, and $\circledast$ represents element-wise multiplication.
- Compute the gradient $D_{o,4} = e_{o,x} \times Z_{x,4}$, where $Z_{x,4}$ is the transpose of $Z_{4,x}$.
- Update the weight on the 4th layer $W_{o,4} = W_{o,4} - \eta_t D_{o,4}$.

The complexity of the above operations is $\mathcal{O}(n_o \times n^x + n_o \times n^x + n_o \times n^x \times n_4 + n_o \times n_4)$.

For the 4th hidden layer (Fc4) to 3rd hidden layer (Fc3), we have $e_{4,x} = \bar{f}'_{ac}(S_{4,x}) \circledast (W_{4,x} - e_{o,x})$, then $D_{4,3} = e_{4,x} \times Z_{x,3}$ and $W_{4,3} = W_{4,3} - \eta_t D_{4,3}$, where $W_{4,3}$ is the transpose of $W_{3,4}$. The complexity is $\mathcal{O}(n_4 \times n^x + n_4 \times n_o \times n^x + n_4 \times n^x \times n_3 + n_4 \times n_3)$.

The BP complexity of the rest of the layers of FCNN can be derived by a similar analogy.

FP for CNN:

The complexity of convolutional layers is found in [102], which is $\mathcal{O}(n_{l-1} \times s_l^2 \times n_l \times m_l^2)$,

where l is the index of convolutional layer, n_l indicates the number of filters in the l-th layer (n_{l-1} is also known as the number of input channels in the l-th layer), s_l is the spatial size of the filter, and m_l is the spatial size of the output feature map, which is calculated as $m_l = (s_x - s_l + 2 \times padding)/stride + 1)$ and s_x is the size of input.

- Conv1: $n_0 = 3, n_1 = 16, s_1 = 5, m_1 = (32 - 5 + 2 \times 0)/1 + 1 = 28$. Then, using the max-pooling layer, the output feature size is $14 \times 14 \times 16$.
- Conv2: $n_1 = 16, n_2 = 32, s_2 = 5, m_2 = (14 - 5 + 2 \times 0)/1 + 1 = 10$. Then, using the max-pooling layer, the output feature size is $5 \times 5 \times 32$.

BP for CNN: From [102], the complexity of the BP process for convolutional layers is roughly twice that of the FP process.

The FP and BP in the fully connected layer in CNN are the same as the cases in FCNN, as discussed above.

The detailed computation is quantitively shown in Table 5.2, where several training models with different model widths are provided. For example, $|\mathcal{I}| = 4$ means four training model widths are available for the server (or devices) to choose. In Table I, *FP+BP (Full)* represents devices with the full model, and *Full - Fc1 (BP)* represents devices that *do not* update the Fc1 layer. *Full - Fc1 (BP) -Fc2 (BP)* represents devices that *do not* update the Fc1 and Fc2 layers, and so on and so forth. The computational complexity of models with different model widths can be calculated according to the above discussion. Meanwhile, to make a fair comparison, we set FedDrop [31] with the same computational complexity as `FedPMT`.

5.5.2 Experiment Setup

Data heterogeneity: Two different data distribution settings are discussed, namely i.i.d. and non-i.i.d. settings. For the i.i.d. setting, data samples on each device are randomly

Table 5.2: Experiment setup to compare `FedPMT` and FedDrop. We set the same computational complexity (or keep a higher computation capability for FedDrop in cases when exact same complexity cannot be made) for `FedPMT` and FedDrop on each device to compare. *rate* in FedDrop indicates that in order to keep the same computational complexity as `FedPMT`, FedDrop needs to keep a percent of hidden layers' neurons, compared with the full model.

Computational complexity of FCNN-MNIST (local epoch $E = 1$, batch size is 12)		
Model Width ($\|\mathcal{I}\| = 4$)	`FedPMT` (ratio)	FedDrop (dropout rate) [31]
Full - Fc1 (BP) - Fc2 (BP) - Fc3 (BP)	6473760 (42.3%)	6431556 ($\approx$ 54%)
Full - Fc1 (BP) - Fc2 (BP)	7496160 (48.98%)	7579990 ($\approx$ 61%)
Full - Fc1 (BP)	9779760 (63.9%)	9717454 ($\approx$ 73%)
FP+BP (Full)	15305968 (100%)	100%
Model Width ($\|\mathcal{I}\| = 2$)		
Full - Fc1 (BP)	9779760 (63.9%)	9717454 ($\approx$ 73%)
FP+BP (Full)	15305968 (100%)	100%
Computational complexity of CNN-MNIST (local epoch $E = 1$, batch size is 12)		
Model Width ($\|\mathcal{I}\| = 4$)	PMT (ratio)	FedDrop (dropout rate)
Full - Conv1(BP) - Conv2(BP) - Fc1 (BP)	745456 (40.8%)	1047049 (cap = 0.1)
Full - Conv1(BP) - Conv2(BP)	1188336 (65%)	1185944 ($\approx$ 0.26)
Full - Conv1(BP)	1597936 (87.4%)	1593950 ($\approx$ 0.73)
FP+BP (Full)	1828336 (100%)	1
Model Width ($\|\mathcal{I}\| = 2$)		
Full - Conv1(BP)	1597936 (87.4%)	1593950 ($\approx$ 0.73)
FP+BP (Full)	1828336 (100%)	1
Computational complexity of CNN-CIFAR10 (local epoch $E = 1$, batch size is 20)		
Model Width ($\|\mathcal{I}\| = 5$)	`FedPMT` (ratio)	FedDrop (dropout rate) [31]
Full - Conv1(BP) - Conv2(BP) - Fc1 (BP) - Fc2 (BP)	12864200 (45.83%)	12885200 ($\approx$ 40%)
Full - Conv1(BP) - Conv2(BP) - Fc1 (BP)	16077200 (57.27%)	16031980 ($\approx$ 54%)
Full - Conv1(BP) - Conv2(BP)	24587200 (87.59%)	24677840 ($\approx$ 88%)
Full - Conv1(BP)	26187200 (93.29%)	26069215 ($\approx$ 93%)
FP+BP (Full)	28068800 (100%)	100%
Model Width ($\|\mathcal{I}\| = 2$)	`FedPMT` (ratio)	FedDrop
Full - Conv1(BP)	26187200 (93.29%)	26069215 ($\approx$ 93%)
FP+BP (Full)	28068800 (100%)	100%

selected from the training dataset. In the non-i.i.d. setting, data samples on each device belong to 2 different classes, which are randomly selected from 10 classes. The data samples on different devices form disjoint sets. We generate each setting and keep it fixed for different experiments to avoid randomness brought by training samples. Each experiment is executed with 10 random trails with fixed seeds in Pytorch.

For each experiment, a set of devices $\mathcal{S}$ is randomly selected in each global round from a set of candidate devices $\mathcal{K}$ with $|\mathcal{K}| = 100$. To better capture the impact of heterogeneous

computation on FL learning performance, we assume that the number of selected devices with the same computation capabilities is evenly distributed among $|\mathcal{S}|$, e.g., $|\mathcal{S}_i| = 10/|\mathcal{I}|, i \in \mathcal{I}$, in CIFAR10 experiments. The training setup is as follows,

MNIST: $|\mathcal{D}_k| = 300, \eta_t = 0.01, E = 1$, $|\mathcal{S}| = 8$.

CIFAR10: $|\mathcal{D}_k| = 500, \eta_t = 0.05, E = 1$, $|\mathcal{S}| = 10$.

FL Training time (CNN-CIFAR10 task)

κ setting: Since `FedPMT` targets reducing the training time for computation heterogeneous FL, we set five different computation levels, $0.2\Psi, 0.25\Psi, 0.33\Psi, 0.5\Psi, 1\Psi$, where 1Ψ represents the maximum computation capability for a set of participating devices. For example, suppose a device with 1Ψ can complete the local training in 10 seconds (i.e., $T_{cmp} = \frac{c_k \cdot |\mathcal{D}_k| \cdot E}{\kappa_k} = 10$), devices with 0.2Ψ takes 50 seconds to finish the same task.

c setting: The computation time in `FedPMT` is analyzed as follows: Since the devices with model width smaller than full model width only need to update part of the whole model, which makes c_k smaller. For the case with five different model widths $|\mathcal{I}| = 5$ (see Table 5.2), the training models with the complexity ratio 0.46, 0.58, 0.88, 0.94, and 1 will be assigned to devices with $0.2\Psi, 0.25\Psi, 0.33\Psi, 0.5\Psi$, and 1Ψ, respectively. Therefore, the training time consumption is $0.46 \times 50s, 0.58 \times 40s, 0.88 \times 30s, 0.94 \times 20s$, and $1 \times 10s$, respectively (assuming 1Ψ can complete the local training of a full model in 10 seconds).

5.5.3 Empirical Results

We first compared `FedPMT` with a dropout-based algorithm, FedDrop [31], in scenarios where different participating devices have different computational levels, reflected by different $|\mathcal{I}|$. Given the above setup, `FedPMT` generates different partial models for computation-heterogeneous devices, where devices with small computation capacity will restrict gradient from back-propagating to more shallow layers. While FedDrop[31] creates different par-

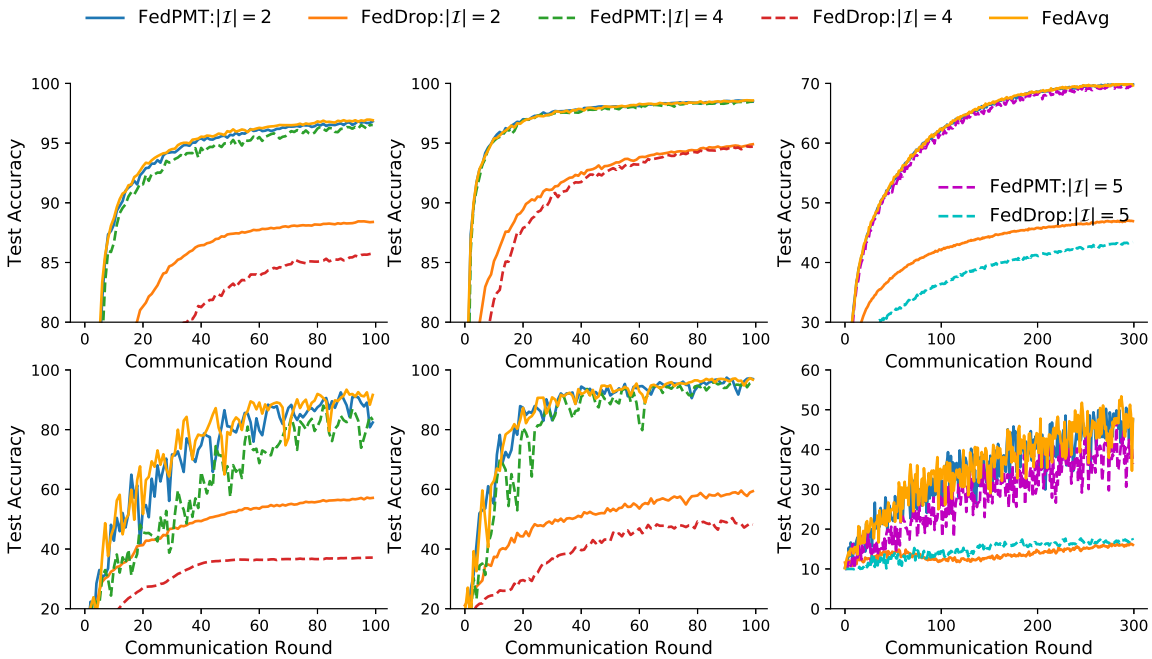

Figure 5.2: Test accuracy over communication rounds of `FedPMT` and FedDrop with different data heterogeneity and different computation levels in FL. From left to right, each column corresponds to the learning result on FCNN-MNIST, CNN-MNIST, and CNN-CIFAR10 tasks, respectively. The upper and lower plots show the learning results for the i.i.d. and non-i.i.d. scenarios, respectively.

tial models by removing varying numbers of neurons in hidden layers to match devices' computation capabilities.

`FedPMT` outperforms FedDrop across different computation heterogeneity and data distribution settings. Both `FedPMT` and FedDrop perform better in the cases with model width $|\mathcal{I}| = 2$, compared to cases with model width $|\mathcal{I}| = 4$. The FL global model converges faster since devices' computation capabilities are generally higher in $|\mathcal{I}| = 2$. With limited computation power on devices, FedDrop randomly removes neurons in hidden layers, making

model capacity small. While devices in FedPMT sacrifice shallow layers and prioritize the most crucial layers, thus ensuring a better performance than FedDrop. Those non-prioritized layers can still be updated in the model aggregation. The inaccuracy in shallow layers impacts model performance less than that in deep layers (as seen in FedDrop). This observation is more evident with non-i.i.d. data. FedPMT with model width $|\mathcal{I}| = 4$ achieves more than 90% accuracy, while FedDrop barely works with an accuracy lower than 60%. This is because data samples share common features in the non-i.i.d. case, and each local classifier (the last layer) is more sensitive to different data distributions. Given limited computation power, we need to prioritize the crucial layers (near classifier) instead of evenly reducing the number of neurons in hidden layers as done in FedDrop. In addition, regardless of the learning completion time, FedAvg's learning result is provided as an upper bound for different tasks. FedAvg assumes homogeneous models across local devices and does not consider devices' heterogeneous computation capabilities. As can be seen in Fig. 5.2, FedPMT with smaller model widths (e.g., $|\mathcal{I}| = 2$) achieves similar learning results as FedAvg for all i.i.d. and non-i.i.d. cases. Among CNN model-related tasks, FedPMT achieves very competitive results even for more computation heterogeneous scenarios ($|\mathcal{I}| = 4$ or 5), although fluctuations in the learning process are observed in non-i.i.d. scenarios, leaving the performance margin to FedAvg negligible, compared to extra computational complexity in FedAvg. For the FCNN-MNIST task with more heterogeneous computation, there is a larger performance gap between FedPMT and FedAvg. However, the proposed design still outperforms FedDrop with a prominent performance gap.

Next, we compare FedPMT and FedAvg on completion time in FL. We consider two different cases: 1) with a constraint (26.5 seconds)[16], the computation time constraint is set in each global round. Beyond this time stamp, the server aggregates the received models (without

[16]This constraint is set as the model training time spent by the devices with the longest completion time in FedPMT, as calculated in 5.5.1.

Table 5.3: Learning time comparison between different FL designs

Accuracy	constraint (26.5 seconds) FedPMT	constraint (26.5 seconds) FedAvg	without constraint FedPMT	without constraint FedAvg
	i.i.d.			
50%	1064.8	1121.8	1029.6	1950
55%	1584	1732	1601.6	2800
60%	2270.4	2464.5	2217.6	4266.6
	non-i.i.d.			
40%	4136	4507	4174	4644
45%	6157	6604.4	6218.7	6577.7
50%	8251	8771.5	9234.1	10361

waiting for the rest) and moves to the next global round. 2) *without constraint* means the server aggregates models after receiving all local models in each global round.

With a time constraint in each round, more devices in `FedPMT` can contribute to the global model aggregation, even though models from devices with limited computation capabilities are not completely updated. `FedPMT` is more effective in the non-i.i.d. case, where aggregating more local models results in faster convergence, as also observed in [5, 21]. If no constraint is set in each global round, `FedPMT` can complete the learning task in almost half the time, compared to `FedAvg` (2217.6 seconds vs. 4266.6 seconds at 60% accuracy). Although `FedAvg` obtains more accurate local models in each round, it is inefficient in terms of completion time. In contrast, `FedPMT` achieves a better trade-off between model accuracy and completion time.

5.6 Complete Proof

5.6.1 Proof of Lemma 2

For the ease of analysis, the gradient $\nabla F_k(\mathbf{w}_k^t, \xi_k)$ and $\nabla F_k(\mathbf{w}_k^t)$ are represented by $g_{k,i}^t$ and $\bar{g}_{k,i}^t$ in the following proof, where the subscript i in $g_{k,i}^t$ indicates that the device k belongs to

set $\mathcal{S}_i$.

From the definition of $\nabla F(\mathbf{w}^t)$, we have

$$
\begin{aligned}
&\|\nabla F(\mathbf{w}^t) - \nabla \bar{F}(\mathbf{w}^t)\|^2 \\
&= \|\sum_{i\in\mathcal{I}} \frac{1}{|\cup \mathcal{S}_{j,j=i,\cdots,|\mathcal{I}|}|} \sum_{k\in\cup\mathcal{S}_j} (g_{k,i}^t - \bar{g}_{k,i}^t) \circ \Upsilon_k^{|\mathcal{I}|-i+1}\|^2 \\
&\overset{1}{\leq} |\mathcal{I}| \cdot \sum_{i\in\mathcal{I}} \frac{1}{|\cup \mathcal{S}_{j,j=i,\cdots,|\mathcal{I}|}|} \|\sum_{k\in\cup\mathcal{S}_j} (g_{k,i}^t - \bar{g}_{k,i}^t) \circ \Upsilon_k^{|\mathcal{I}|-i+1}\|^2 \\
&\overset{2}{\leq} \sum_{i\in\mathcal{I}} \frac{|\mathcal{I}||\mathcal{S}|}{|\cup \mathcal{S}_{j,j=i,\cdots,|\mathcal{I}|}|} \sum_{k\in\cup\mathcal{S}_j} \|(g_{k,i}^t - \bar{g}_{k,i}^t) \circ \Upsilon_k^{|\mathcal{I}|-i+1}\|^2 \\
&\overset{3}{\leq} \sum_{i\in\mathcal{I}} \frac{|\mathcal{I}||\mathcal{S}|}{|\cup \mathcal{S}_{j,j=i,\cdots,|\mathcal{I}|}|} \sum_{k\in\cup\mathcal{S}_j} \|g_{k,i}^t - \bar{g}_{k,i}^t\|^2,
\end{aligned} \tag{5.9}
$$

where inequality 1 holds by Cauchy-Schwartz inequality, inequality 2 holds by Cauchy-Schwartz inequality and $\cup\mathcal{S}_{j,j=i,\cdots,|\mathcal{I}|} \subset \mathcal{S}, \forall i \in \mathcal{I}$, and inequality 3 holds because the norm of partial gradient is smaller than the norm of full gradient, i.e., $\|(g_{k,i}^t - \bar{g}_{k,i}^t) \circ \Upsilon_k^{|\mathcal{I}|-i+1}\|^2 < \|(g_{k,i}^t - \bar{g}_{k,i}^t)\|^2$ for all model width $i \in \mathcal{I}$.

Taking the expectation on both sides of (5.9), we have

$$
\begin{aligned}
&\mathbb{E}\left[\|\nabla F(\mathbf{w}^t) - \nabla \bar{F}(\mathbf{w}^t)\|^2\right] \\
\leq &\mathbb{E}\left[\sum_{i\in\mathcal{I}} \frac{|\mathcal{I}||\mathcal{S}|}{|\cup \mathcal{S}_{j,j=i,\cdots,|\mathcal{I}|}|} \sum_{k\in\cup\mathcal{S}_j} \|g_{k,i}^t - \bar{g}_{k,i}^t\|^2\right] \\
= &\sum_{i\in\mathcal{I}} \frac{|\mathcal{I}||\mathcal{S}|}{|\cup \mathcal{S}_{j,j=i,\cdots,|\mathcal{I}|}|} \sum_{k\in\cup\mathcal{S}_j} \mathbb{E}\left[\|g_{k,i}^t - \bar{g}_{k,i}^t\|^2\right] \\
\overset{4}{\leq} &\sum_{i\in\mathcal{I}} \frac{|\mathcal{I}||\mathcal{S}|}{|\cup \mathcal{S}_{j,j=i,\cdots,|\mathcal{I}|}|} \sum_{k\in\mathcal{S}} \mathbb{E}\left[\|g_{k,i}^t - \bar{g}_{k,i}^t\|^2\right] \\
\overset{5}{=} &\sum_{i\in\mathcal{I}} \frac{|\mathcal{I}||\mathcal{S}|}{|\mathcal{S}| \cdot \sum_{j=i}^{|\mathcal{I}|} p_j} \sum_{k\in\mathcal{S}} \mathbb{E}\left[\|g_{k,i}^t - \bar{g}_{k,i}^t\|^2\right] \\
= &\frac{1}{|\mathcal{S}|} \sum_{i\in\mathcal{I}} \frac{|\mathcal{I}||\mathcal{S}|}{\sum_{j=i}^{|\mathcal{I}|} p_j} \sum_{k\in\mathcal{S}} \delta_k^2 \\
\leq &2\delta^2 \sum_{i\in\mathcal{I}} \frac{|\mathcal{I}||\mathcal{S}|}{\sum_{j=i}^{|\mathcal{I}|} p_j},
\end{aligned}
\tag{5.11}
$$

where inequality 4 holds by $\cup\mathcal{S}_{j,j=i,\cdots,|\mathcal{I}|} \subset \mathcal{S}, \forall i \in \mathcal{I}$, and p_j in the denominator in equality 5 is defined as the ratio between the number of devices in $\mathcal{S}_j$ and the number of participated devices in a global round, i.e., $\frac{|\mathcal{S}_j|}{|\mathcal{S}|}$.

5.6.2 Proof of Lemma 3

By the definition of $\mathbb{E}[\nabla F(\mathbf{w}^t)] = \nabla \bar{F}(\mathbf{w}^t)$, we have

$$
\begin{aligned}
&\|\mathbf{w}^{t+1} - \mathbf{w}^*\|^2 \\
= &\|\mathbf{w}^t - \eta_t \nabla F(\mathbf{w}^t) - \mathbf{w}^* - \eta_t \nabla \bar{F}(\mathbf{w}^t) + \eta_t \nabla \bar{F}(\mathbf{w}^t)\|^2 \\
= &\underbrace{\|\mathbf{w}^t - \eta_t \nabla \bar{F}(\mathbf{w}^t) - \mathbf{w}^*\|^2}_{\mathfrak{C}_1} + \eta_t^2 \underbrace{\|\nabla F(\mathbf{w}^t) - \nabla \bar{F}(\mathbf{w}^t)\|^2}_{\mathfrak{C}_2} + \underbrace{2\eta_t \langle \mathbf{w}^t - \eta_t \nabla \bar{F}(\mathbf{w}^t) - \mathbf{w}^*, \nabla \bar{F}(\mathbf{w}^t) - \nabla F(\mathbf{w}^t)\rangle}_{\mathfrak{C}_3} \\
= &\|\mathbf{w}^t - \mathbf{w}^*\|^2 + \eta_t^2 \|\nabla \bar{F}(\mathbf{w}^t)\|^2 - 2\eta_t \langle \mathbf{w}^t - \mathbf{w}^*, \nabla \bar{F}(\mathbf{w}^t)\rangle + \eta_t^2 \mathfrak{C}_2 + \mathfrak{C}_3.
\end{aligned}
\tag{5.12}
$$

• Bounding term $\eta_t^2\|\nabla\bar{F}(\mathbf{w}^t)\|^2$

By the definition of $\nabla\bar{F}(\mathbf{w}^t)$, we have

$$\begin{aligned}\|\nabla\bar{F}(\mathbf{w}^t)\|^2 &= \|\sum_{i\in\mathcal{I}}\frac{1}{|\cup\mathcal{S}_{j,j=i,\cdots,|\mathcal{I}|}|}\sum_{k\in\cup\mathcal{S}_j}\nabla F_k(\mathbf{w}_k^t)\circ\Upsilon_k^{|\mathcal{I}|-i+1}\|^2\\ &\overset{1,2,3}{\leq}\sum_{i\in\mathcal{I}}\frac{|\mathcal{I}||\mathcal{S}|}{|\cup\mathcal{S}_{j,j=i,\cdots,|\mathcal{I}|}|}\sum_{k\in\cup\mathcal{S}_j}\|\nabla F_k(\mathbf{w}_k^t)\|^2\\ &\overset{4,5}{\leq}\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{I}}\frac{|\mathcal{I}||\mathcal{S}|}{\sum_{j=i}^{|\mathcal{I}|}p_j}\sum_{k\in\mathcal{S}}\|\nabla F_k(\mathbf{w}_k^t)\|^2. \end{aligned}\tag{5.13}$$

$\|\nabla F_k(\mathbf{w}_k^t)\|^2$ in equation (5.13) is bounded as follows. Given any models $\mathbf{w}_k^t$ and $\mathbf{w}_k'$ satisfying Assumption 2, we have $F_k(\mathbf{w}_k')-F_k(\mathbf{w}_k^t)-(\mathbf{w}_k'-\mathbf{w}_k^t)^\top\nabla F_k(\mathbf{w}_k^t)\leq\frac{L}{2}\|\mathbf{w}_k'-\mathbf{w}_k^t\|^2$. By defining $\mathbf{w}_k'=\mathbf{w}_k^t-\frac{1}{L}\nabla F_k(\mathbf{w}_k^t)$, we have $F_k(\mathbf{w}_k')-F_k(\mathbf{w}_k^t)\leq-\frac{1}{L}(\nabla F_k(\mathbf{w}_k^t))^\top\nabla F_k(\mathbf{w}_k^t)+\frac{L}{2}\frac{1}{L^2}\|\nabla F_k(\mathbf{w}_k^t)\|^2\leq-\frac{1}{2L}\|\nabla F_k(\mathbf{w}_k^t)\|^2$. Taking the minimal loss F_k^* on device k, we have

$$\|\nabla F_k(\mathbf{w}_k^t)\|^2\leq 2L(F_k(\mathbf{w}_k^t)-F_k(\mathbf{w}_k'))\leq 2L(F_k(\mathbf{w}_k^t)-F_k^*).\tag{5.14}$$

We use ψ to denote the constant $\sum_{i\in\mathcal{I}}\frac{|\mathcal{I}||\mathcal{S}|}{\sum_{j=i}^{|\mathcal{I}|}p_j}$ hereinafter. As such, $\eta_t^2\|\nabla\bar{F}(\mathbf{w}^t)\|^2$ is bounded by combing (5.14) and (5.13), and we have

$$\eta_t^2\|\nabla\bar{F}(\mathbf{w}^t)\|^2\leq 2|\mathcal{I}|L\eta_t^2\psi\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}_k^t)-F_k^*).\tag{5.15}$$

• Bounding term $-2\eta_t\langle\mathbf{w}^t-\mathbf{w}^*,\nabla\bar{F}(\mathbf{w}^t)\rangle$.

Again, by the definition of $\nabla \bar{F}(\mathbf{w}^t)$ and (5.4), we have

$$
\begin{aligned}
&-2\eta_t \langle \mathbf{w}^t - \mathbf{w}^*, \nabla \bar{F}(\mathbf{w}^t) \rangle \\
=& -2\eta_t \langle \mathbf{w}^t - \mathbf{w}^*, \sum_{k\in\mathcal{S}} A_k \circ \nabla F_k(\mathbf{w}_k^t) \rangle \\
=& \underbrace{-2\eta_t \sum_{k\in\mathcal{S}} \langle \mathbf{w}^t - \mathbf{w}_k^t, \nabla \tilde{\tilde{F}}_k(\mathbf{w}_k^t) \rangle}_{\mathfrak{C}_{4.1}} \underbrace{-2\eta_t \sum_{k\in\mathcal{S}} \langle \mathbf{w}_k^t - \mathbf{w}^*, \nabla \tilde{\tilde{F}}_k(\mathbf{w}_k^t) \rangle}_{\mathfrak{C}_{4.2}} .
\end{aligned} \tag{5.16}
$$

where $\nabla \tilde{\tilde{F}}_k(\mathbf{w}_k^t) = A_k \circ \nabla F_k(\mathbf{w}_k^t)$ is the result of local ground-truth gradient after layer-wise multiplication with weight A_k.

Each term in $\mathfrak{C}_{4.1}$ is bounded as follows: By Cauchy-Schwarz inequality, AM-GM inequality, we have the first inequality hold in (5.17). The last inequality in (5.17) is achieved since $\|\nabla \tilde{\tilde{F}}_k(\mathbf{w}_k^t)\|^2 = \|A_k \circ \nabla F_k(\mathbf{w}_k^t)\|^2 < \|\nabla F_k(\mathbf{w}_k^t)\|^2$.

$$
\begin{aligned}
&-2\eta_t \langle \mathbf{w}^t - \mathbf{w}_k^t, \nabla \tilde{\tilde{F}}_k(\mathbf{w}_k^t) \rangle \\
\leq& \eta_t (\frac{1}{\eta_t} \|\mathbf{w}^t - \mathbf{w}_k^t\|^2 + \eta_t \|\nabla \tilde{\tilde{F}}_k(\mathbf{w}_k^t)\|^2) \\
\leq& \|\mathbf{w}^t - \mathbf{w}_k^t\|^2 + \eta_t^2 \|\nabla F_k(\mathbf{w}_k^t)\|^2.
\end{aligned} \tag{5.17}
$$

By Assumption 1 and Proposition 1, each term in $\mathfrak{C}_{4.2}$ is bounded as

$$
\begin{aligned}
&-2\eta_t \langle \mathbf{w}_k^t - \mathbf{w}^*, \nabla \tilde{\tilde{F}}_k(\mathbf{w}_k^t) \rangle \\
\leq& 2\eta_t \varepsilon \frac{1}{|\mathcal{S}|} \left((-(F_k(\mathbf{w}_k^t) - F_k(\mathbf{w}^*)) - \frac{\mu}{2} \|\mathbf{w}_k^t - \mathbf{w}^*\|^2) \right).
\end{aligned} \tag{5.18}
$$

Based on the above intermediate results, $-2\eta_t\langle \mathbf{w}^t - \mathbf{w}^*, \nabla \bar{F}(\mathbf{w}^t)\rangle$ is bounded as

$$
\begin{aligned}
&-2\eta_t\langle \mathbf{w}^t - \mathbf{w}^*, \nabla \bar{F}(\mathbf{w}^t)\rangle \\
\leq &\sum_{k\in\mathcal{S}}(\|\mathbf{w}^t - \mathbf{w}_k^t\|^2 + \eta_t^2\frac{|\mathcal{S}|}{|\mathcal{S}|}\|\nabla F_k(\mathbf{w}_k^t)\|^2 - 2\eta_t\varepsilon\frac{1}{|\mathcal{S}|}(F_k(\mathbf{w}_k^t) - F_k(\mathbf{w}^*)) - \mu\eta_t\varepsilon\frac{1}{|\mathcal{S}|}\|\mathbf{w}_k^t - \mathbf{w}^*\|^2) \\
= &-\mu\eta_t\varepsilon\|\mathbf{w}^t - \mathbf{w}^*\|^2 + \sum_{k\in\mathcal{S}}(\|\mathbf{w}^t - \mathbf{w}_k^t\|^2 + \eta_t^2\frac{|\mathcal{S}|}{|\mathcal{S}|}\|\nabla F_k(\mathbf{w}_k^t)\|^2 - 2\eta_t\varepsilon\frac{1}{|\mathcal{S}|}(F_k(\mathbf{w}_k^t) - F_k(\mathbf{w}^*)).
\end{aligned}
\tag{5.19}
$$

Inserting (5.14), (5.15), and (5.19) to (5.12), we have

$$
\begin{aligned}
\|\mathbf{w}^{t+1} - \mathbf{w}^*\|^2 \leq &(1-\eta_t\mu)\|\mathbf{w}^t - \mathbf{w}^*\|^2 + \sum_{k\in\mathcal{S}}\|\mathbf{w}^t - \mathbf{w}_k^t\|^2 + \eta_t^2\mathfrak{C}_1 + \mathfrak{C}_2 \\
&+ \underbrace{(2|\mathcal{I}|L\eta_t^2\psi + 2L\eta_t^2|\mathcal{S}|)\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}_k^t) - F_k^*) - 2\eta_t\varepsilon\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}_k^t) - F_k(\mathbf{w}^*))}_{\mathfrak{C}_5}.
\end{aligned}
\tag{5.20}
$$

Defining $\gamma_t = 2\eta_t(\varepsilon - \eta_t L(|\mathcal{I}|\psi + |\mathcal{S}|))$. In addition, we have $\eta_t \leq \frac{\varepsilon}{2L(|\mathcal{I}|\psi+|\mathcal{S}|}$) and $\eta_t\varepsilon \leq \gamma_t \leq 2\eta_t\varepsilon$. $\mathfrak{C}_5$ is transformed as

$$
\begin{aligned}
&\mathfrak{C}_5 \\
= &-\gamma_t\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}_k^t) - F_k^*) + 2\eta_t\varepsilon\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}_k^t) - F_k^*) - 2\eta_t\varepsilon\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}_k^t) - F_k(\mathbf{w}^*)) \\
= &-\gamma_t\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}_k^t) - F_k^* + F^* - F^*) + 2\eta_t\varepsilon\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}^*) - F_k^*) \\
= &-\gamma_t\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}_k^t) - F^*) + (2\eta_t\varepsilon - \gamma_t)\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F^* - F_k^*) \\
= &-\gamma_t\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}_k^t) - F^*) + 2L\eta_t^2(|\mathcal{I}|\psi + |\mathcal{S}|)\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F^* - F_k^*) \\
= &\underbrace{-\gamma_t\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}_k^t) - F^*)}_{\mathfrak{C}_{5.1}} + 2L\eta_t^2(|\mathcal{I}|\psi + |\mathcal{S}|)\Lambda,
\end{aligned}
\tag{5.21}
$$

where $\Lambda = \frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}} F^* - F_k^*$ measures the degree of non-i.i.d. in federated optimization. F^*, F_k^*, and $F_k(\mathbf{w}^*)$ represent the optional global loss, the optional local loss on device k, and the local loss on device k with optimal model $\mathbf{w}^*$, respectively.

To bound $\mathfrak{C}_{5.1}$, we have

$$\begin{aligned}
\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}_k^t) - F^*) &= \frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}_k^t) - F_k(\mathbf{w}^t)) + \frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}^t) - F^*) \\
&\geq \frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(\langle \nabla F_k(\mathbf{w}^t), \mathbf{w}_k^t - \mathbf{w}^t\rangle + F(\mathbf{w}^t) - F^*) \\
&\overset{6}{\geq} -\frac{1}{2}\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(\eta_t\|\nabla \bar{F}_k(\mathbf{w}^t)\|^2 + \frac{1}{\eta_t}\|\mathbf{w}_k^t - \mathbf{w}^t\|^2 + \frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F(\mathbf{w}^t) - F^*) \\
&\geq -\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}[\eta_t L(F_k(\mathbf{w}^t) - F_k^*) + \frac{1}{2\eta_t}\|\mathbf{w}_k^t - \mathbf{w}^t\|^2 + F(\mathbf{w}^t) - F^*],
\end{aligned} \tag{5.22}$$

where the first inequality results from the convexity of local loss F_k, inequality 6 holds by AM-GM inequality, and the last inequality is achieved by (5.14).

By combing (5.22) and (5.21), $\mathfrak{C}_5$ is bounded as

$$\begin{aligned}
&\mathfrak{C}_5 \\
&\leq \gamma_t\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}[\eta_t L(F_k(\mathbf{w}^t) - F_k^*) + \frac{1}{2\eta_t}\|\mathbf{w}_k^t - \mathbf{w}^t\|^2] - \gamma_t(F(\mathbf{w}^t) - F^*) + 2L\eta_t^2(|\mathcal{I}|\psi + |\mathcal{S}|)\Lambda \\
&= \gamma_t\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}[\eta_t L(F_k(\mathbf{w}^t) - F^* + F^* - F_k^*) + \frac{1}{2\eta_t}\|\mathbf{w}_k^t - \mathbf{w}^t\|^2] - \gamma_t(F(\mathbf{w}^t) - F^*)) + 2L\eta_t^2(|\mathcal{I}|\psi + |\mathcal{S}|)\Lambda \\
&= \gamma_t(\eta_t L - 1)\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}^t) - F^*) + \frac{\gamma_t}{2\eta_t}\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}\|\mathbf{w}_k^t - \mathbf{w}^t\|^2 + [\gamma_t\eta_t L + 2L\eta_t^2(|\mathcal{I}|\psi + |\mathcal{S}|)]\Lambda \\
&\leq \frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}\|\mathbf{w}_k^t - \mathbf{w}^t\|^2 + 2L\eta_t^2(|\mathcal{I}|\psi + |\mathcal{S}| + \varepsilon)\Lambda,
\end{aligned} \tag{5.23}$$

where the last inequality achieves because: 1) We have $\gamma_t > 0$ since $\eta_t\varepsilon \leq \gamma_t \leq 2\eta_t\varepsilon$, and $\eta_t L - 1 = \frac{\varepsilon}{2(|\mathcal{I}|\psi + |\mathcal{S}|)} - 1 \leq 0$, so that $\gamma_t(\eta_t L - 1)\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}(F_k(\mathbf{w}^t) - F^*) \leq 0$. 2) Since $\eta_t\varepsilon \leq \gamma_t \leq 2\eta_t\varepsilon$, we have $\frac{\gamma_t}{2\eta_t}\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}\|\mathbf{w}_k^t - \mathbf{w}^t\|^2 < \frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}\|\mathbf{w}_k^t - \mathbf{w}^t\|^2$ and $[\gamma_t\eta_t L +$

$2L\eta_t^2(|\mathcal{I}|\psi + |\mathcal{S}|)]\Lambda < 2L\eta_t^2(|\mathcal{I}|\psi + |\mathcal{S}| + \varepsilon)\Lambda.$

By replacing term $\mathfrak{C}_5$ of (5.20) with (5.23), taking the expectation on both sides of (5.20), and leveraging the Lemma 2 to represent $\mathbb{E}[\mathfrak{C}_1]$, we have

$$\begin{aligned}
&\mathbb{E}\|\mathbf{w}^{t+1} - \mathbf{w}^*\|^2 \\
&\leq (1 - \eta_t\mu\varepsilon)\mathbb{E}\|\mathbf{w}^t - \mathbf{w}^*\|^2 + \mathbb{E}\sum_{k\in\mathcal{S}}\|\mathbf{w}^t - \mathbf{w}_k^t\|^2 \\
&\quad + \mathbb{E}\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}\|\mathbf{w}_k^t - \mathbf{w}^t\|^2 + \eta_t^2(|\mathcal{I}|\psi + |\mathcal{S}| + \varepsilon)\Lambda + \mathbb{E}[\eta_t^2\mathfrak{C}_1] + \mathbb{E}\mathfrak{C}_2, \qquad (5.24)
\end{aligned}$$

- Bounding term $\mathbb{E}\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}\|\mathbf{w}_k^t - \mathbf{w}^t\|^2$.

Assume that between any two consecutive rounds, there is an aggregated model $\mathbf{w}^{t-1,r}, 1 \leq r \leq \tau$, which is not achieved in reality since aggregation happens only after every τ local steps. It is straightforward that $\mathbf{w}^{t-1,\tau} = \mathbf{w}^t$. The learning rate η is fixed between two consecutive rounds. With that, $\mathbb{E}\sum_{k\in\mathcal{S}}\frac{1}{|\mathcal{S}|}\|\mathbf{w}^t - \mathbf{w}_k^t\|^2$ is bounded as follows

$$\begin{aligned}
\mathbb{E}\sum_{k\in\mathcal{S}}\frac{1}{|\mathcal{S}|}\|\mathbf{w}^t - \mathbf{w}_k^t\|^2 &= \mathbb{E}\sum_{k\in\mathcal{S}}\frac{1}{|\mathcal{S}|}\|(\mathbf{w}_k^t - \mathbf{w}^{t-1,r}) - (\mathbf{w}^t - \mathbf{w}^{t-1,r})\|^2 \\
&\overset{7}{\leq} \mathbb{E}\sum_{k\in\mathcal{S}}\frac{1}{|\mathcal{S}|}\|\mathbf{w}_k^t - \mathbf{w}^{t-1,r}\|^2 \\
&\overset{8}{\leq} \sum_{k\in\mathcal{S}}\frac{1}{|\mathcal{S}|}\mathbb{E}\sum_{j=r}^{\tau}(\tau - r)\eta_{t-1}^2\|\nabla F_k(\mathbf{w}_k^{t-1,j}, \xi_k)\|^2 \\
&\overset{9}{\leq} \sum_{k\in\mathcal{S}}\frac{1}{|\mathcal{S}|}\sum_{j=r}^{\tau}(\tau - r)\eta_{t-1}^2 G^2 \\
&\leq \sum_{k\in\mathcal{S}}\frac{1}{|\mathcal{S}|}(\tau - 1)^2\eta_{t-1}^2 G^2 \\
&\overset{10}{\leq} 4\eta_t^2(\tau - 1)^2 G^2 \qquad (5.25)
\end{aligned}$$

where the inequality 7 is from $\mathbb{E}\|X - \mathbb{E}X\|^2 \leq \mathbb{E}\|X\|^2$ [15] and the inequality 8 is achieved by Jensen inequality $\|\mathbf{w}_k^t - \mathbf{w}^{t-1,r}\|^2 = \|\sum_{j=r}^{\tau}\eta_{t-1}\nabla F_k(\mathbf{w}_k^{t-1,j}, \xi_k)\|^2 \leq (\tau - r)\sum_{j=r}^{\tau}\eta_{t-1}^2\|\nabla F_k(\mathbf{w}_k^{t-1,j}, \xi_k)\|^2$ ∎

Inequality 9 is from Assumption 3, and the inequality 10 holds since $\eta_{t-1} \leq 2\eta_t$.

Analogously, we can bound $\mathbb{E}\sum_{k\in\mathcal{S}} \|\mathbf{w}^t - \mathbf{w}_k^t\|^2 \leq 4\eta_t^2(\tau-1)^2G^2$ in the same way.

By inserting (5.25) to (5.24), we have

$$\begin{aligned}
&\mathbb{E}\|\mathbf{w}^{t+1} - \mathbf{w}^*\|^2 \\
\leq&(1-\eta_t\mu\varepsilon)\mathbb{E}\|\mathbf{w}^t - \mathbf{w}^*\|^2 + 8\eta_t^2(\tau-1)^2G^2 \\
&+ 2L\eta_t^2(|\mathcal{I}|\psi + |\mathcal{S}| + \varepsilon)\Lambda + 2\eta_t^2\delta^2\psi,
\end{aligned} \tag{5.26}$$

where the inequality holds because $\mathbb{E}[\mathfrak{C}_2] = \mathbb{E}[2\eta_t\langle \mathbf{w}^t - \eta_t\nabla\bar{F}(\mathbf{w}^t) - \mathbf{w}^*, \nabla\bar{F}(\mathbf{w}^t) - \nabla F(\mathbf{w}^t)\rangle] = 0$ due to $\mathbb{E}[\nabla F(\mathbf{w}^t)] = \nabla\bar{F}(\mathbf{w}^t)$ and $\mathbb{E}[\mathfrak{C}_1] = 2\eta_t^2\delta^2\psi$ by Lemma 2.

5.6.3 Proof of Theorem 5

From Lemma 3, it follows that $\Gamma_{t+1} \leq (1-\eta_t\mu\varepsilon)\Gamma_t + \eta_t^2\Delta$, where $\Gamma_{t+1} = \mathbb{E}\left[\|\mathbf{w}^{t+1} - \mathbf{w}^*\|^2\right]$, $\Gamma_t = \mathbb{E}\left[\|\mathbf{w}^t - \mathbf{w}^*\|^2\right]$ and $\Delta = 8(\tau-1)^2G^2 + 2L(|\mathcal{I}|\psi + |\mathcal{S}| + \varepsilon)\Lambda + 2\delta^2\psi$.

For a diminishing step size $\eta_t = \frac{\beta}{t+\lambda}$ and for some $\lambda > 0, \beta > \frac{1}{\mu}$ such that $\eta_t \leq \frac{\varepsilon}{2L(|\mathcal{I}|\psi+|\mathcal{S}|)}$ and $\eta_t \leq 2\eta_{t+1}$, we aim to prove $\Gamma_t \leq \frac{\chi}{t+\lambda}$ where $\chi = \max\{\frac{\beta^2\Delta}{\beta\mu\varepsilon} - 1, (\lambda+1)\Gamma_1\}$.

Firstly, the definition of χ ensures that Γ_t holds for $t = 1$. Assume that $\Gamma_t \leq \frac{\chi}{t+\lambda}$ holds for some t, we have

$$\begin{aligned}
\Gamma_{t+1} &\leq (1-\eta_t\mu\varepsilon)\Gamma_t + \eta_t^2\Delta \\
&\leq (1 - \frac{\beta\mu\varepsilon}{t+\lambda})\frac{\chi}{t+\lambda} + \frac{\beta^2\Delta}{(t+\lambda)^2} \\
&= \frac{t+\lambda-1}{(t+\lambda)^2}\chi + [\frac{\beta^2\Delta}{(t+\lambda)^2} - \frac{\beta\mu\varepsilon - 1}{(t+\lambda)^2}\chi] \\
&\leq \frac{\chi}{t+\lambda+1}.
\end{aligned} \tag{5.25}$$

By the definition of χ, we have

$$\chi = \max\{(\lambda+1)\Gamma_1, \frac{\beta^2\Delta}{\beta\mu\varepsilon - 1}\} \leq (\lambda+1)\Gamma_1 + \frac{\beta^2\Delta}{\beta\mu\varepsilon - 1}. \tag{5.26}$$

Then, by choosing $\beta = \frac{2}{\mu\varepsilon}$ ($\eta_t = \frac{2}{\mu\varepsilon(t+\lambda)}$ in the meantime) and L-smoothness property of F, Theorem 5 is proven as

$$\mathbb{E}\left[F(\mathbf{w}^T) - F(\mathbf{w}^*)\right] \leq \frac{L}{2}\Gamma_T \leq \frac{1}{T+\lambda}\left(\frac{(\lambda+1)\Gamma_1}{2} + \frac{2\Delta}{\mu^2\varepsilon^2}\right). \tag{5.27}$$

5.7 Summary

In this chapter, we have presented our model-heterogeneous FL design, `FedPMT`, which enables computation-constrained devices to participate in federated learning and contribute to the global model. As a partial model training strategy, `FedPMT` achieves sub-model training from the backpropagation perspective. Unlike Dropout-based partial model training that randomly removes neurons in hidden layers, `FedPMT` allows all participating devices to prioritize the most crucial parts (deep layers) of the global model, ensuring a relatively large model capacity. We have analyzed the convergence rate of `FedPMT`, which shows a similar convergence property as `FedAvg`, with a slightly larger sub-optimality gap factored with a model splitting-related constant. Our experimental results show that `FedPMT` consistently outperforms the state-of-the-art Dropout-based algorithm, FedDrop. Meanwhile, `FedPMT` reaches the learning target in a shorter completion time and achieves a better trade-off between the learning accuracy and FL training time, compared to the widely adopted model-homogeneous benchmark, `FedAvg`.

Chapter 6

Conclusions and Further Research

In this chapter, we summarize the major research contributions and point out further research directions.

6.1 Summary of the Thesis

This thesis aims at understanding and addressing the challenges of federated learning system design in mobile edge networks. We target the statistical and system heterogeneity of federated networks and build FL systems that fulfill the accuracy, efficiency, and robustness requirements. Specifically, the main contributions of this research are summarized as follows:

- We propose an adaptive weighting strategy for federated learning with statistical heterogeneity to reduce the communication cost and expedite model training. Quantifying local devices' contribution by the relationship between the separately-trained local model and the aggregated global model is fundamentally important for data-heterogeneous FL, which remains unexplored. The main intuition is to measure the contribution of the participating device based on the gradient information, then assign different weights accordingly and adaptively at each communication round for global model aggregation.

With theoretical analysis, we show how weighting impacts the expected training loss decrement of the learning objective. Empirically, we evaluate the learning performance of the proposed algorithm and compare it with the commonly adopted benchmark via extensive experiments. While the algorithmic modification is minor and in a simple full-participation setting, the proposed algorithm untangles the direction to speed up FL training in non-i.i.d. scenarios by quantifying device contributions.

- We introduce a probabilistic device selection to support FL systems in large scale and better balance the exploitation and exploration of candidate devices in federated networks. Random device selection poses learning difficulty in non-i.i.d. datasets where the misalignment between the global objective and local objectives exists. To align local model updates with minimizing the global objective, we develop `Optimal Aggregation` algorithm to determine the optimal subset to aggregate local model updates of participating devices. By excluding the local updates with adverse contribution, the data heterogeneity can be profiled, which will be further used to adjust the probability for each device to be selected in the subsequent global rounds. The proposed algorithm involves minor calculations on the server side, does not impose additional communication costs, and is easy to implement in a scalable fashion.

- We develop a layerwise partial model training strategy to handle the computation heterogeneity of candidate devices in federated networks. The implicit assumption that all devices are capable of doing model training and exchanging model information is unrealistic in building robust FL systems. To accommodate different types of devices with heterogeneous computational capabilities, model-heterogeneous FL is proposed, where participants are allowed to train models with different complexity (i.e., the subset of a learning model). Unlike the existing methods using dropout or pruning based sub-model generation, a novel layer-wise model-splitting method to match the device's

capability is proposed to mitigate the straggler effect. Theoretically, we find that the proposed partial model training strategy achieves a similar convergence rate to FedAvg in strongly convex and smoothness loss functions. However, by allowing adaptive partial model allocation, all participating devices can contribute to the global model punctually, making the task completion time shorter and the FL system robust to straggler effects and training disruption.

6.2 Future Work

6.2.1 Personalized Federated Learning

Federated learning is initiated to learn one global model by leveraging information from distributed edge devices in a privacy-preserving manner. To incentivize devices to participate in the FL process for data augmentation, it is crucial that the resulting global model performs better than the local models trained by the devices. However, this may not always be guaranteed in scenarios with highly heterogeneous data. In addition, some FL applications, such as recommendation systems and personalized advertisements, require customized results for different devices [9], which is another critical factor in FL design. Overall, personalized FL should achieve a better trade-off between the benefit of collaboration with other devices and the negative effect caused by data heterogeneity across different devices' domains.

Existing literature focuses on various perspectives to achieve personalized FL design, including the variant of learning full model personalization (e.g., one global model based on *meta-learning* [103–105], *regularization* methods [106–109]), and learning partial model personalization (e.g., a shared global model with *personalized Layers* [110–112] and using *clustering* based methods [81, 113, 114]). However, many of these algorithms involve additional computation [103, 106, 107] or communication costs [113, 114] to edge devices, making the

superiority of those personalized FL designs questionable. In response to this concern, we have implemented the representative works mentioned above and achieved some preliminary results. Our findings suggest that learning one global model and making local adaptations is generally ineffective, especially for complex tasks. On the other hand, clustering methods generally work well but at the cost of increased computation/communication. Since participating devices are typically resource-constrained, the personalized representation approaches might be good choices for the future, given the negotiable modification of the current FL design and decent learning results.

Further, the following research directions might be of interest: 1) The standardization of evaluating personalized FL is needed, including fair setups for the experiment, proper datasets for comprehensive evaluation, and more criteria to measure the model performance. 2) Methods that can migrate the additional cost of learning the personalized information to the server are promising, as demonstrated in [115]. 3) Exploring personalized FL in heterogeneous models will be more beneficial and practical from the applicability perspective.

6.2.2 Model Diagnosis in Applicable Federated Learning

Most research in the FL context focuses on improving the training performance, leaving the model evaluation step under-explored. To deploy the machine learning models for applications, a model evaluation and feedback mechanism is needed to measure the model's effectiveness, such as MLOps (Machine Learning Operations) [116], which indicates whether the model is satisfactory or requires further improvement.

Considering a classification or object detection task (e.g., autonomous cars scenario), model evaluation is difficult without the label data, making it more complex in data-heterogeneous distributed learning scenarios. Particularly, unlike the model evaluation in centralized machine learning, the FL model targets more unseen devices. Two major challenges are involved:

1) the Out-of-domain (OOD) problem, i.e., the test distribution on new devices is likely different from that of training devices; 2) the training data can not be accessed twice in FL context due to the random device selection and device constraints. These challenges make traditional accuracy estimation without label data inaccessible, including measuring the difference between 1) train and test distributions [117], 2) the inference confidence on train and test dataset [118], or learning multiple ensemble models for prediction [119]. In addition, model evaluation on edge devices needs to adhere to the algorithm budget.

Given those challenges in FL model diagnosis, we sought to find effective methods that work well in the general classification tasks. For instance, the ATC method [120] achieves accuracy estimation by learning a threshold from the confidence score of model inference with a limited amount of labeled data, as empirically observed. Further, it would be interesting to form the connection between the size of observation space from labeled data and the projection from the confidence score of model inference to accuracy estimation. With that, one can achieve the trade-off between the assumption (the amount of labeled sample) and the model evaluation result since applications have varying sensitivity to the model evaluation results. Beyond the threshold-based accuracy estimation, it is crucial to consider performance evaluation in other OOD scenarios, such as objective detection, which includes bounding box positioning and classification.

6.2.3 Robustness in Federated Learning

Robustness to various kinds of heterogeneity encountered in practice, such as data heterogeneity, system heterogeneity, model heterogeneity, and malicious behaviors from training and communication, is required for the development of federated learning and broader context, trustworthy decentralized machine learning. The heterogeneity challenges and untrustworthy behaviors that FL raises are diverse in multiple aspects, including but not limited to,

- *Training data heterogeneity.* It is seen that the heterogeneity among candidate devices in federated networks results in slow and unstable learning. Most of the existing works address the data-invariant data heterogeneity problem. However, these approaches are far from dealing with practical tasks in reality, where local data and data distribution are changing, from both theoretical and applicable design perspectives. There is an early attempt [121] that focuses on time-evolving heterogeneous data, but the theoretical results are far from enough to address real-world challenges.

- *Training system heterogeneity.* To achieve a satisfying learning model with efficiency, the system needs to handle the dynamically changing devices and communication networks. Despite its prevalence and importance, research in this area is limited. It is important to design the FL system from both the training level and coordination level. For example, to deal with the behaviors of disappearing and straggling, one might build up a reputation-based device scheduling mechanism [122], combining with advanced local optimizers and an effective model aggregation scheme, even for model-heterogeneous FL.

- *Malicious behaviors.* Most current FL systems assume participating devices are trusted in training and communication, which might not be true in many real applications. Collaborative learning systems can be manipulated by multiple parties in different steps of learning [123], and the defense process may cause fairness problems among multiple participants [107], e.g., simply filtering out the potentially risky devices that hold data that are simply diverse from the average devices. This leads to an alarming fairness counter-effect with regard to robustness. It would be of interest to consider robust FL designs with guaranteed fairness for candidate devices and the interplay between the constrains, including fairness, robustness, and efficiency.

Bibliography

[1] K. L. Lueth, *State of the IoT 2018: Number of IoT devices now at 7B-Market accelerating*. [Online]. Available: https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/, Aug. 2019.

[2] T. Zhang, J. Gao, T. Uehara, *et al.*, "Testing location-based function services for mobile applications," in *Proc. the IEEE Symposium on Service-Oriented System Engineering (SOSE)*, 2015.

[3] C. H. Liu, X. Ma, X. Gao, and J. Tang, "Distributed energy-efficient multi-uav navigation for long-term communication coverage by deep reinforcement learning," *IEEE Transactions on Mobile Computing*, vol. 19, no. 6, pp. 1274–1285, 2020.

[4] M. Chiang and T. Zhang, "Fog and iot: An overview of research opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, 2016.

[5] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. the Artificial Intelligence and Statistics Conference (AISTATS)*, 2017.

[6] 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.

[7] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato,

and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[9] 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.

[10] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Federated learning for ultra-reliable low-latency v2v communications," in *Proc. the IEEE Global Communications Conference (GLOBECOM)*, 2018.

[11] A. Pantelopoulos and N. G. Bourbakis, "A survey on wearable sensor-based systems for health monitoring and prognosis," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 1, pp. 1–12, 2009.

[12] 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.

[13] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-iid data with reinforcement learning," in *Proc. the IEEE Conference on Computer Communications (INFOCOM)*, 2020.

[14] H. Wu and P. Wang, "Fast-convergent federated learning with adaptive weighting," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 4, pp. 1078–1088, 2021.

[15] H. Wu and P. Wang, "Node selection toward faster convergence for federated learning on non-iid data," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 5, pp. 3099–3111, 2022.

[16] H. Wu and P. Wang, "Fast-convergent federated learning with adaptive weighting," in *Proc. the IEEE International Conference on Communications (ICC), 2021*.

[17] H. Wu and P. Wang, "Probabilistic node selection for federated learning with heterogeneous data in mobile edge," in *Proc. the IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 2453–2458, 2022.

[18] H. Wu, P. Wang, and A. C. Narayan, "Model-heterogeneous federated learning with partial model training," in *Proc. the IEEE/CIC International Conference on Communications in China (ICCC)*, pp. 1–6, 2023.

[19] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," vol. 2, 2020.

[20] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," in *Proc. International Conference on Learning Representations (ICLR)*, 2019.

[21] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[22] S. U. Stich, "Local sgd converges fast and communicates little," in *Proc. International Conference on Learning Representations (ICLR)*, 2019.

[23] H. Baek, W. J. Yun, Y. Kwak, S. Jung, M. Ji, M. Bennis, J. Park, and J. Kim, "Joint superposition coding and training for federated learning over multi-width neural

networks," in *Proc. IEEE Conference on Computer Communications (INFOCOM)*, 2022.

[24] H. T. Nguyen, V. Sehwag, S. Hosseinalipour, C. G. Brinton, M. Chiang, and H. V. Poor, "Fast-convergent federated learning," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 201–218, 2020.

[25] H. Yu, S. Yang, and S. Zhu, "Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning," in *Proc. the AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

[26] 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," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[27] N. H. Tran, W. Bao, A. Zomaya, N. M. NH, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. the IEEE Conference on Computer Communications (INFOCOM)*, 2019.

[28] C. T. Dinh, N. H. Tran, M. N. Nguyen, C. S. Hong, W. Bao, A. Y. Zomaya, and V. Gramoli, "Federated learning over wireless networks: Convergence analysis and resource allocation," *IEEE/ACM Transactions on Networking*, vol. 29, no. 1, pp. 398–409, 2020.

[29] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *Proc. International Conference on Machine Learning (ICML)*, 2020.

[30] E. Diao, J. Ding, and V. Tarokh, "Heterofl: Computation and communication efficient

federated learning for heterogeneous clients," in *Proc. International Conference on Learning Representations (ICLR)*, 2021.

[31] D. Wen, K.-J. Jeon, and K. Huang, "Federated dropout—a simple approach for enabling federated learning on resource constrained devices," *IEEE Wireless Communications Letters*, vol. 11, no. 5, pp. 923–927, 2022.

[32] H. Yu, R. Jin, and S. Yang, "On the linear speedup analysis of communication efficient momentum sgd for distributed non-convex optimization," in *Proc. International Conference on Machine Learning (ICML)*, 2019.

[33] H. Yuan and T. Ma, "Federated accelerated stochastic gradient descent," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[34] W. Liu, L. Chen, Y. Chen, and W. Zhang, "Accelerating federated learning via momentum gradient descent," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 8, pp. 1754–1766, 2020.

[35] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smithy, "Feddane: A federated newton-type method," in *Proc. the 53rd Asilomar Conference on Signals, Systems, and Computers (ACSSC)*, 2019.

[36] D. A. E. Acar, Y. Zhao, R. M. Navarro, M. Mattina, P. N. Whatmough, and V. Saligrama, "Federated learning based on dynamic regularization," in *Proc. the International Conference on Learning Representations (ICLR)*, 2021.

[37] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in *Proc. the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[38] Z. Qu, X. Li, R. Duan, Y. Liu, B. Tang, and Z. Lu, "Generalized federated learning

via sharpness aware minimization," in *Proc. the International Conference on Machine Learning (ICLR)*, 2022.

[39] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur, "Sharpness-aware minimization for efficiently improving generalization," in *Proc. the International Conference on Machine Learning (ICLR)*, 2021.

[40] Y. Chen, X. Sun, and Y. Jin, "Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 10, pp. 4229–4238, 2020.

[41] C.-H. Hu, Z. Chen, and E. G. Larsson, "Scheduling and aggregation design for asynchronous federated learning over wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 4, pp. 874–886, 2023.

[42] Z. Chai, Y. Chen, L. Zhao, Y. Cheng, and H. Rangwala, "Fedat: A communication-efficient federated learning method with asynchronous tiers under non-iid data," *arXiv preprint arXiv:2010.05958*, 2020.

[43] M. Zhang, K. Sapra, S. Fidler, S. Yeung, and J. M. Alvarez, "Personalized federated learning with first order model optimization," in *Proc. the International Conference on Machine Learning (ICLR)*, 2021.

[44] M. Beaussart, F. Grimberg, M.-A. Hartley, and M. Jaggi, "Waffle: Weighted averaging for personalized federated learning," *arXiv preprint arXiv:2110.06978*, 2021.

[45] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. the IEEE International Conference on Communications (ICC)*, 2019.

[46] M. M. Amiri, D. Gündüz, S. R. Kulkarni, and H. V. Poor, "Convergence of update aware device scheduling for federated learning at the wireless edge," *IEEE Transactions on Wireless Communications*, vol. 20, no. 6, pp. 3643–3658, 2021.

[47] M. Chen, N. Shlezinger, H. V. Poor, Y. C. Eldar, and S. Cui, "Communication-efficient federated learning," *Proceedings of the National Academy of Sciences*, vol. 118, no. 17, p. e2024789118, 2021.

[48] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence time optimization for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 4, pp. 2457–2471, 2020.

[49] J. Ren, Y. He, D. Wen, G. Yu, K. Huang, and D. Guo, "Scheduling for cellular federated edge learning with importance and channel awareness," *IEEE Transactions on Wireless Communications*, vol. 19, no. 11, pp. 7690–7703, 2020.

[50] Y. J. Cho, J. Wang, and G. Joshi, "Client selection in federated learning: Convergence analysis and power-of-choice selection strategies," in *Proc. the 25th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2022.

[51] W. Chen, S. Horváth, and P. Richtárik, "Optimal client sampling for federated learning," *Transactions on Machine Learning Research*, 2022.

[52] E. Rizk, S. Vlaski, and A. H. Sayed, "Optimal importance sampling for federated learning," in *Proc. the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

[53] Y. G. Kim and C.-J. Wu, "Autofl: Enabling heterogeneity-aware energy efficient federated learning," in *Proc. the 54th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2021.

[54] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-iid data," in *Proc. the IEEE International Conference on Big Data (Big Data)*, 2020.

[55] S. Zheng, Q. Meng, T. Wang, W. Chen, N. Yu, Z.-M. Ma, and T.-Y. Liu, "Asynchronous stochastic gradient descent with delay compensation," in *Proc. the International Conference on Machine Learning (ICML)*, 2017.

[56] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," *arXiv preprint arXiv:1903.03934*, 2019.

[57] M. R. Sprague, A. Jalalirad, M. Scavuzzo, C. Capota, M. Neun, L. Do, and M. Kopp, "Asynchronous federated learning for geospatial applications," in *Proc. the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2018.

[58] D. Huba, J. Nguyen, K. Malik, R. Zhu, M. Rabbat, A. Yousefpour, C.-J. Wu, H. Zhan, P. Ustinov, H. Srinivas, *et al.*, "Papaya: Practical, private, and scalable federated learning," *Proc. of Machine Learning and Systems (MLSys)*, 2022.

[59] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Transactions on Wireless Communications*, 2020.

[60] M. M. Amiri and D. Gündüz, "Federated learning over wireless fading channels," *IEEE Transactions on Wireless Communications*, 2020.

[61] W.-T. Chang and R. Tandon, "Communication efficient federated learning over multiple access channels," *arXiv preprint arXiv:2001.08737*, 2020.

[62] G. Zhu, Y. Wang, and K. Huang, "Broadband analog aggregation for low-latency federated edge learning," *IEEE Transactions on Wireless Communications*, vol. 19, pp. 491–506, Jan 2020.

[63] T. Sery, N. Shlezinger, K. Cohen, and Y. C. Eldar, "Over-the-air federated learning from heterogeneous data," *IEEE Transactions on Signal Processing*, vol. 69, pp. 3796–3811, 2021.

[64] P. Yang, Y. Jiang, T. Wang, Y. Zhou, Y. Shi, and C. N. Jones, "Over-the-air federated learning via second-order optimization," *IEEE Transactions on Wireless Communications*, vol. 21, no. 12, pp. 10560–10575, 2022.

[65] A. Ghosh, R. K. Maity, and A. Mazumdar, "Distributed newton can communicate less and resist byzantine workers," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[66] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," in *Proc. the International Conference on Learning Representations (ICLR)*, 2016.

[67] S. Caldas, J. Konečny, H. B. McMahan, and A. Talwalkar, "Expanding the reach of federated learning by reducing client resource requirements," *arXiv preprint arXiv:1812.07210*, 2018.

[68] J. Mills, J. Hu, and G. Min, "Communication-efficient federated learning for wireless edge intelligence in iot," *IEEE Internet of Things Journal*, 2019.

[69] Y. Wang, Y. Xu, Q. Shi, and T.-H. Chang, "Quantized federated learning under transmission delay and outage constraints," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 1, pp. 323–341, 2021.

[70] S. Zheng, C. Shen, and X. Chen, "Design and analysis of uplink and downlink communications for federated learning," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 7, pp. 2150–2167, 2021.

[71] J. Wangni, J. Wang, J. Liu, and T. Zhang, "Gradient sparsification for communication-efficient distributed optimization," in *Proc. the Advances in Neural Information Processing Systems*, 2018.

[72] P. Han, S. Wang, and K. K. Leung, "Adaptive gradient sparsification for efficient federated learning: An online learning approach," in *Proc. the International Conference on Distributed Computing Systems (ICDCS)*, 2020.

[73] H. Wang, S. Sievert, S. Liu, Z. Charles, D. Papailiopoulos, and S. Wright, "Atomo: Communication-efficient learning via atomic sparsification," in *Proc. Advances in neural information processing systems (NeurIPS)*, 2018.

[74] Z. Tang, S. Shi, B. Li, and X. Chu, "Gossipfl: A decentralized federated learning framework with sparsified and adaptive communication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 3, pp. 909–922, 2023.

[75] J. Yu, L. Yang, N. Xu, J. Yang, and T. Huang, "Slimmable neural networks," in *Proc. the International Conference on Learning Representations (ICLR)*, 2019.

[76] J. Yu and T. S. Huang, "Universally slimmable networks and improved training techniques," in *Proc. the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[77] B. Yuan, C. R. Wolfe, C. Dun, Y. Tang, A. Kyrillidis, and C. Jermaine, "Distributed learning of fully connected neural networks using independent subnet training," *Proc. VLDB Endowment*, 2022.

[78] A. Mohtashami, M. Jaggi, and S. Stich, "Masked training of neural networks with partial gradients," in *Proc. International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2022.

[79] G. Hinton, O. Vinyals, J. Dean, *et al.*, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, vol. 2, no. 7, 2015.

[80] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[81] Y. J. Cho, J. Wang, T. Chirvolu, and G. Joshi, "Communication-efficient and model-heterogeneous personalized federated learning via clustered knowledge transfer," *IEEE Journal of Selected Topics in Signal Processing*, pp. 1–14, 2023.

[82] C. He, M. Annavaram, and S. Avestimehr, "Group knowledge transfer: Federated learning of large cnns at the edge," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 14068–14080, 2020.

[83] Z. Zhu, J. Hong, and J. Zhou, "Data-free knowledge distillation for heterogeneous federated learning," in *Proc. International Conference on Machine Learning (ICML)*, 2021.

[84] S. Horvath, S. Laskaridis, M. Almeida, I. Leontiadis, S. Venieris, and N. Lane, "Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[85] S. Alam, L. Liu, M. Yan, and M. Zhang, "Fedrolex: Model-heterogeneous federated learning with rolling sub-model extraction," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[86] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[87] L. Wang, W. Wang, and B. Li, "Cmfl: Mitigating communication overhead for federated learning," in *Proc. the IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2019.

[88] M. N. Gibbs and D. J. MacKay, "Variational gaussian process classifiers," *IEEE Transactions on Neural Networks*, vol. 11, no. 6, pp. 1458–1464, 2000.

[89] C. T. Dinh, N. H. Tran, M. N. H. Nguyen, C. S. Hong, W. Bao, A. Y. Zomaya, and V. Gramoli, "Federated learning over wireless networks: Convergence analysis and resource allocation," *IEEE/ACM Transactions on Networking*, vol. 29, no. 1, pp. 398–409, 2021.

[90] A. W. Marshall and I. Olkin, "Multivariate chebyshev inequalities," *The Annals of Mathematical Statistics*, pp. 1001–1014, 1960.

[91] O. Shamir, N. Srebro, and T. Zhang, "Communication-efficient distributed optimization using an approximate newton-type method," in *Proc. the International Conference on Machine Learning (ICML)*, 2014.

[92] Y. LeCun and C. Cortes, "MNIST handwritten digit database," *http://yann.lecun.com/exdb/mnist/*, 2010.

[93] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research)," *http://www.cs.toronto.edu/ kriz/cifar.html*.

[94] Z. Jiang, Y. Xu, H. Xu, Z. Wang, C. Qiao, and Y. Zhao, "Fedmp: Federated learning through adaptive model pruning in heterogeneous edge computing," in *Proc. the 38th International Conference on Data Engineering (ICDE)*, pp. 767–779, 2022.

[95] Y. Jiang, S. Wang, V. Valls, B. J. Ko, W.-H. Lee, K. K. Leung, and L. Tassiulas,

"Model pruning enables efficient federated learning on edge devices," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2022.

[96] X. Sun, X. Ren, S. Ma, and H. Wang, "meprop: Sparsified back propagation for accelerated deep learning with reduced overfitting," in *Proc. International Conference on Machine Learning (ICML)*, 2017.

[97] F. Haddadpour and M. Mahdavi, "On the convergence of local descent methods in federated learning," *arXiv preprint arXiv:1910.14425*, 2019.

[98] Y. Zhan, P. Li, and S. Guo, "Experience-driven computational resource allocation of federated learning by deep reinforcement learning," in *2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pp. 234–243, 2020.

[99] B. Luo, X. Li, S. Wang, J. Huang, and L. Tassiulas, "Cost-effective federated learning design," in *Proc. the IEEE Conference on Computer Communications (INFOCOM)*, pp. 1–10, 2021.

[100] 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 *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[101] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," in *Proc. the International Conference on Learning Representations (ICLR)*, 2020.

[102] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *Proc. IEEE conference on computer vision and pattern recognition (CVPR)*, 2015.

[103] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theo-

retical guarantees: A model-agnostic meta-learning approach," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[104] M. Khodak, M.-F. F. Balcan, and A. S. Talwalkar, "Adaptive gradient-based meta-learning methods," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019.

[105] Y. Jiang, J. Konečný, K. Rush, and S. Kannan, "Improving federated learning personalization via model agnostic meta learning," in *Proc. Workshops at Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[106] C. T Dinh, N. Tran, and J. Nguyen, "Personalized federated learning with moreau envelopes," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 21394–21405, 2020.

[107] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *Proc. the International Conference on Machine Learning (ICML)*, 2021.

[108] F. Hanzely and P. Richtárik, "Federated learning of a mixture of global and local models," *arXiv preprint arXiv:2002.05516*, 2020.

[109] O. Marfoq, G. Neglia, A. Bellet, L. Kameni, and R. Vidal, "Federated multi-task learning under a mixture of distributions," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[110] M. G. Arivazhagan, V. Aggarwal, A. K. Singh, and S. Choudhary, "Federated learning with personalization layers," *arXiv preprint arXiv:1912.00818*, 2019.

[111] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai, "Exploiting shared representa-

tions for personalized federated learning," in *Proc. International Conference on Machine Learning (ICML)*, 2021.

[112] K. Pillutla, K. Malik, A.-R. Mohamed, M. Rabbat, M. Sanjabi, and L. Xiao, "Federated learning with partial model personalization," in *Proc. International Conference on Machine Learning (ICML)*, 2022.

[113] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," vol. 33, pp. 19586–19597, 2020.

[114] S. Sarkar and A. K. Ghosh, "On perfect clustering of high dimension, low sample size data," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 9, pp. 2257–2272, 2019.

[115] A. Shamsian, A. Navon, E. Fetaya, and G. Chechik, "Personalized federated learning using hypernetworks," in *Proc. International Conference on Machine Learning (ICML)*, 2021.

[116] D. Kreuzberger, N. Kühl, and S. Hirschl, "Machine learning operations (mlops): Overview, definition, and architecture," *IEEE Access*, 2023.

[117] W. Deng and L. Zheng, "Are labels always necessary for classifier accuracy evaluation?," in *Proc. the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 15069–15078, 2021.

[118] D. Guillory, V. Shankar, S. Ebrahimi, T. Darrell, and L. Schmidt, "Predicting with confidence on unseen distributions," in *Proc. the IEEE/CVF international conference on computer vision (ICCV)*, pp. 1134–1144, 2021.

[119] J. Chen, F. Liu, B. Avci, X. Wu, Y. Liang, and S. Jha, "Detecting errors and estimating

accuracy on unlabeled data with self-training ensembles," vol. 34, pp. 14980–14992, 2021.

[120] S. Garg, S. Balakrishnan, Z. C. Lipton, B. Neyshabur, and H. Sedghi, "Leveraging unlabeled data to predict out-of-distribution performance," *arXiv preprint arXiv:2201.04234*, 2022.

[121] Y. Guo, T. Lin, and X. Tang, "Towards federated learning on time-evolving heterogeneous data," *arXiv preprint arXiv:2112.13246*, 2021.

[122] J. Kang, Z. Xiong, D. Niyato, Y. Zou, Y. Zhang, and M. Guizani, "Reliable federated learning for mobile networks," *IEEE Wireless Communications*, vol. 27, no. 2, pp. 72–80, 2020.

[123] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. the International conference on artificial intelligence and statistics (AISTATS)*, pp. 2938–2948, PMLR, 2020.