COOPERATION & COMPETITION: MECHANISM DESIGN FOR FEDERATED OPTIMIZATION OF EDGE INTELLIGENCE

LEIJIE WU

PhD

The Hong Kong Polytechnic University

2024

The Hong Kong Polytechnic University
Department of Computing

Cooperation & Competition: Mechanism Design for Federated Optimization of Edge Intelligence

Leijie Wu

A thesis submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy
May 2024

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

Signature: ____________________

Name of Student: Leijie Wu

Abstract

With the proliferation of Internet of Things (IoT) applications, a huge amount of data is generated at the network edge. Due to bandwidth, storage, and most importantly privacy concerns, it is impractical to move the local data to the cloud for centralized analytics model training. Federated learning (FL) has been widely recognized as a promising approach that enables individual edge devices (also known as “clients”) to train a global model cooperatively without exposing their data and other private information.

However, although FL has substantial advantages, it still faces the following challenges: First, due to the significant energy consumption for joining FL, the non-independent and identically distributed (Non-IID) data and heterogeneous hardware resources, the clients may be reluctant to participate in the FL without proper rewards from well-designed incentive mechanisms. Second, the Non-IID data distribution among extensive clients makes it impossible for a single global model to adapt to the requirements of all clients simultaneously. This naturally derives the demand of personalization on different clients, i.e., each client requests a personalized model that can perfectly fit their local Non-IID data. Third, the extensive clients involved in the FL system have strong dynamics. i.e., the new clients join and bring new knowledge, while the old ones exit and leave obsolete knowledge in the system. Normally, a fixed-size model has an upper limit on its knowledge capacity, i.e., it cannot learn new knowledge indefinitely. Therefore, under the cooperation of all clients, model unlearning for obsolete knowledge is highly required in the FL system to accommodate its strong dynamics.

In this thesis, we investigated the problems and challenges of these issues from the perspectives of game theory and mechanism design, where the corresponding solutions are proposed to handle them. For the first issue, we present long-term online VCG auction incentive mechanisms based on deep reinforcement learning, which can adaptively assign proper rewards to clients with different resource and data conditions. It considers several vital economic properties to guarantee a sustainable environment for the long-term development of the FL system. For the second issue, we propose a multiwise collaboration framework based on cooperative game theory, which only encourages clients with relevant data distribution for collaboration and trains their own personalized model. For the third issue, we take the early step to comprehensively investigate the machine unlearning paradigm in the context of FL (i.e., federated unlearning) and thereby propose a general pipeline for federated unlearning based on stochastic gradient ascent (SGA) and client cooperation.

We conduct extensive experiments to show the remarkable performance improvement of our proposed methods compared with the existing methods on various datasets and settings.

Publications Arising from the Thesis

- **Leijie Wu**, Song Guo, Yaohong Ding, Yufeng Zhan, and Jie Zhang. "*Rethinking Personalized User Collaboration when Facing An Agnostic Federated Learning System*". IEEE Transactions on Mobile Computing (TMC) (CCF-A), 2024.

- **Leijie Wu**, Song Guo, Yi Liu, Zicong Hong, Yufeng Zhan, and Wenchao Xu. "*Long-term Adaptive VCG Auction Mechanism for Sustainable Federated Learning with Periodical Client Shifting*". IEEE Transactions on Mobile Computing (TMC) (CCF-A), 2023.

- **Leijie Wu**, Song Guo, Yi Liu, Zicong Hong, Yufen Zhan, and Wenchao Xu. "*Sustainable Federated Learning with Long-term Online VCG Auction Mechanism*". IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)(CCF-B), 2022.

- **Leijie Wu**, Song Guo, Junxiao Wang, Zicong Hong, Jie Zhang, and Yaohong Ding. "*Federated Unlearning: Guarantee the Right of Clients to Forget*". IEEE Network (JCR-Q1), 2022.

- Yi Liu*, **Leijie Wu***, Yufeng Zhan, Song Guo, and Zicong Hong (* indicates Co-first author). "*Incentive-Driven Long-term Optimization for Edge Learning by Hierarchical Reinforcement Mechanism*". IEEE 41st International Conference on Distributed Computing Systems (ICDCS) (CCF-B), 2021.

- Yi Liu, Song Guo, Yufeng Zhan, **Leijie Wu**, Zicong Hong, and Qihua Zhou. "*Chiron: A Robustness-aware Incentive Scheme for Edge Learning via Hierarchical Reinforcement Learning*". IEEE Transactions on Mobile Computing (TMC) (CCF-A), 2023.

- Yufeng Zhan, Peng Li, **Leijie Wu**, and Song Guo. "*L4L: Experience-driven computational resource control in federated learning*". IEEE Transactions on Computers (TC)(CCF-A), 2021.

- Yufeng Zhan, Jie Zhang, Zicong Hong, **Leijie Wu**, Peng Li, and Song Guo, "*A Survey of Incentive Mechanism Design for Federated Learning*". IEEE Transactions on Emerging Topics in Computing (TETC) (JCR Q1), 2021.

Acknowledgments

I would like to express my sincere gratitude to the individuals and institutions that have played an integral role during my Ph.D. study. Their support, guidance, and encouragement have been indispensable in this challenging yet rewarding journey.

First of all, I would like to express my heartfelt gratitude to my supervisor, Prof Yan Liu, for her great inspiration and guidance in pursuing my PhD degree at The Hong Kong Polytechnic University. Despite all the ups and downs I faced during my four years of PhD, Prof. Yan Liu always encouraged me to move forward and guided me to find the right path for my research! Her expertise, continuous guidance, and constructive feedback have shaped the trajectory of my research and contributed significantly to the quality of this thesis. It is my honor to become a student of such a brilliant researcher and to benefit greatly from her teachings!

Then, I would like to thank all the research collaborators during my Ph.D. study, who have shared their expertise, proposed constructive advice, discussed research insights, and worked together. In particular, thanks to Prof. Yufeng Zhan from the Beijing Institute of Technology, Prof. Wenchao Xu, Dr. Junxiao Wang, Mr. Zicong Hong, Ms. Jie Zhang, Mr. Rui Zhang, Mr. Yaohong Ding, Dr. Xin Xie, Mr. Yi Liu from the Hong Kong Polytechnic University, Prof. Richard Yida Xu from the Hong Kong Baptist University, Prof. Anne-Marie Kermarrec, Dr. Marinus Abraham de Vos, and Mr. Akash Balasaheb Dhasade from the École Polytechnique Fédérale de Lausanne.

Next, I would like to thank all my lovely and fantastic group mates and friends:

Thanks to Yufeng Zhan, Junxiao Wang, Wenchao Xu, Zicong Hong, Jie Zhang, Yaohong Ding, Rui Zhang, Xin Xie, Enyuan Zhou, Yang Deng, Zhihao Qu, Quan Chen, Haozhao Wang, Qihua Zhou, Ziming Liu, Yingchun Wang, Bingjie Wang, Yi Liu at the Hong Kong Polytechnic University. Thanks to Anne-Marie Kermarrec, Marinus Abraham de Vos, Rafael Pereira Pires, Akash Balasaheb Dhasade, Madeleine Robert, Rishi Sharma, Elia Guerra at the École Polytechnique Fédérale de Lausanne. They give me encouragement and support during my Ph.D. study. I am grateful to meet such a wonderful and talented group of people to work with and learn from!

Last but not the least, I would like to sincerely thank my parents (Mr. Haiming Wu & Ms. Fang Lai) and my girlfriend Ms. Shuyuan Huang. They fully understand and support me in in pursuing my Ph.D., not only by giving me financial assistance, but also by granting me emotional comfort! Without their deep love and constant supports, this thesis will never have been completed.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

In summary, this thesis investigates the complex competition and cooperation relationship among massive edge devices in FL systems, which is caused by clients' heterogeneous local resources and non-independent and identically distributed (Non-IID) data. This is a critical research problem in the field of federated learning with wide-ranging applications, which has also attracted significant attention from numerous researchers. In this chapter, we first provide a brief overview of the research challenges and problems involved in this thesis in Section 1.1. Then, we summarize and highlight the main contributions of this thesis in Section 1.2. Finally, we illustrate the thesis organization in Section 1.3 to provide a quick roadmap guide.

1.1 Thesis Overview

With the rapid advancement of machine learning technology nowadays, it is widely applied in a variety of different fields, such as finance, smart healthcare, smart cities, autonomous driving. Different from the traditional cloud-based centralized data centers, the above application scenarios are usually deployed in a series of network edge devices such as smartphones, laptops, cars, and so on, resulting in a huge amount of data being generated at the network edge. Massive data growth poses serious challenges to the traditional centralized data processing framework regarding computation, communication and storage resources [89, 114]. As the recent performance improvements of mobile devices in chip, storage and other various aspects, edge devices are equipped with powerful computation capacity to directly process data locally. This edge-side data processing manner brings a series of benefits: faster information extraction, less response time, and better bandwidth allocation [82, 59]. Furthermore, the widespread utilization of deep learning technologies has also raised increasing clients' concerns about data privacy protection, client's refusal of data sharing resulting in every edge device becoming a "data silo" [72, 51]. Without the client data sharing, it's impractical for traditional centralized framework to obtain desirable model on each data silo, which only has limited data.

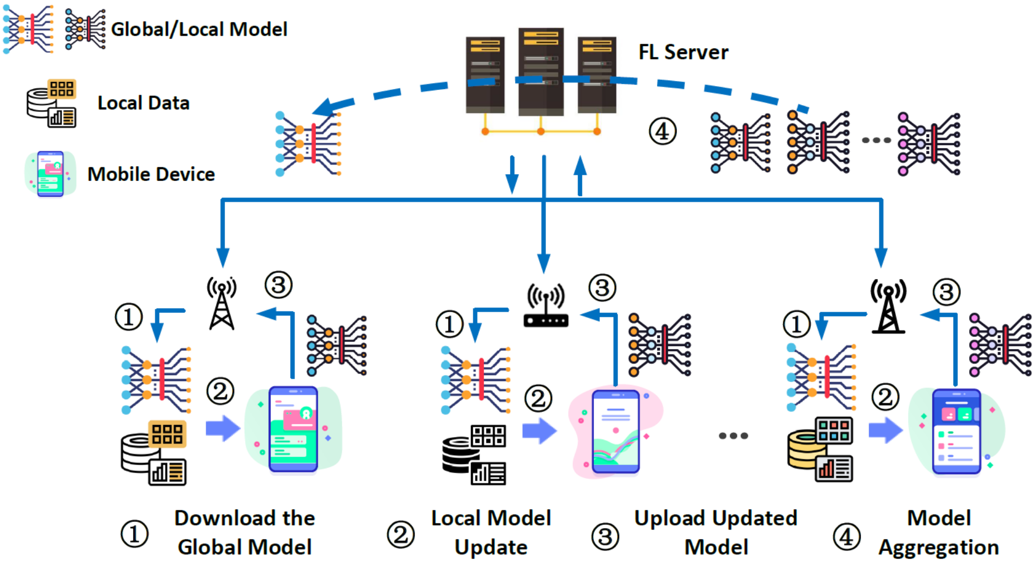


Figure 1.1: An overall demonstration of federated learning system architecture. Different from the traditional distributed machine learning paradigm, the edge devices (clients) in the FL system are dynamic (old clients exit & new clients join), and all the data are generated in the local edge, which is highly heterogeneous.

To break down data silos to achieve collaborative model training while meeting the data privacy, security and regulatory requirements, **Federated Learning** (FL) has been proposed as a new generation of distributed machine learning paradigm, which allows various clients to collaboratively train a shared global model without sharing their local data [47]. A detailed demonstration of the FL system architecture is shown in Figure 1.1. First, all clients (mobile devices) download the shared global model from the server side. Then, each client updates the downloaded model on their own local data to obtain respective local models. Next, each client uploads their local model to the server side. Finally, the server generates a new global model by conducting model aggregation process on all local models. The above steps will keep repeating round-by-round until the global model reaches convergence.

Despite its desirable advantages and potential on various edge applications, FL faces a number of urgent technical challenges. First, the local data generated by different client has a high degree of heterogeneity, which is academically referred to as "*non independently and identically distributed*" (Non-IID). This data heterogeneity will significantly degrade the global model performance and the training efficiency of FL [113, 57]. Second, the diversity of clients (e.g., smartphones, laptops, cars, etc.) leads to the local resource heterogeneity of clients, including computation capabilities, communication bandwidth, storage memories and so on. The resource heterogeneity can lead to an uncoordinated FL training process since the server need to wait for all clients to finish local model uploading, which further undermines the overall training efficiency [55, 38]. Third, FL system improve the generalization capability of its global model by involving extensive clients for training, which also results in a strong client dynamics within the system. After using the services provided by FL, many old clients will take their local data away and directly exit the FL system, while other new clients continue to participate in the system simultaneously. However, a fixed-size model can only contain limited knowledge, Obsolete knowledge extracted from old clients' data still remains in the global model after their leaving, which significantly hinders the model from learning new knowledge.

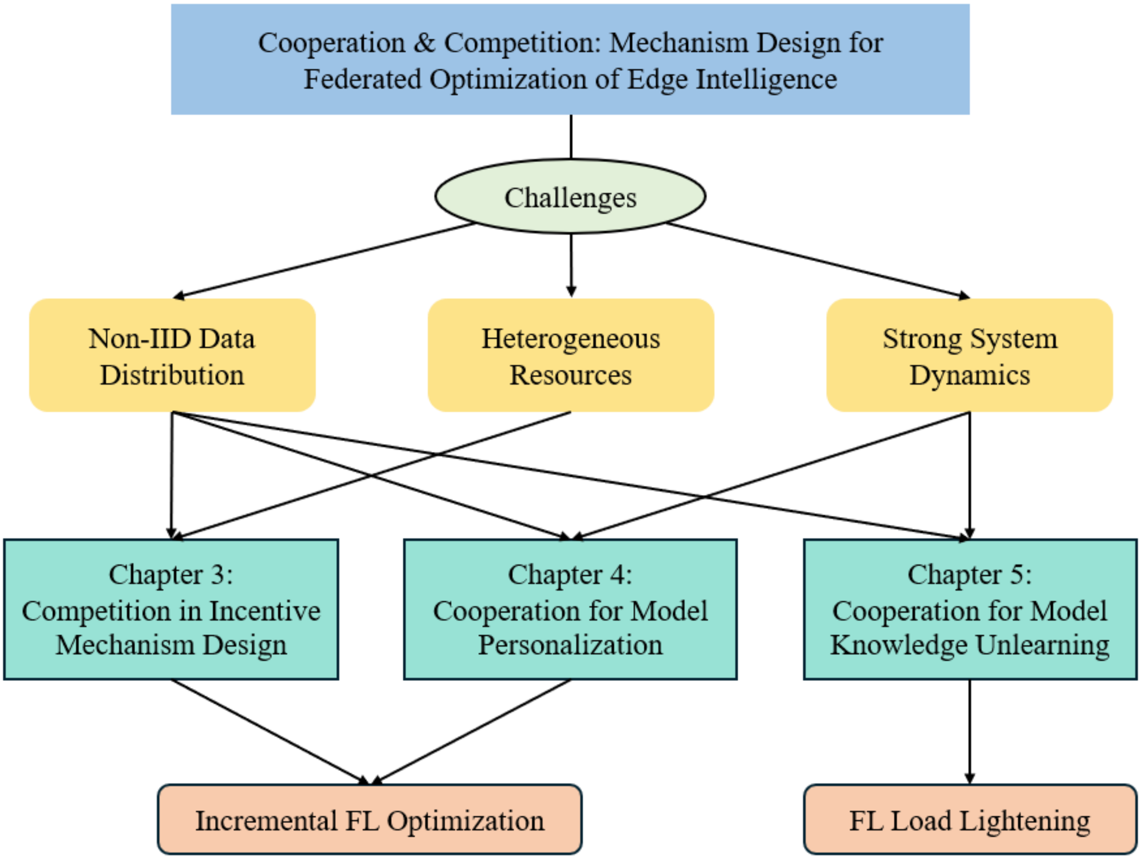

Cooperation & Competition: Mechanism Design for Federated Optimization of Edge Intelligence
Challenges
Non-IID Data Distribution
Heterogeneous Resources
Strong System Dynamics
Chapter 3: Competition in Incentive Mechanism Design
Chapter 4: Cooperation for Model Personalization
Chapter 5: Cooperation for Model Knowledge Unlearning
Incremental FL Optimization
FL Load Lightening

Figure 1.2: The theoretical thesis framework overview

Therefore, in this thesis, we focus on utilizing the coexisting competition and cooperation relationships among various clients to design effective mechanisms, which can tackle the above issues raised by the heterogeneous FL environment and its dynamics. The theoretical research framework of this thesis is demonstrated on Figure 1.2, which consists of the following three main parts. In the first part, we consider to utilize the competition relationship among clients and present a long-term online VCG auction-based mechanism that can incentivize the required clients to participate in training according to the demands of the different stages in FL. In the second part, we rethink the nature of client collaboration by domain relevance in FL and propose to design personalized coalition cooperation for each client to generate their customized models. In the third part, we innovatively derive the machine unlearning problem to the FL context and define different FL unlearning types. Furthermore, we proposed a general pipeline for federated learning to collaboratively unlearn specific knowledge from the current model.

1.2 Thesis Contribution

In this section, we briefly summarize the contributions of this thesis as follows:

1. Incentive Mechanism Design under Heterogeneous Client Competition. Federated learning (FL) poses challenges related to client participation reluctance and biased server selection, necessitating the implementation of a suitable incentive mechanism. Moreover, the heterogeneity of client resources, encompassing local data, computation, and communication capabilities, further contributes to disparities in server selection for FL training. Consequently, a competitive relationship emerges among clients in the absence of substantial incentives from the server side. To address these challenges, we propose a long-term online Vickrey-Clarke-Groves (VCG) auction mechanism for FL that employs an experience-driven deep reinforcement learning algorithm to ascertain the optimal strategy. Notably, this mechanism extends the economic properties pertinent to the successive FL process, ensuring its efficacy in the long run. Additionally, we incorporate knowledge transfer techniques to mitigate the excessive training overhead incurred by the VCG payment rules. By capitalizing on the environmental similarity among sub-auctions, we introduce strategy sharing, which yields a substantial 50% reduction in training time. To substantiate the efficacy of our approach, we provide theoretical proofs of the extended economic properties and conduct extensive experiments using multiple real-world datasets. Our results demonstrate a notable improvement over state-of-the-art approaches, with a 36% increase in long-term social welfare achieved alongside a 37% reduction in payment.

2. Rethinking Personalized Client Collaboration in Federated Learning.

The model performance in FL can be compromised due to the high heterogeneity in clients' local data distributions, commonly known as Non-IID (non-independent and identically distributed). Moreover, collaboration among highly dissimilar clients exacerbates this performance degradation. Personalized FL seeks to mitigate this by enabling clients to collaborate primarily with others who have similar data characteristics, thereby producing personalized models. To tackle this issue, we enhance personalized client collaboration in FL by introducing a metric for domain relevance between clients. Specifically, to facilitate optimal coalition formation, we measure the marginal contributions of client models using coalition game theory, providing a more accurate representation of potential client domain relevance within the FL privacy-preserving framework. Based on this metric, we then adjust each client's coalition membership and implement a personalized FL aggregation algorithm that is robust to Non-IID data domain. We provide a theoretical analysis of the algorithm's convergence and generalization capabilities. Our extensive evaluations on multiple datasets, including MNIST, Fashion-MNIST, CIFAR-10, and CIFAR-100, and under varying Non-IID data distributions (Pathological and Dirichlet), demonstrate that our personalized collaboration approach consistently outperforms contemporary benchmarks in terms of accuracy for individual clients.

3. Collaborative Model Knowledge Unlearning in Federated Learning.

In general, an FL system will include extensive edge devices (i.e., clients) in its training process to collaboratively strengthen its model generalization capability. However, these various clients also bring strong dynamics to the FL system (e.g., new clients join & old clients exit). Since a fixed-size model can only contain a limited amount of knowledge, obsolete knowledge from old clients still remains in the current model after they leave the FL system, which significantly impacts the ability of the model to learn new knowledge and adapt to new clients. To solve these issues, we present a general pipeline for federated learning to collaboratively unlearn specific knowledge from the current model, which is referred to as "Federated Unlearning (FU)". Besides, we define three common types of federated unlearning requests namely class unlearning, client unlearning, and sample unlearning. Then, we revisit the nature of how the training data affects the FL model and thereby empower the proposed pipeline with the reverse stochastic gradient ascent (SGA) and elastic weight consolidation (EWC), which can achieve FU for different types of unlearning requests. Various experiments are conducted to verify the effectiveness of the proposed method in both aspects of unlearning efficacy and efficiency.

1.3 Thesis Roadmap

The rest of this thesis is composed of five chapters, which are organized as follows. In § 2, we conduct a detailed background knowledge review for the research areas covered in this thesis, including Federated Learning (FL), Machine Unlearning (MU), Federated Unlearning (FU), Game Theory, and Deep Reinforcement Learning (DRL). In § 3, we introduce a sustainable federated learning framework by the long-term online VCG auction mechanism, which can motivate a continual stream of clients to participate in FL training from a long-term perspective. In § 4, we rethink the collaboration relationship among heterogeneous clients in the FL system and utilize cooperative game theory to guide the personalized collaboration of each client to improve their respective performance. In § 5, we define different types of unlearning requests in the context of federated learning and propose a general pipeline based on reverse stochastic gradient ascent (SGA) to collaboratively unlearn the specified knowledge from the FL model. In § 6, we further summarize this thesis, explore some potential research directions, and provide future visions involved research topics.

Chapter 2

Background

2.1 Federated Learning

Federated learning (FL) constitutes a decentralized machine learning framework comprising a server and a collection of clients denoted by $\mathcal{N} = 1, \cdots, N$. Each client i maintains a local dataset $\mathcal{D}_i$, comprising its individual data samples $\{\boldsymbol{x}_j, y_j\}_j \in \mathcal{D}_i$, and engages in collaborative training on a shared global model governed by a loss function f.

In the t-th round, each client i initially receives the global model $\boldsymbol{\omega}^t$ from the server and conducts training on its local dataset $\mathcal{D}i$ to compute its local loss, expressed as

$$F_i(\boldsymbol{\omega}^t) = \frac{1}{d_i} \sum_{j \in \mathcal{D}_i} f(j, \boldsymbol{\omega}^t), \tag{2.1}$$

where $d_i = |\mathcal{D}_i|$ denotes the size of client i's local dataset, and $f(j, \boldsymbol{\omega}^t)$ represents the loss incurred by model $\boldsymbol{\omega}^t$ on training sample $j \in \mathcal{D}_i$. Subsequently, the local loss informs the model update via stochastic gradient descent (SGD), yielding the updated local model $\boldsymbol{\omega}_i^t$, given by

$$\boldsymbol{\omega}_i^t = \boldsymbol{\omega}^t - \eta_i \frac{\partial F_i(\boldsymbol{\omega}^t)}{\partial \boldsymbol{\omega}^t}, \tag{2.2}$$

where η_i denotes the local learning rate. Each client then transmits its updated local model $\boldsymbol{\omega}_i^t$ to the server for model aggregation, governed by

$$\boldsymbol{\omega}^{t+1} = \sum_{i=1}^{N} \frac{d_i}{d} \boldsymbol{\omega}_i^t, \tag{2.3}$$

where $d = \sum_{i=1}^{N} d_i$ represents the total data size across all clients. Finally, the server dispatches the new global model $\boldsymbol{\omega}^{t+1}$ to the clients as the starting point for the subsequent round $t+1$. This iterative process continues until the model converges or fulfills predefined convergence criteria.

Typically, the server maintains its own validation dataset $\mathcal{D}v$, aiming to identify the optimal global model $\boldsymbol{\omega}^*$ that minimizes the validation loss function $F(\boldsymbol{\omega})$, given by

$$\boldsymbol{\omega}^* = \arg\min_{\boldsymbol{\omega}} F(\boldsymbol{\omega}), \text{ where } F(\boldsymbol{\omega}) = \frac{1}{|\mathcal{D}_v|} \sum_{j \in \mathcal{D}_v} f(j, \boldsymbol{\omega}). \tag{2.4}$$

Here, $|\mathcal{D}_v|$ denotes the size of the validation dataset, and $f(j, \boldsymbol{\omega})$ represents the loss incurred by model $\boldsymbol{\omega}$ on the validation sample $j \in \mathcal{D}_v$. A visual representation of the federated learning system is depicted in Fig. 2.1.

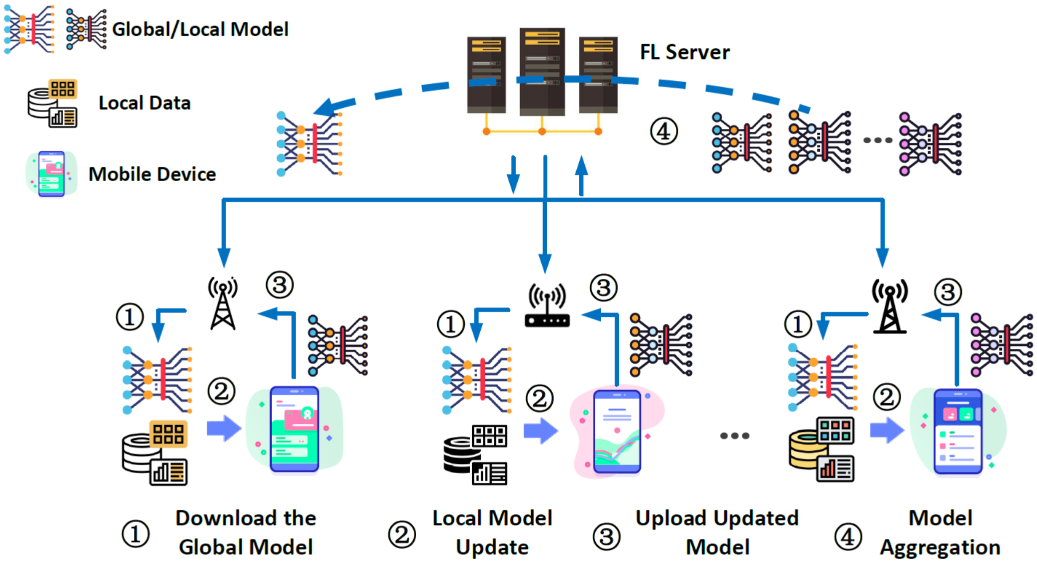


Figure 2.1: Overall architecture of federated learning.

Personalized Federated Learning (PFL).

Recently we have witnessed significant progress in developing novel methods that address different challenges in FL [42]. In particular, there have been several works on various aspects of FL, including preserving the privacy of clients and lowering communication cost. Several works develop algorithms for the homogeneous setting, where the data points of all clients are sampled from the same probability distribution, while more other works study the statistical heterogeneity of clients data points in FL, but they do not attempt to find a personalized solution for each client.

Personalized Federated Learning (PFL) is a collaborative approach that aims to tailor individualized models for each client, taking into account their unique data distribution while maintaining data privacy. Consider a group of clients denoted as $C_1, C_2, \ldots, C_n$, where each client employs a model $\mathcal{M}$ with distinct weight parameters $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \ldots, \boldsymbol{\omega}_n$. The personalized models for each client can be represented as $\mathcal{M}(\boldsymbol{\omega}_i)$. In contrast to traditional federated learning, PFL recognizes that each client i possesses a locally held dataset $\mathcal{D}_i$, which is independently sampled from its own distinct data distribution $\mathcal{P}_i$. Let ℓ_i denote the loss function associated with client i, and $\mathcal{L}_i$ represent the average loss over the private local dataset $\mathcal{D}_i$. Mathematically, this is expressed as:

$$\mathcal{L}_i(\boldsymbol{\omega}_i) = \frac{1}{d_i} \sum_{j \in \mathcal{D}_i} \ell_i(j, \boldsymbol{\omega}_i), \tag{2.5}$$

where d_i denotes the size of the dataset $\mathcal{D}_i$, and j represents an individual data sample within $\mathcal{D}_i$. Let $\boldsymbol{\omega}$ denote the set of personalized model parameters $\{\boldsymbol{\omega}_i\}_{i=1}^{n}$ for all clients. The optimization objective of PFL is to find the optimal personalized model parameter set $\boldsymbol{\omega}^* = \{\boldsymbol{\omega}_1^*, \boldsymbol{\omega}_2^*, \ldots, \boldsymbol{\omega}_n^*\}$ that minimizes the average loss across all clients:

$$\boldsymbol{\omega}^* = \arg\min_{\boldsymbol{\omega}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_i(\boldsymbol{\omega}_i). \tag{2.6}$$

2.2 Machine Unlearning & Federated Unlearning

2.2.1 Machine Unlearning

Many enterprises and organizations collect clients' data to train machine learning (ML) models for a wide range of applications, e.g., medical diagnosis, movie recommendation, etc. While the clients get convenience from these promising technologies, the personal information is also recorded and used all the time, leading to concerns about privacy leakage. With increased attention being paid to data privacy, clients tend to delete or hide their personal information after the service is no longer required. Recent privacy regulations like the "General Data Protection Regulatio (GDPR)" and "California Consumer Privacy Act (CCPA)" grant data owners with the "right to be forgotten" [22, 13]. In this case, the most straightforward way is to delete the data of particular clients from the dataset and retrain a new model from scratch using the remaining data. However, it is impractical to naively retrain models after every deletion request in terms of the expensive training cost of both time and money. Machine unlearning (MU) is proposed to unlearn the knowledge of the data that needs to be deleted from the model without incurring the cost of retraining from scratch.

Following the introduction of *MU* in [14], several algorithms have been proposed [20, 28, 34, 30, 31, 8]. The existing research primarily centers around the process of unlearning knowledge from straightforward classification models, such as logistic regression. However, these approaches are not applicable to more intricate models like deep neural networks. Furthermore, certain algorithms have specific limitations, rendering them suitable only for particular model architectures or scenarios. For instance, the algorithm proposed by Brophy *et al.* exclusively fits random forest models [11], while the work of Nguyen *et al.* focuses solely on Bayesian learning [71]. In traditional machine learning scenarios, all the local data of clients will be uploaded to the server for centralized management, so the server has a high level of flexibility to conduct arbitrary operations on all data. Therefore, various unlearning techniques (e.g., ensemble learning for data splitting [8] or gradient amnesia of arbitrary batch [33]) are designed to operate in settings where training data is readily available.

More specifically, the objective of the unlearning process, referred to as Model Unlearning (MU), is to eliminate the influence of a specific subset of data, denoted as the "forget dataset" D_f, on the trained model parameters $\boldsymbol{\theta}$. Conversely, the remaining dataset $D\backslash D_f$ is referred to as the "retain dataset." The unlearning algorithm $\mathcal{U}$ is defined as $\boldsymbol{\theta}_f = \mathcal{U}(\boldsymbol{\theta}, D_f)$, where $\boldsymbol{\theta}_f$ represents the unlearned model. The primary goal of unlearning is to obtain an unlearned model $\boldsymbol{\theta}_f$ that exhibits comparable performance to a model trained solely on the retain dataset $(D\backslash D_f)$. In other words, the unlearned model $\boldsymbol{\theta}_f$ should demonstrate favorable performance on the retain dataset while exhibiting a state that it has never been exposed to the forget dataset D_f. The aim is to achieve a model that effectively generalizes to the data it was originally trained on while minimizing the impact of the forget dataset.

2.2.2 Federated Unlearning

In federated learning (FL), beyond the “right to be forgotten”, removing data from the model proves essential for several other purposes. For instance, being able to quickly eliminate outdated, manipulated, or erroneously included data enhances the security, responsiveness, and reliability of FL systems [17].

Federated Unlearning (FU) is a technique for knowledge removal from a trained model that operates in a distributed and collaborative manner. Compared to traditional Model Unlearning (MU), Federated Unlearning presents additional challenges stemming from its decentralized nature. There are two primary reasons why FU is more challenging than MU. Firstly, client data is strictly confined to the respective clients' local environments and cannot be transferred to a central server in FU. This constraint necessitates active client participation in the unlearning process. Unlike MU, where the entire dataset is typically available centrally, FU requires coordinated efforts from multiple clients to collectively contribute to the unlearning process. It also implies that the server cannot conduct any fine-grained operations at the data level, rendering many existing MU techniques inapplicable in FL settings. Second, when the original model is also trained in a distributed, collaborative manner, e.g., using FL, the server does not always have access to intermediate, granular training information produced by clients. Some MU techniques rely on recorded training information to carry out an unlearning operation [33]. In FU settings, however, the server might be unable to collect specific training information for unlearning, e.g., model updates per batch for each client.

In our scenario, we have a FL system consisting of N clients, such as mobile devices. Each client $i \in N$ possesses a local training dataset D_i. The clients engage in collaborative training to develop a global FL model denoted as $\boldsymbol{\theta}$. This training process follows a standard FL algorithm, such as FedAvg [63]. After the global model $\boldsymbol{\theta}$ has been trained, the parameter server, which serves as a central coordinating entity, may receive a request to perform unlearning on a specific subset of data, denoted as D_f. The exact nature of the unlearning process is determined by the characteristics and composition of the subset D_f. The specific definition and composition of D_f determine the type of unlearning that will be carried out on the global model $\boldsymbol{\theta}$. This characterization can vary depending on the particular requirements and objectives of the unlearning request. We distinguish between the following three types of unlearning:

- **Class-level unlearning.** This type of unlearning erases the knowledge of a class. Here, D_f contains the data of the entire class. Denoting by D_i^c the data of class c with the i-th client, we have $D_f = \cup_i D_i^c$, where the union is over all clients i which possess samples of class c. Thus, D_f for class-level unlearning is distributed across clients.

- **Client-level unlearning.** This type of unlearning erases the knowledge of a particular client, e.g., when exercising the right to be forgotten. Here, D_f contains the data of a single client. When unlearning for i-th client, $D_f := D_i$. In this case, D_f is concentrated on a single client.
- **Sample-level unlearning.** This type of unlearning erases the knowledge of one or more samples. Here, D_f can contain arbitrary samples from arbitrary clients. Sample-level unlearning is the most general and difficult form of unlearning [100, 75]. We note that D_f can be distributed across clients.

Class-level and client-level unlearning are the two most common use cases in federated settings [100, 75].

Retraining from scratch. A naive way to unlearn D_f is to retrain the model from scratch while omitting samples from D_f. While this algorithm perfectly achieves the desired goal, complete retraining is prohibitively expensive as it initiates new FL training rounds on $D \backslash D_f$. Even executing a single unlearning request in such a way is highly compute- and time-intensive. We refer to this algorithm as Retrain-Or, i.e., as a retraining oracle due to its ideal achieved performance.

Gradient calibration. One way to speed up retraining from scratch is to reuse gradient information from the original training to avoid regenerating all gradients from scratch. However, these gradients must be adapted based on the target D_f and $D \backslash D_f$, through a process referred to as *gradient calibration.* Algorithms employing gradient calibration like FedEraser [56] thus trade the central server's storage for unlearned model's construction time by leveraging historical parameter updates from FL training. However, the storage costs can grow quite large while the efficiency gains compared to retraining from scratch remain modest.

SGA. Another approach to FU includes performing several SGA steps on the forget dataset D_f [100]. In this approach, during each round of FL, clients that possess data in D_f perform local SGA steps, while the server aggregates the updates received from these clients. However, the SGA training process introduces noise that can have a detrimental impact on the performance of the remaining data. To mitigate the effects of this noise, subsequent recovery rounds are necessary. During these recovery rounds, clients engage in regular SGD training on the remaining data, i.e., $D \backslash D_f$. As a result, the unlearning process consists of two stages: unlearning on D_f and recovery on $D \backslash D_f$. Each round involves updating the model with the entire dataset. Unfortunately, this process can be inefficient, particularly when dealing with large volumes of data or when multiple unlearning requests need to be executed.

S2U. Inspired by the observation that the up- or down-scaling of model updates can substantially influence the global model, S2U scales down the forgetting client's updates while scaling up the updates of remaining clients [26]. Its unlearning and recovery stages are integrated together, similar to Retrain-Or. We remark that S2U is only applicable to client-level unlearning.

Model Pruning. In the FU-MP approach proposed by Wang *et al.* [94], model pruning is employed as the unlearning mechanism. The process begins by measuring the class discrimination of different channels in the model, which reflects the relevance of different classes to the model's channels. Based on these measurements, the most relevant channel of the target class is pruned, effectively unlearning that specific class. Compared to the Retrain-Or approach, FU-MP demonstrates higher efficiency. However, it is important to note that FU-MP is specifically designed for class-level unlearning and may not be applicable to unlearning at a more granular level, such as specific data instances or subsets. One limitation of model pruning, as used in FU-MP, is that it irreversibly modifies the model. Once a channel is pruned, it cannot be easily relearned or recovered. This means that if there is a need to reintroduce the unlearned class or retrain the model with the previously pruned channels, additional measures or techniques would be required.

2.3 Game Theory

Game theory generally refers to the study of mathematical models that describe the behavior of logical decision-makers [69]. It is widely used in many fields such as economics, political science, politics, and computer science. Generally, a game refers to a situation involving a set of players who each have a set of possible choices, in which the outcome for any individual player depends partially on the choices made by other players.

A player is the basic entity of a game who makes decisions and then performs actions. A game is a precise description of the strategic interaction that includes the constraints of, and payoffs for, actions that the players can take, but says nothing about what actions they actually take. A solution concept is a systematic description of how the game will be played by employing the best possible strategies and what the outcomes might be. The consequence function associates a consequence with each action the decision makers take. A preference relation is a complete relation on the set of consequences which model the preference of each player in the game. A strategy for a player is a complete plan of actions in all possible situations throughout the game. If the strategy specifies to take a unique action in a situation then it is called a pure strategy. If the plan specifies a probability distribution for all possible actions in a situation then the strategy is referred to as a mixed strategy. A Nash equilibrium is a solution concept that describes a steady state condition of the game; no player would prefer to change his strategy as that would lower his payoffs given that all other players are adhering to the prescribed strategy. This solution concept only specifies the steady state but does not specify how that steady state is reached in the game. The Nash equilibrium is the most famous equilibrium, even though there are many other solution concepts used occasionally. This information will be used to define games that have relevant features for representing network security problems.

There are different types of Games in Game theory, they help in the analysis of different types of problems, which can be categorized into two main branches: Cooperative Game and Non-cooperative Game. They are formed on the basis of number of players involved in a game, symmetry of the game, and cooperation among players. A game is cooperative if the players are able to form binding commitments externally enforced (e.g. through contract law). A game is non-cooperative if players cannot form alliances or if all agreements need to be self-enforcing (e.g. through credible threats).

Cooperative games are often analyzed through the framework of cooperative game theory, which focuses on predicting which coalitions will form, the joint actions that groups take, and the resulting collective payoffs. It is opposed to the traditional non-cooperative game theory which focuses on predicting individual players' actions and payoffs and analyzing Nash equilibrium. The focus on individual payoff can result in a phenomenon known as Tragedy of the Commons, where resources are used to a collectively inefficient level. The lack of formal negotiation leads to the deterioration of public goods through over-use and under provision that stems from private incentives.

Cooperative game theory provides a high-level approach as it describes only the structure, strategies, and payoffs of coalitions, whereas non-cooperative game theory also looks at how bargaining procedures will affect the distribution of payoffs within each coalition. As non-cooperative game theory is more general, cooperative games can be analyzed through the approach of non-cooperative game theory (the converse does not hold) provided that sufficient assumptions are made to encompass all the possible strategies available to players due to the possibility of external enforcement of cooperation. While using a single theory may be desirable, in many instances insufficient information is available to accurately model the formal procedures available during the strategic bargaining process, or the resulting model would be too complex to offer a practical tool in the real world. In such cases, cooperative game theory provides a simplified approach that allows analysis of the game at large without having to make any assumption about bargaining powers.

2.4 Deep Reinforcement Learning

The main concept of reinforcement learning is that the agent needs to find the "right" actions to achieve the overall goal through interaction with the environment. This interaction process can be modeled as a Markov Decision Process by a 5-tuple ($\mathcal{S}$, $\mathcal{A}$, R, P, γ), where $\mathcal{S}$ denotes the state set, $\mathcal{A}$ denotes the action set, a reward function R is mapping each state $s \in S$ and action a taken in it to an expected immediate reward $r_t = R(s_t, a_t)$, $P(\cdot|s, a)$ is the transaction probability, while $\gamma \in [0, 1]$ is the discount factor to reflect the diminishing importance of current reward on future ones. The Goal of the MDP is to find a policy $\pi^*(a|s)$ that determinate the selected action a under state s, so as to maximize the expected cumulative reward of the agent, i.e., its $return, g_T = \sum_{t=1}^{T} \gamma^{t-1} r_t$. We define a *trajectory* τ as a sequence of transition from some state s_i to state s_{j+1};$\tau = [(s_i, a_i, r_i, s_{i+1}), ..., (s_j, a_j, r_j, s_{j+1})]$. the expected discounted cumulative reward usually defined as the value function by the Bellman equation:

$$\begin{aligned} V^{\pi}(s) &= \mathbb{E}_{\pi}[\sum_{t=1}^{\infty} \gamma^{t-1} r_t | s_1 = s] \\ &= \mathbb{E}_{\pi}[r_1 + \gamma \sum_{s_\tau \in S} P(s_\tau | s_1, a_1) V^{\pi}(s_\tau)] \end{aligned}$$

the optimal policy, π^*, has a corresponding state-value function $V^*(s)$, therefore, the optimal state-value function can be obtain by:

$$V^{\star}(s) = \max_{\pi} V^{\pi}(s), \forall s \in S \tag{2.7}$$

There are two types of methods that commonly used in reinforcement learning:

Value-based reinforcement Learning: The value-based method is to output the value of all actions, and the agent will choose the action based on the highest value. The action value can be represented by a function approximation, such as a neural network. Let $Q^{\pi}(s, a)$ denotes an approximate action-value function with policy π, which is similar to V^{π}. Correspondingly, the Q function $Q^{\pi}: S \times A \rightarrow R$ maps a state-action pair to the expected return which obtained from that state when the selected action is executed, and then the π is followed from the next state onwards:

$$Q^{\pi}(\mathbf{s}, \mathbf{a}) = \mathbb{E}[R|\mathbf{s}, \mathbf{a}, \pi] \tag{2.8}$$

It should be noticed that the value function V^{*} and Q-function Q^{*} which correspond to the optimal policy π^{*} are the *optimal* value function. The optimal policy π^{*} can be calculated from the optimal Q-function Q^{*}: $\pi^{*}(a|s) = \arg\max_{a} Q(s, a)$, i.e., at each time-step, the agent executes the action that related Q value in current state is maximal. Similar to $V^{\pi}(s)$, the Q value can be calculated by the following expression:

$$Q^{\pi}\left(\mathbf{s}_{t}, \mathbf{a}_{t}\right) = \mathbb{E}_{\mathbf{s}_{t+1}}\left[r_{t+1} + \gamma Q^{\pi}\left(\mathbf{s}_{t+1}, \pi\left(\mathbf{s}_{t+1}\right)\right)\right] \tag{2.9}$$

There are several typical value-based reinforcement learning algorithms: Q-learning [96], SARSA (the state-action-reward-state'-action') [76], DQN (Deep Q-learning) [68] etc. The difference between Q learning algorithm and SARSA algorithm is the action selection strategies. For example, in Q-learning, the agent will firstly update the Q value function according to the action that has maximal Q value, then choose the actual action by ϵ-greedy, while the agent will immediately execute the action choosing by ϵ-greedy and then update the Q value function in SARSA. Their update process of Q value function can be summarized as follows:

- Q-Learning:

$$Q(s, a) \leftarrow Q(s, a) + \alpha[\underbrace{r + \gamma \max_{a'} Q\left(s', a'\right)}_{\text{Q target}} - \underbrace{Q(s, a)]}_{\text{Q eval}} \tag{2.10}$$

- SARSA:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[\underbrace{r + \gamma Q\left(s', a'\right)}_{\text{Q target}} - \underbrace{Q(s,a)]}_{\text{Q eval}} \tag{2.11}$$

Q-learning and SARSA both use Q-table to store the Q values, which is impractical to implement in real-world due to the finite size for state-action pairs. In this case, DQN are proposed to overcome the limitation in Q-table, a deep neural network are employed to approximate the Q value function.

Policy-based reinforcement Learning: In contrast to value-based methods, policy-based methods are the most direct type of reinforcement learning. It can analyze the environment and directly output the probabilities of various actions to be taken in next step, and then choose actions based on the probability. Compared with the policy-based methods that each action might be chosen even if the probability of a certain action is small, the decision part of value-based methods is more definite and merciless.

In policy-based methods, the objective is to find a policy that can maximize the expected discounted cumulative reward, the objective function can be expressed as:

$$J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=1}^{\infty} \gamma^t r_t \left| s_{t+1} \sim P\left(s' | s_t, \pi_\theta\left(s_t\right)\right), s_1 = s \right|\right]. \tag{2.12}$$

Where $P\left(s'|s_t, \pi_\theta\left(s_t\right)\right)$ is the probability function. To find the optimal policy, gradient ascent method can be used to update the policy π_θ:

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\pi_0}\left(\nabla_\theta \log \pi_\theta(s,a) R\left(s'|s, \pi_\theta\left(s\right)\right)\right) \tag{2.13}$$

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta) \tag{2.14}$$

One example of such a method is the REINFORCE algorithm [99]. Standard REINFORCE updates the policy parameters θ in the direction $\nabla_\theta \log \pi_\theta\left(s_t, a_t\right) R_t$, which is an unbiased estimate of $\nabla_\theta J(\pi_\theta)$. It is possible to reduce the variance of this estimate while keeping it unbiased by subtracting a learned function of the state b, known as a baseline, from the return. The resulting gradient is $\nabla_\theta \log \pi_\theta\left(s_t, a_t\right)\left(R_t - b\right)$.

The combination of two methods: The value-based methods are suitable to make one-step update, which is faster than the policy-based methods that using one-episode update. Therefore, another method that combine the advantages of value-based methods and policy-based methods are generated to make the learning process more efficient, that is, Actor-Critic algorithms. In Actor-Critic algorithms, Actor is used to update the parameters of value function.

$$Q_w(s,a) \approx Q^\pi(s,a) \tag{2.15}$$

And Critic is used to update the parameters of policy, the gradients become:

$$\nabla_\theta J(\pi_\theta) \approx \mathbb{E}_{\pi_0}\left(\nabla_\theta \log \pi_\theta(s,a) Q_w\left(s, a\left(s\right)\right)\right) \tag{2.16}$$

TRPO (Trust Region Policy Optimization) [78], DDPG (Deep Deterministic Policy Gradient) [54], A3C (Asynchronous Advantage Actor-Critic) [66] and PPO (Proximal Policy Optimization) [79] are both the implementations of this concept.

Chapter 3

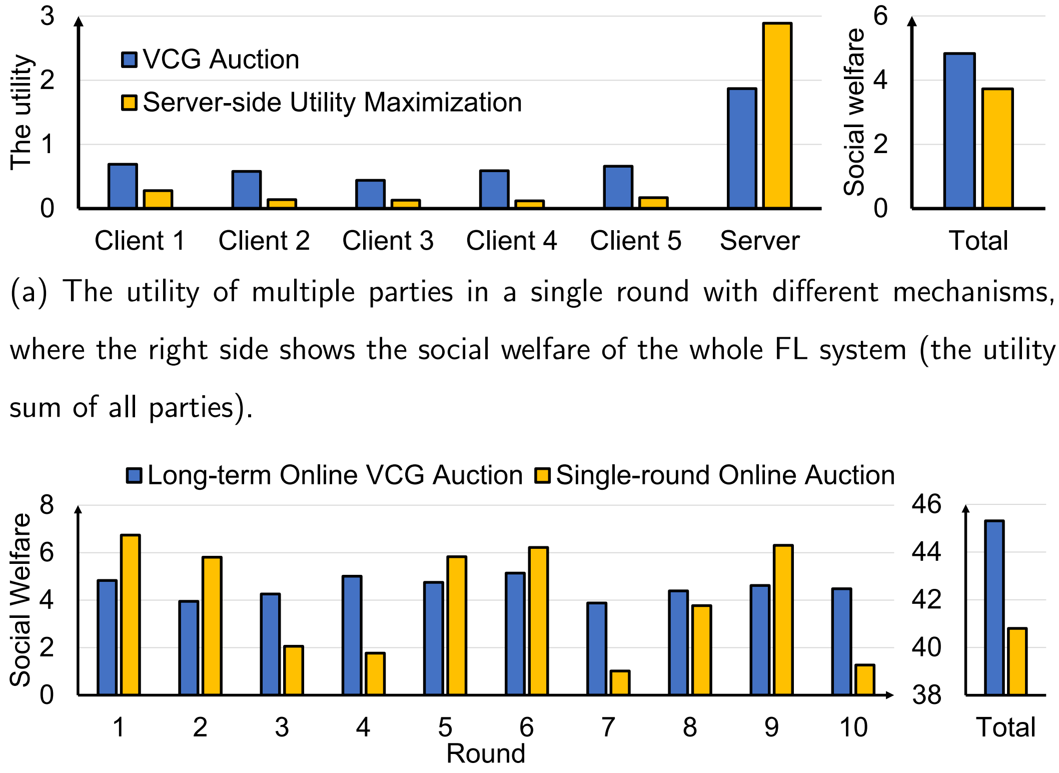
Sustainable Federated Learning with Long-term Online Vickrey–Clarke–Groves Auction Mechanism

In the context of federated learning (FL), the willingness of clients to participate in the energy-intensive FL process is contingent upon the provision of appropriate incentives. Currently, existing incentive mechanisms fail to adequately consider crucial economic properties, such as social welfare, individual rationality, and incentive compatibility. This limitation significantly impedes the sustainability of FL in attracting a larger pool of clients. To address this issue, the Vickrey-Clarke-Groves (VCG) auction emerges as an ideal mechanism that simultaneously ensures the fulfillment of all vital economic properties, thereby maximizing social welfare. Nevertheless, the direct application of the VCG auction to FL scenarios encounters several challenges. Firstly, the need for precise analytical derivation of the optimal strategy poses a problem, given the inherent model-unknown and privacy-sensitive nature of FL. Secondly, the current auction modeling approach decomposes the entire process into multiple independent rounds, solving them sequentially. This approach, however, disrupts the inherent correlation between rounds in the long-term training process of FL. To surmount these challenges, this chapter introduces a long-term online VCG auction mechanism for FL, which leverages an experience-driven deep reinforcement learning algorithm to determine the optimal strategy. Moreover, we extend the economic properties relevant to the successive FL process in a long-term context. Additionally, knowledge transfer techniques are employed to mitigate the excessive training overhead resulting from VCG payment rules. By capitalizing on the environmental similarity among sub-auctions, we develop a strategy sharing framework that effectively reduces the training time by half. Finally, we provide theoretical proofs of the extended economic properties and conduct extensive experiments using multiple real-world datasets. The results demonstrate a significant improvement over state-of-the-art approaches, with a 36% increase in the long-term social welfare of FL, accompanied by a 37% reduction in payment.

3.1 Introduction

Federated learning (FL) has gained considerable attention due to its ability to train a global model among distributed clients and a server without the need to aggregate raw data. This approach is highly regarded for its privacy-preserving and bandwidth-efficient characteristics. However, the conventional FL scheme assumes that all clients will willingly participate in the FL training process, which is not realistic given the computational and communication resources required for local training and gradient transmission. Without appropriate incentives, clients are reluctant to contribute their energy and data, thus limiting the practical application of FL.

Some existing studies have focused solely on maximizing server-side utility and have employed reward designs based on resource conditions [108, 107, 104] or reputation



(a) The utility of multiple parties in a single round with different mechanisms, where the right side shows the social welfare of the whole FL system (the utility sum of all parties).

(b) The multiple round social welfare with different auction mechanisms, where the right side shows the total social welfare of the whole process (social welfare sum of all round).

Figure 3.1: The pre-experiment results with different mechanism to illustrate why FL needs a long-term social welfare maximization mechanism.

credit [43] to attract client participation. However, as FL is a multi-party collaborative system, it is undesirable to adopt mechanisms that selfishly maximize server-side utility, as this can discourage client involvement. Economic studies have shown that constructing a sustainable development environment is crucial for facilitating the joint development of the entire system and attracting a steady stream of clients [36, 65, 3]. Maximizing social welfare is one of the vital economic properties in achieving this objective. Our pre-experiments in Fig.3.1(a) shows the utility of multiple parties and the social welfare of the system in a single round with different incentive mechanisms. Evidently, the utility offered by these methods is primarily concentrated on the server-side, resulting in insufficient utility for all clients to effectively incentivize their participation. In contrast, our proposed approach enables the server to motivate clients significantly by sacrificing a portion of its own utility. This strategy ultimately leads to higher social welfare within the system. However, it should be noted that the majority of widely used auction mechanisms only satisfy a subset of the essential economic properties necessary for maximizing social welfare. Consequently, these mechanisms are inadequate in ensuring a sustainable FL environment.

Moreover, given that the training process of federated learning (FL) encompasses a series of long-term successive rounds, the utilization of online auctions has been deemed suitable for modeling incentive mechanisms that aim to maximize social welfare [81, 16, 103]. Nonetheless, existing online auction approaches typically involve decomposing the overall process into a sequence of sub-problems within each individual round, subsequently optimizing them independently. Applying such online auctions directly to FL disrupts the inherent long-term continuous correlation between the rounds. Our preliminary experimental findings, as depicted in Figure 3.1(b), reveal that this decomposition approach results in a substantial degradation of total social welfare when viewed from a long-term perspective. Notably, the long-term mechanism fails to achieve optimality in specific rounds, such as rounds 5 and 6. However, through consideration of the inter-round correlation in FL, the mechanism can appropriately sacrifice a portion of the current social welfare to facilitate subsequent rounds, thereby guaranteeing optimality throughout the entire long-term process.

In order to bridge the existing gaps in economic properties and establish a sustainable environment for the current federated learning (FL) system, the development of a mechanism that ensures all essential economic properties, including incentive compatibility (IC) and individual rationality (IR), while maximizing social welfare is crucial. Among various auction mechanisms, the Vickrey–Clarke–Groves (VCG) auction stands out as the sole mechanism capable of simultaneously satisfying all aforementioned requirements [48, 90]. Consequently, the VCG auction is adopted as a guiding framework for the design of our mechanism. Furthermore, considering the long-term optimization needs of the FL process, we extend the traditional VCG economic properties to encompass a long-term perspective. This extension involves the incorporation of long-term social welfare, long-term incentive compatibility, and long-term individual rationality, which are vital considerations for achieving optimal outcomes in the FL system.

Nonetheless, auction mechanisms encounter a significant challenge that must be addressed. This challenge stems from the privacy protection measures inherent in federated learning (FL), whereby clients remain unwilling to disclose their private information to the server, including computational overhead, communication overhead, and data-related information. Of greater importance is the fact that the quality of FL's global model relies on the aggregation of local models, which entails a complex and dynamic process due to the black-box nature of the FL model, typically implemented as a deep neural network. Consequently, formulating an accurate system model that captures the accuracy change process of the FL global model is unattainable. These inherent difficulties render all existing auction mechanisms, including the VCG auction we have employed, incapable of accurately modeling the FL training process.

In order to tackle the aforementioned challenges related to long-term optimization, black-box models, privacy protection, and other factors, this chapter proposes the utilization of deep reinforcement learning (DRL) as a solution. DRL offers the advantage of acquiring the optimal strategy through a model-free and experience-driven approach. To this end, we employ an experience-driven DRL algorithm to design a long-term online Vickrey-Clarke-Groves (VCG) auction mechanism specifically tailored for federated learning (FL). In contrast to the traditional approach of independent optimization through single-round decomposition, our proposed mechanism directly focuses on optimizing the long-term social welfare of the online auction. This is achieved by leveraging the environment and its inherent dynamics, without relying on any prior knowledge of the underlying system.

Moreover, in our utilization of the Vickrey-Clarke-Groves (VCG) mechanism, we have noticed that a substantial number of deep reinforcement learning (DRL) agents must be trained in order to acquire the optimal strategies for various sub-auctions. Notably, the number of agents required is directly proportional to the scale of the participants, resulting in a significant consumption of system resources. To address this particular challenge, we leverage the inherent similarity across the different sub-auctions within the environment. Consequently, we propose a parameter-based knowledge transfer scheme that enables the sharing of learned strategies among the DRL agents. This scheme proves to be highly effective in reducing the training overhead, ultimately alleviating the resource demands associated with training a large number of agents.

The contributions of this chapter are highlighted as follows:

- In order to establish a sustainable development environment for federated learning (FL), we extend the essential economic properties to encompass the long-term training process of FL. This extension is formulated as a long-term online Vickrey-Clarke-Groves (VCG) auction, which aims to maximize social welfare while maintaining economic desiderata.

- To overcome the challenges associated with accurately modeling the FL process, attributed to privacy protection measures and the black-box nature of deep neural networks (DNNs), we develop a reinforcement framework that is both experience-driven and model-free. This framework enables us to exploit an optimal strategy without relying on explicit system modeling. Furthermore, we adopt a parameter-based knowledge transfer technique to facilitate the sharing of learned strategies among deep reinforcement learning (DRL) agents. By doing so, we are able to significantly reduce the extensive training overhead typically associated with the VCG payment rules.

- The theoretical analysis of our proposed mechanism establishes its long-term economic properties. Additionally, we conduct extensive experiments on different datasets, including MNIST, Fashion-MNIST, and CIFAR-10, to evaluate the performance of our mechanism. The experimental results demonstrate that our approach achieves a 36% increase in long-term social welfare while reducing payment by 37% compared to the current baselines.

3.2 Related Work

3.2.1 AI-Empowered Auction

Auction theory has emerged as one of the most widely employed approaches for designing incentive mechanisms, finding applications in numerous domains. However, traditional auction theory has long been confronted with inherent challenges related to computational complexity. To address these challenges and attain optimal auction designs, researchers have explored the integration of various artificial intelligence (AI) techniques into the auction design process. For instance, Dutting *et al.* [21] utilized a multi-layer neural network (NN) to encode auction mechanisms and learned the optimal auction design through iterative sampling. Building upon this work, Feng *et al.*[23] extended this approach to address the budget-constrained bidder problem, while Golowich *et al.* [32] applied it to solve the facility location problem. Furthermore, Luong *er al.* [58] constructed a multi-layer NN framework based on the optimal analytical solution of auctions, demonstrating the substantial advantages of employing deep learning approaches to derive optimal auction designs. However, these existing works have primarily focused on utilizing deep learning technology to design optimal auctions for single-round scenarios, rendering them unsuitable for addressing long-term online auction scenarios with multiple rounds. In contrast, the objective of this chapter is to employ deep reinforcement learning (DRL) approaches to design a long-term optimal online auction mechanism specifically tailored for federated learning (FL). By leveraging the capabilities of DRL, our proposed mechanism aims to overcome the challenges associated with the prolonged and dynamic nature of FL training, enabling the development of an effective long-term online auction mechanism.

3.2.2 Incentive Mechanism for FL

The design of incentive mechanisms in the field of federated learning (FL) has been a subject of extensive research from various perspectives. Initially, some studies concentrated on maximizing the utility at the server-side through different approaches. For example, the implementation of reputation-based mechanisms aimed to encourage the participation of trustworthy edge nodes in the training process [98]. Additionally, incentive mechanisms formulated with Stackelberg game models were explored to enhance communication efficiency in FL [74, 45]. Taking a step further, recent studies have recognized the significance of incorporating economic properties into incentive mechanisms. These studies have embraced auction-based solutions as a means of optimizing the overall social welfare of the system. For example, the work of Zeng *et al.* introduced the "Fmore" framework, which utilizes auction modeling to allocate multi-dimensional resources in the context of FL training [106]. Similarly, Jiao *et al.* proposed an automated auction framework for the FL services market [41]. Furthermore, Yuan *et al.* developed an online auction mechanism to facilitate an efficient FL client selection strategy while considering energy constraints [105]. These studies exemplify the application of auction-based approaches in FL, aiming to enhance the efficiency and effectiveness of resource allocation and client selection while promoting overall social welfare. However, these existing works fail to address several critical challenges that arise in the context of federated learning (FL): **Firstly**, while optimizing social welfare is an essential economic property for establishing a sustainable development environment, it is crucial to consider other economic properties such as incentive compatibility and individual rationality. Unfortunately, most widely used auction mechanisms are unable to simultaneously satisfy all of these economic properties. The absence of these economic properties poses a significant obstacle to ensuring a sustainable mechanism for long-term FL training. **Secondly**, the system modeling approach adopted by current online auction methods decomposes the entire FL training process into independent sub-problems for each round, thereby disregarding the long-term successive correlation that exists between FL rounds. **Thirdly**, the inherent challenges associated with privacy protection and the black-box nature of deep neural network (DNN) models in FL render current auction methods ineffective. These methods are unable to provide precise system modeling for the dynamic FL training process. Consequently, alternative approaches are required to overcome these limitations and address the privacy and modeling challenges inherent in FL.

3.3 Preliminaries and Definitions

In this section, we will introduce some parameter definitions for both FL and DRL and provide the corresponding symbols used in later sections.

3.3.1 Federated Learning

Federated learning operates as a distributed machine learning framework, comprising a server and a collection of clients denoted by $\mathcal{N} = 1, \cdots, N$. Each client i possesses a local dataset $\mathcal{D}_i$, which consists of its own data samples represented as $\{\boldsymbol{x}_j, y_j\}_j \in \mathcal{D}_i$. Collaborative training takes place on a shared global model, involving the optimization of a loss function f.

In the t-th round, each client i initially receives a global model denoted as $\boldsymbol{\omega}^t$ from the server. Subsequently, the client engages in training using its local dataset $\mathcal{D}_i$ to compute its local loss, denoted as $F_i(\boldsymbol{\omega}^t)$. Specifically, the local loss is determined as follows:

$$F_i(\boldsymbol{\omega}^t) = \frac{1}{d_i} \sum_{j \in \mathcal{D}_i} f(j, \boldsymbol{\omega}^t), \tag{3.1}$$

where d_i represents the size of client i's local dataset ($d_i = |\mathcal{D}_i|$), and $f(j, \boldsymbol{\omega}^t)$ corresponds to the loss of the model $\boldsymbol{\omega}^t$ on the training sample $j \in \mathcal{D}_i$. Subsequently, the local loss is utilized to update the model through the stochastic gradient descent (SGD) approach, given by the equation

$$\boldsymbol{\omega}_i^t = \boldsymbol{\omega}^t - \eta_i \frac{\partial F_i(\boldsymbol{\omega}^t)}{\partial \boldsymbol{\omega}^t}, \tag{3.2}$$

where η_i denotes the local learning rate, and $\boldsymbol{\omega}_i^t$ represents the updated local model of client i. Following the local model updates, each client proceeds to transmit its updated local model $\boldsymbol{\omega}_i^t$ to the server for model aggregation. This aggregation process is expressed as

$$\boldsymbol{\omega}^{t+1} = \sum_{i=1}^{N} \frac{d_i}{d} \boldsymbol{\omega}_i^t, \tag{3.3}$$

where $d = \sum_{i=1}^{N} d_i$ denotes the total size of the global dataset. Finally, the server transmits the newly aggregated global model $\boldsymbol{\omega}^{t+1}$ back to the clients as the starting point for the subsequent round, denoted as $t+1$. The aforementioned process is repeated iteratively until the model converges or satisfies the specified requirements.

The server typically possesses its own validation dataset denoted as $\mathcal{D}_v$, and the objective is to identify the optimal global model $\boldsymbol{\omega}^*$ that minimizes the function $F(\boldsymbol{\omega})$. Specifically, the optimal global model is obtained by solving the optimization problem:

$$\boldsymbol{\omega}^* = \arg\min_{\boldsymbol{\omega}} F(\boldsymbol{\omega}), \tag{3.4}$$

where $F(\boldsymbol{\omega})$ is defined as the average loss over the validation dataset $\mathcal{D}_v$, computed as:

$$F(\boldsymbol{\omega}) = \frac{1}{|\mathcal{D}_v|} \sum_{j \in \mathcal{D}_v} f(j, \boldsymbol{\omega}). \tag{3.5}$$

In general, as the loss $F(\boldsymbol{\omega})$ decreases, the model validation accuracy $A(\boldsymbol{\omega})$ gradually improves, where the model validation accuracy $A(\boldsymbol{\omega})$ is defined by:

$$A(\boldsymbol{\omega}) = \frac{\sum_{j\in\mathcal{D}_v} 1\left(y_j = \hat{y}_j(\boldsymbol{\omega})\right)}{|\mathcal{D}_v|} \tag{3.6}$$

Consequently, we employ the model validation accuracy $A(\boldsymbol{\omega})$ on the validation dataset $\mathcal{D}_v$ as the performance metric to evaluate the model's effectiveness in the subsequent analysis.

3.3.2 Deep Reinforcement Learning

In a conventional deep reinforcement learning (DRL) framework, an agent engages in an iterative interaction with the environment, denoted as $\boldsymbol{E}$, in order to evolve its decision-making policy. At each discrete timestep t, the agent actively observes the current state of the environment, denoted as $s_t \in \boldsymbol{S}$, and subsequently takes an action $a_t \in \boldsymbol{A}$ based on its policy $\pi : \boldsymbol{S} \rightarrow \boldsymbol{A}$. The agent is then rewarded with a scalar value $r_t = r(s_t, a_t) \in \boldsymbol{R}$ from the environment and transitions to the subsequent state s_{t+1}, as dictated by a probabilistic state transition function $P : \boldsymbol{S} \times \boldsymbol{A} \rightarrow \boldsymbol{S}$. The primary objective of the agent is to learn an optimal policy π^* that maximizes the expected cumulative reward $R = \sum_{t=0}^{T} \gamma^t r_t$, where $\gamma \in (0, 1]$ denotes the discount factor. The cumulative reward is computed over a finite time horizon T, and the discount factor ensures that future rewards are given less weight compared to immediate rewards.

In the context of deep reinforcement learning (DRL), the agent aims to acquire the mapping relationship between the current state and the available actions to maximize the cumulative reward. However, attempting to learn this mapping for all possible state-action combinations is often impractical due to the large number of such combinations. Consequently, function approximation techniques are commonly employed to parameterize the agent's policy $\pi_\theta(a|s)$ using a parameter vector θ, which has a significantly smaller dimensionality compared to the space of all possible state-action pairs. Various methods exist for constructing this approximator, with deep neural networks (DNNs) being the most widely utilized approach. By employing a DNN, the DRL agent can effectively learn and represent the policy $\pi_\theta(a|s)$. Subsequently, the agent updates the parameter vector θ to adapt and improve its policy based on the observed quadruple $\{s_t, a_t, r_t, s_{t+1}\}$.

3.4 System Model and Problem Formulation

3.4.1 Auction-based System Model

In an auction-based federated learning (FL) system, the overall process unfolds over a series of rounds denoted as $1, 2, \cdots, K$, where K represents the total number of rounds required for global model convergence. Within each round $k \in [1, K]$, every client i initially submits its bidding price b_i^k per local epoch to the server, based on its corresponding true value v_i^k (see Fig. 3.2-❶). Subsequently, the server determines a strategy $\boldsymbol{\tau}^k = \{\tau_1^k, \cdots, \tau_N^k\}$, which specifies the number of local epochs allocated to each client in the k-th round (see Fig. 3.2-❷). Here, τ_i^k denotes the number of local epochs assigned to client i during the k-th round. Following the local training process carried out by the clients and the subsequent global model aggregation performed by the server (see Fig. 3.2-❸), the server proceeds to calculate the payment p_i^k that is to be provided to each client i in the k-th round (see Fig. 3.2-❹). For a more comprehensive visual representation of the auction-based FL model, please refer to Fig. 3.2. The subsequent sections discuss the detailed auction process for each step within the k-th FL communication round.

During the initial step, each client $i \in \mathcal{N}$ within the set of clients $\mathcal{N}$ provides a bid b_i^k to participate in the federated learning (FL) training during the k-th round. The bid is determined based on the client's true value v_i^k. The true value of each client represents an estimation of the resources they may consume as a result of their involvement in the FL training process. It is important to note that the true value varies significantly

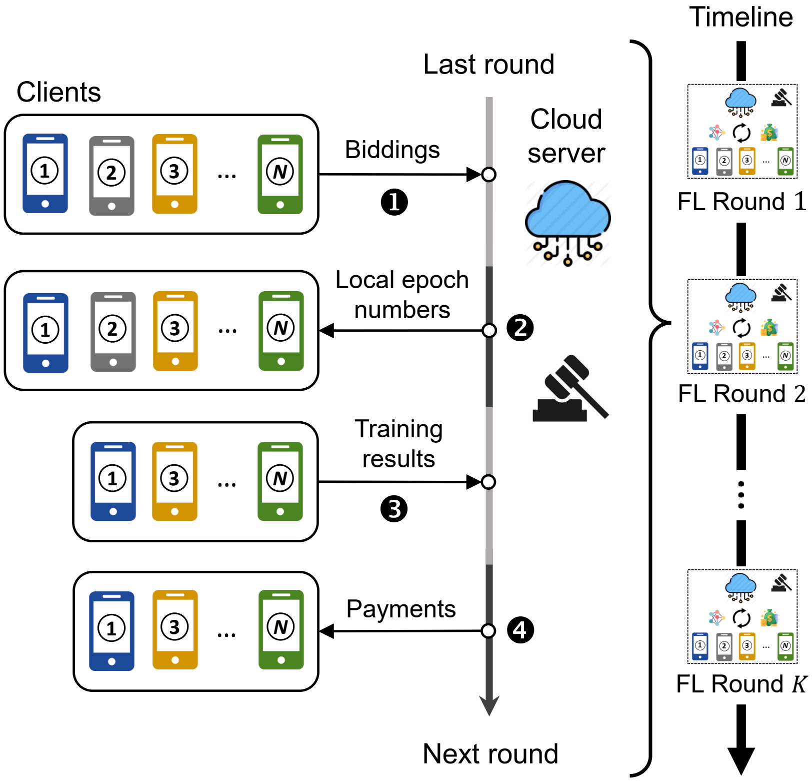


Figure 3.2: Illustration of online auction-based FL.

among clients due to factors such as heterogeneity in hardware configurations and non-IID (non-independent and identically distributed) data distribution. The true value of a client comprises various components, which are as follows:

1) Computation Energy Consumption. Each client i possesses a dataset $\mathcal{D}_i$ consisting of data samples that are uniform in type and volume, such as images with a fixed number of bits. Consequently, the number of CPU cycles required for client i to process a single training data sample is denoted by c, which remains constant across all clients and can be determined beforehand. Let β_i represent the effective capacitance coefficient of client i's computation chip-set, and δ_i denote the CPU-cycle frequency employed by client i during the execution of the federated learning task. Notably, due to the inherent hardware heterogeneity among clients, the values of β_i and δ_i differ across individual clients i. Recalling that $d_i = |\mathcal{D}_i|$ denotes the size of the local dataset held by client i, we adopt the widely accepted system energy model [12] to describe the computation energy consumption incurred by client i per local epoch. This energy consumption is represented by the equation:

$$e_i^{comp} = \beta_i c d_i \delta_i^2. \tag{3.7}$$

2) Communication Energy Consumption. After completing the local training phase, each client i is required to transmit its local model to the server using a wireless communication network, such as WiFi. The energy consumed during this communication process can be estimated using the widely accepted system energy model [12].

Let ξ represent the size of the local model, which remains consistent among all clients due to the identical model structure. Additionally, B_i^k denotes the network bandwidth available to client i during the k-th round. With these parameters, we can calculate the communication time for client i in the k-th round as $T_{i,k}^{comm} = \xi / B_i^k$. It is important to note that the local model size ξ is the same across all clients.

The communication energy consumption of client i in the k-th round can be calculated using the equation:

$$E_{i,k}^{comm} = \epsilon_i T_{i,k}^{comm} = \frac{\epsilon_i \xi}{B_i^k}, \tag{3.8}$$

Here, ϵ_i represents the unit energy consumption of client i when uploading the local model, and its specific value is determined by the hardware configuration of the respective clients.

3) Data Usage. Apart from the aforementioned energy resource consumption, as evidenced by several prior studies [27, 9, 86], the local dataset owned by each client is also a crucial resource for federated learning (FL) training. This dataset possesses inherent data value and should be appropriately accounted for in terms of its usage. Nevertheless, owing to the non-IID (non-identically distributed) data profiles among clients, the data value associated with each client is distinct and subject to dynamic changes based on the FL model requirements for the current FL round. Each round of federated learning introduces the need for different combinations of data to enhance the performance of the FL model. Consequently, this dynamic requirement leads to varying demands for each client's dataset. In simpler terms, if a client is frequently selected by the server (assigned a higher number of local epochs), their data will contribute more significantly to the server's accuracy improvement. This implies that the client possesses a higher data value to the server. This fluctuating data value can be aptly depicted using the economic model of supply and demand [24], where the client with a local dataset represents the supply-side, while the server represents the demand-side. To incorporate the dynamics of data value within the market, we can apply principles from the field of marketing [7]. Let's assume that the value of data owned by client i in the k-th round is denoted as D_i^k. Similar to the approach described in the context of housing (reference omitted), we can describe the dynamics of data value in the market using the equation $D_i^k = M(D_i^{k-1}, \tau_i^{k-1})$, where M is a Markovian function that captures the influence of demand on the value. We have implemented the function M by comparing the current epoch number τ_i^k with the historical average epoch number $\bar{\tau}_i$ assigned to client i in previous training rounds. This average epoch number can be interpreted as the server's level of demand for the local dataset of client i. If the current epoch number τ_i^k determined by the server is greater than the historical average $\bar{\tau}_i$, it indicates a favorable condition for the supply-side in the market. In such cases, client i will consider increasing the data value by a factor of $(\tau_i^k - \bar{\tau}_i)/\bar{\tau}_i$. Conversely, if the current epoch number is lower than the historical average, it implies a less favorable supply-side market, and client i may adjust its data value accordingly.

To provide bidding, each client i needs to follow a two-step process, given that its local epoch number τ_i^k for the current round k is determined by the server at a later stage and is unknown beforehand. The steps involve: 1) Estimating the true value of the client for the current round. 2) Providing the bidding based on the estimated value. The estimated true value of client i in the k-th round can be represented by a function that incorporates its historical average epoch number $\bar{\tau}_i^k$. This function is given by:

$$v_i^k(\bar{\tau}_i^k) = \mu_1 \cdot (e_i^{comp} \times \bar{\tau} i^k) + \mu_2 \cdot Ei, k^{comm} + \mu_3 \cdot D_i^k, \tag{3.9}$$

Here, μ_1, μ_2, and μ_3 are hyperparameters that allow for preference adjustment. The computation energy consumption e_i^{comp} is evaluated for each local epoch, while the communication energy consumption $E_{i,k}^{comm}$ and data value D_i^k are evaluated for the entire round. This is because each local training round only requires data usage and model upload once.

Client i may exhibit selfish and rational behavior, leading to the possibility of raising a bidding price b_i^k that differs from its true value v_i^k, with the intention of maximizing payment from the server (Fig. 3.2-❶). Consequently, the bidding b_i^k provided by client i in the k-th round, based on its estimated true value v_i^k, can be represented as:

$$b_i^k(\bar{\tau}_i^k) = v_i^k(\bar{\tau}_i^k) + N(\mu, \sigma^2), \tag{3.10}$$

Here, $N(\mu, \sigma^2)$ represents Gaussian noise with a mean (μ) of 0 and a variance (σ^2) of 1. This noise factor accounts for the deviation from the true value and reflects the strategic behavior of client i in setting its bidding price.

Subsequently, the server can determine the local epoch number strategy $\boldsymbol{\tau}^k = \tau_1^k, \cdots, \tau_N^k$ for all clients as the output of the current deep reinforcement learning (DRL) strategy network (Fig. 3.2-❷). The input to the network is the clients' bidding. It is worth noting that the server has the authority to reject the participation of certain clients by assigning a local epoch number of zero to them. Additionally, clients have the option to voluntarily exit the current federated learning (FL) round by intentionally overbidding their true values.

Following the determination of the local epoch number strategy $\boldsymbol{\tau}^k$ by the server, clients will execute the corresponding federated learning training task based on the received local epoch number τ_i^k (Fig. 3.2-❸). Subsequently, they will upload the training results back to the server.

During this process, the client data value D_i^k will be updated according to the local epoch number τ_i^k as $D_i^{k+1} = M(D_i^k, \tau_i^k)$.

Finally, the server will distribute the respective payment p_i^k to each client (Fig. 3.2-❹) in accordance with the VCG (Vickrey-Clarke-Groves) payment rules, which will be further defined in Section 3.5.3.

After the auction process of the current round is completed, both the server and clients will calculate their own utility and update their strategies for the next round. On the client side, the utility of client i in the k-th round can be computed based on the payment p_i^k received from the server, using the following equation:

$$U_i^k(\tau_i^k) = p_i^k - v_i^k(\tau_i^k) = p_i^k - (\mu_1 e_i^{comp} \tau_i^k + \mu_2 E_{i,k}^{comm} + \mu_3 D_i^k). \tag{3.11}$$

Here, $U_i^k(\tau_i^k)$ represents the utility of client i in the k-th round, which is the difference between the payment received and the estimated true value based on the local epoch number τ_i^k.

Following the usual convention in economics, we assume that clients are individually rational, meaning that client i aims to maximize its long-term utility over the entire online auction. Consequently, the overall long-term utility for client i is given by:

$$U_i = \sum_{k=1}^{K} U_i^k, \tag{3.12}$$

where K represents the total number of rounds in the auction process.

On the server side, after performing the global aggregation of all the uploaded local models from clients, the server can obtain the global federated learning (FL) model performance $A(\cdot)$ for the current k-th round by evaluating it on its own validation dataset $\mathcal{D}_v$, as described in Equation (3.4). After obtaining the global FL model performance, the server proceeds to calculate the payment p_i^k that will be distributed to each client i. This payment calculation is based on the Vickrey-Clarke-Groves (VCG) payment rules, which are defined in Section 3.5.3. The VCG payment rules are designed to ensure efficiency and truthfulness in the auction mechanism, taking into account the individual contributions of each client and their impact on the overall FL model performance. The utility of the server in the k-th round can be computed as follows:

$$U_s^k(\boldsymbol{\tau}^k) = \lambda \cdot \Delta A^k(\boldsymbol{\tau}^k) - \sum_{i \in N} p_i^k, \tag{3.13}$$

Here, $\Delta A^k(\boldsymbol{\tau}^k) = A(\omega^k) - A(\omega^{k-1})$ represents the model accuracy increment of the federated learning (FL) task after applying the local epoch number strategy $\boldsymbol{\tau}^k$ in the k-th round. The term λ is a non-negative parameter that allows for adjusting the server's preference. A higher value of λ indicates that the server places greater importance on maximizing the model accuracy, while a lower value of λ indicates a relatively higher weight on other factors. The utility of the server is calculated as the difference between the model accuracy increment (weighted by λ) and the sum of the payments distributed to the clients. This formulation allows the server to balance the improvement in model accuracy with the cost associated with compensating the clients.

Indeed, the decision of the local epoch number strategy $\boldsymbol{\tau}^k$ has two significant impacts on the federated learning (FL) system. Firstly, since the clients' data profiles are non-IID (i.e., each client possesses a unique subset of data), the choice of $\boldsymbol{\tau}^k$ affects the accuracy performance of the FL model. By selecting different combinations of clients to participate in the FL training, the model's performance can vary. This problem is commonly known as the client selection problem in FL. Making an optimal decision for $\boldsymbol{\tau}^k$ becomes crucial in order to maximize the accuracy of the FL model. Secondly, the decision of $\boldsymbol{\tau}^k$ also impacts the training cost of the system. This cost includes factors such as computation, communication energy consumption, and data usage. The selection of clients and the allocation of local epoch numbers directly affect these costs. Achieving an optimal trade-off between system performance (model accuracy) and training cost (computation, communication, and data usage) is essential in order to ensure efficient and effective FL. Therefore, finding a balance between system performance and training cost through the decision of the local epoch number strategy in each communication round is a critical challenge in FL. It requires careful consideration of the characteristics of the clients' data, the available resources, and the objectives of the FL system.

3.4.2 Problem Formulation

Prior to presenting the problem formulation, it is pertinent to introduce a set of foundational economic properties derived from traditional auction theory [48]. These properties, widely acknowledged in the field, provide a fundamental framework for the analysis and design of auction mechanisms.

- *Social welfare*: the utility sum of all participants within the system, not the utility of a specific participant.
- *Incentive compatibility*: each participant can obtain the best return if and only if it bids truthfully.
- *Individual rationality*: all participants can obtain non-negative utility.

Given that the online auction for federated learning (FL) comprises a sequence of interconnected auctions, representing the communication rounds of the FL training process, the aforementioned classic economic properties are not inherently suitable for addressing the long-term requirements of our proposed mechanism, as they primarily focus on individual rounds. Consequently, we extend these properties to encompass the overarching objective of our mechanism, which aims to ”maximize the long-term social welfare of a FL system while ensuring the fulfillment of key economic properties over an extended duration.” In the subsequent sections, we provide a comprehensive description of this objective.

1) Maximizing long-term social welfare: In order to evaluate the performance of our proposed online auction mechanism [48, 81, 16, 103], we adopt the concept of long-term social welfare within the federated learning (FL) system. This welfare is defined as the cumulative utility of both the server, denoted as $\sum_{k\in K} U_s^k$, and the individual utility gains of all participating clients, represented by $\sum_{k\in K}\sum_{i\in N} U_i^k(\boldsymbol{\tau}^k)$, across all communication rounds. The social welfare in the k-th round can be computed as follows:

$$\begin{aligned} S^k(\boldsymbol{\tau}^k) &= U_s^k(\boldsymbol{\tau}^k) + \sum_{i\in N} U_i^k(\boldsymbol{\tau}^k) \\ &= \lambda \cdot \Delta A^k(\boldsymbol{\tau}^k) - \sum_{i\in N}(\mu_1 e_i^{comp}\tau_i^k + \mu_2 E_{i,k}^{comm} + \mu_3 D_i^k), \end{aligned} \tag{3.14}$$

where λ, μ_1, μ_2, and μ_3 are weighting factors, $\Delta A^k(\boldsymbol{\tau}^k)$ represents the aggregate accuracy improvement achieved in the k-th round, e_i^{comp} denotes the computational efficiency of client i, τ_i^k represents the computational resource allocated to client i in round k, $E_{i,k}^{comm}$ represents the energy consumption of client i for communication in round k, and D_i^k represents the data discrepancy between client i and the server in round k. The long-term social welfare of the FL system is given by:

$$S(\boldsymbol{\tau}) = \sum_{k=1}^{K} S^k(\boldsymbol{\tau}^k). \tag{3.15}$$

Consequently, to achieve the highest long-term social welfare, it is imperative to determine the optimal strategy for selecting the number of local epochs, denoted as $\boldsymbol{\tau}^*$. This optimal strategy can be obtained by solving the following maximization problem:

$$\begin{aligned} \boldsymbol{\tau}^* &= \arg\max S(\boldsymbol{\tau}) \\ s.t.\quad & U_i(\boldsymbol{\tau}^*) \geq U_i(\tilde{\boldsymbol{\tau}}^*), \forall i. \quad \text{(Long-term IC)} \\ & U_i(\boldsymbol{\tau}^*) \geq 0, \forall i. \quad \text{(Long-term IR)} \end{aligned} \tag{3.16}$$

, where the two constraints here, i.e., Long-term IC and Long-term IR, will be elaborated later.

2) Long-term incentive compatibility: In order to ensure long-term incentive compatibility in the auction mechanism, it is crucial that every client, denoted as i, has the incentive to truthfully disclose their true value, v_i^k, in any given communication round, regardless of the bids submitted by other clients. In other words, if we assume that $\boldsymbol{\tau}^*$ represents the socially optimal strategy when client i bids truthfully and $\tilde{\boldsymbol{\tau}}^*$ represents the strategy when client i bids untruthfully, we can establish the following condition:

$$U_i(\boldsymbol{\tau}^*) \geq U_i(\tilde{\boldsymbol{\tau}}^*), \forall i. \tag{3.17}$$

3) Long-term individual rationality: To attract more potential clients to the FL system, the mechanism needs to satisfy individual rationality which means each client can obtain a non-negative utility when participating in a FL task, i.e.,

$$U_i(\boldsymbol{\tau}^*) \geq 0, \forall i. \tag{3.18}$$

This inequality signifies that client i will obtain at least the same utility or greater when employing the truthful bidding strategy ($\boldsymbol{\tau}^*$) compared to the scenario where they adopt an untruthful bidding strategy ($\tilde{\boldsymbol{\tau}}^*$), regardless of the bids made by other clients.

It is important to note that the extended long-term properties should not be interpreted as an absolute guarantee of their fulfillment in every round of the federated learning (FL) process. Striving for perfect adherence to these properties in every round would be excessively idealistic and essentially unattainable in practice. In reality, the attainment of the long-term properties signifies that both the server and clients may be willing to accept a lower level of utility in specific rounds of FL to optimize the overall utility and effectiveness of the entire FL training process. This recognition acknowledges that achieving the maximum long-term benefits may involve making certain trade-offs and adjustments in utility allocation across different rounds of FL. By adopting such an approach, a more realistic and practical optimization strategy can be pursued that takes into account the overall utility and welfare over the entire duration of FL.

3.5 Mechanism Design

3.5.1 Motivation: why must DRL

The VCG (Vickrey-Clarke-Groves) mechanism [90] is a widely studied auction mechanism that offers a general framework guaranteeing both social welfare maximization and truthfulness. Given its desirable properties, the VCG mechanism is an ideal choice for our auction design. The standard VCG scheme consists of two main steps. First, it involves determining an optimal strategy by solving the social welfare optimization problem, which aligns with the objective expressed in Equation (3.16) in our specific problem setting. Second, the VCG mechanism calculates payments to each client, taking into account the externalities imposed by each client on the others. The payment rule, which will be discussed in detail in Section 3.5.3, ensures that clients are compensated properly. However, employing the standard VCG mechanism for our problem is challenging due to various critical obstacles that hinder the acquisition of the optimal long-term social welfare strategy, $\boldsymbol{\tau}^*$. These challenges pose difficulties in determining the most beneficial allocation of computational resources over multiple rounds, thereby limiting the direct applicability of the standard VCG mechanism to our specific auction design.

1) *Myopic Observation*: The optimal solution to Equation (3.16) requires a long-term optimization approach for social welfare. However, obtaining this solution directly using traditional optimization methods is impractical due to the need for complete knowledge about the entire lifespan of the federated learning (FL) system.

2) *Information Isolation*: To derive the theoretical optimal solution $\boldsymbol{\tau}^*$ in Equation

(3.16), we need to substitute Equations (3.7)-(3.9) into Equation (3.14) to obtain the social welfare $S^k(\boldsymbol{\tau}^k)$ in the k-th round. However, Equations (3.7)-(3.9) incorporate private information (such as $\beta_i, c_i, d_i, \delta_i$) of clients, which cannot be made publicly available due to privacy protection principles in FL.

3) *Model Unknown*: In a FL system, the involvement of neural networks and collaborative training introduces elements that cannot be accurately modeled. For instance, deriving an analytical model of the global performance, denoted as $A(\omega^k)$, is infeasible. Consequently, the final global model performance can only be obtained through actual training processes.

Given the aforementioned challenges, Deep Reinforcement Learning (DRL) emerges as a promising approach as it is an experience-driven method that does not rely on prior knowledge during its training process. Consequently, it is well-suited for addressing the challenges outlined above. Recent research has demonstrated the effectiveness of DRL as a model-free approach, surpassing human-level performance in various complex environments such as Atari games [67] and Dota [5]. This success showcases the suitability of DRL in tackling the challenge of model uncertainty present in the FL system. Furthermore, DRL is capable of optimizing long-term cumulative rewards, enabling the auction mechanism to overcome the myopic observation challenge and focus on the optimization of long-term social welfare. By considering the cumulative rewards over time, DRL allows for a more comprehensive and forward-looking approach to optimizing the FL process.

Furthermore, practical machine learning applications, as shown in previous studies [88], are often deployed in dynamic environments where data patterns change over time. For instance, in a recommendation system for a shopping platform, clients' interests frequently shift. As a result, the global model on the server side in federated learning (FL) needs to be continuously updated to adapt to the dynamic environment. This continuous updating process accumulates substantial historical experience on the server side, creating a favorable condition for the integration of Deep Reinforcement Learning (DRL) into the FL server.

Motivated by these advantages, we propose to combine DRL with the long-term online VCG auction mechanism. By leveraging the historical experience accumulated by the server, DRL can effectively optimize the auction mechanism over time, taking into account the changing dynamics of the FL environment. This integration enables the auction mechanism to adapt and make optimal decisions in response to evolving client preferences and data patterns, leading to improved overall performance and long-term social welfare in FL.

3.5.2 DRL Design

The online auction in the federated learning (FL) setting is structured as a sequence of successive rounds of auctions. This sequential nature makes it well-suited for integration with the Markov Decision Process (MDP) framework of Deep Reinforcement Learning (DRL), as well as the FL training process. Each auction segment corresponds to a state transition process denoted by s_k, a_k, r_k, s_{k+1}, representing the state, action, reward, and next state in the k-th FL training round. In our approach, we employ the Proximal Policy Optimization (PPO) algorithm as our chosen DRL algorithm [79]. PPO is selected due to its ability to address the sensitivity of policy gradient methods to the update step size. If the step size is too small, progress becomes exceedingly slow, while if it is too large, the signal can be overwhelmed by noise or result in catastrophic drops in performance. Although alternative algorithms such as TRPO and ACER have attempted to mitigate these limitations, ACER is comparatively complex as it requires additional off-policy corrections and a replay buffer. On the other hand, TRPO presents challenges in parameter sharing with other algorithms, which hampers subsequent transfer learning in our context. In our DRL model design for the online auction in FL, we define the state, action, and reward components as follows:

State Design: The state of the agent in the k-th FL round, denoted by s_k, is represented by the bidding price vector of all clients for one local epoch. It can be written as $s_k = b_1^k, \cdots, b_i^k, \cdots, b_N^k$, where b_i^k represents the bidding price of client i in the k-th round (as discussed in Section 3.4.1).

Action Design: The action of the agent in the k-th FL training round, denoted by a_k, is defined as the local epoch number vector for each client. It can be represented as $a_k = \tau_1^k, \cdots, \tau_i^k, \cdots, \tau_N^k$, where τ_i^k corresponds to the local epoch number chosen by client i in the k-th round.

Reward Design: The reward for the agent is determined by the social welfare of the system in the k-th FL training round, denoted as $r_k = S^k$. The social welfare S^k is calculated using the formula $\lambda \cdot \Delta A^k - \sum_{i \in N}(\mu_1 e_i^{comp} \tau_i^k + \mu_2 E_{i,k}^{comm} + \mu_3 D_i^k)$, where λ is a weighting factor, ΔA^k represents the change in global model performance from the previous round, e_i^{comp} is the computation cost of client i, $E_{i,k}^{comm}$ denotes the communication cost of client i in the k-th round, and D_i^k represents the data utility loss of client i. Although the agent receives a reward in each round, we adopt a Monte-Carlo update approach where the strategy is updated based on the reward obtained in the last round. This update scheme aims to optimize long-term social welfare.

State Transition: At the onset of the k-th federated learning (FL) training round, all participating clients contribute their bids, denoted as the state s_k, to the server for the ongoing FL task. Upon receiving the state, the server employs its policy $\pi_\theta(a_k|s_k)$ to determine the appropriate local epoch number, referred to as the action a_k, for each client. Subsequently, all clients engage in FL task training based on the specified requirements and subsequently upload their respective local training results to the server. Following this, the server performs global aggregation to obtain the new model, thereby facilitating the calculation of the social welfare, denoted as the reward r_k, for the current round. Finally, all clients receive their respective payments and update their bids in preparation for the subsequent round of the FL task. This results in the emergence of a new state, denoted as s_{k+1}, which serves as the basis for the next FL training round.

3.5.3 VCG-based Payment Rule

In mathematical terms, we can define $S_{\mathcal{N}\setminus i}(\boldsymbol{\tau}^*)$ as the social welfare achieved when applying the optimal global strategy $\boldsymbol{\tau}^*$ but without taking into account the cost of client i. Furthermore, we define $\boldsymbol{\tau}^*_{-i}$ as the optimal individual strategy obtained from a sub-auction. This strategy is derived from an individual environment that excludes client i, as illustrated in Figure 3.3(b). By considering the individual environment without client i, we can assess the optimal strategy for the remaining clients in a scenario where client i is absent.

According to the payment rule of the VCG mechanism, the payment made to client i is determined using the following equation:

$$p_i = S_{\mathcal{N}\setminus i}(\boldsymbol{\tau}^*) - S(\boldsymbol{\tau}^*_{-i}), \tag{3.19}$$

where $p_i = \sum_{k=1}^{K} p_i^k$ is the total payment of client i throughout the FL training. To implement the VCG payment rule, a bookkeeping scheme (as shown in Section 3.4.1) is employed, where the contributions of clients in each round are recorded. In this scheme, each client's payment is determined at the end of the FL training based on the recorded contributions.

Then, we can get the following theorem.

Theorem 1. *The reinforcement online mechanism that produces allocation $\boldsymbol{\tau}$ and payments $\mathbf{p}$, is both incentive-compatible and individual rationality.*

Proof. Based on the payment rule given in (3.19), we can get

$$
\begin{aligned}
& U_i(\boldsymbol{\tau}) = p_i - v_i(\boldsymbol{\tau}) \\
& = p_i - \sum_{k=1}^{K} (\mu_1 e_i^{comp} \tau_i^k + \mu_2 E_{i,k}^{comm} + \mu_3 D_i^k) \\
& = S_{\mathcal{N}\setminus i}(\boldsymbol{\tau}^*) - S(\boldsymbol{\tau}_{-i}^*) - \sum_{k=1}^{K} (\mu_1 e_i^{comp} \tau_i^k + \mu_2 E_{i,k}^{comm} + \mu_3 D_i^k) \\
& \geq S(\boldsymbol{\tau}^*) - S(\boldsymbol{\tau}_{-i}^*) \geq 0
\end{aligned}
\tag{3.20}
$$

Therefore, the individual rationality property of reinforcement online mechanism is satisfied.

Next, to prove the truthfulness of the mechanism, we compare the utility of client i under the truthful bid and an untruthful bid. Suppose that in k-th round, client i submits an untruthful bid b_i^k which is not equal to its true value v_i^k, i.e., $b_i^k \neq v_i^k$, then the optimal strategy becomes $\tilde{\boldsymbol{\tau}}^*$. His utility under untruthful bidding can be calculated by

$$
U_i(\tilde{\boldsymbol{\tau}}^*) = (\widetilde{S}_{\mathcal{N}\setminus i}(\tilde{\boldsymbol{\tau}}^*) - S(\boldsymbol{\tau}_{-i}^*)) - v_i(\boldsymbol{\tau}). \tag{3.21}
$$

Then, the difference of utilities under truthful and untruthful bidding is

$$
\begin{aligned}
U_i(\boldsymbol{\tau}) - U_i(\tilde{\boldsymbol{\tau}}^*) \geq & (S(\boldsymbol{\tau}^*) - S(\boldsymbol{\tau}_{-i}^*)) \\
& - ((\widetilde{S}_{\mathcal{N}\setminus i}(\tilde{\boldsymbol{\tau}}^*) - S(\boldsymbol{\tau}_{-i}^*)) - v_i(\boldsymbol{\tau})) \\
= & S(\boldsymbol{\tau}^*) - (\widetilde{S}_{\mathcal{N}\setminus i}(\tilde{\boldsymbol{\tau}}^*) - v_i(\boldsymbol{\tau})) \\
= & S(\boldsymbol{\tau}^*) - (S_{\mathcal{N}\setminus i}(\tilde{\boldsymbol{\tau}}^*) - v_i(\boldsymbol{\tau})) \\
\geq & S(\boldsymbol{\tau}^*) - S(\tilde{\boldsymbol{\tau}}^*).
\end{aligned}
\tag{3.22}
$$

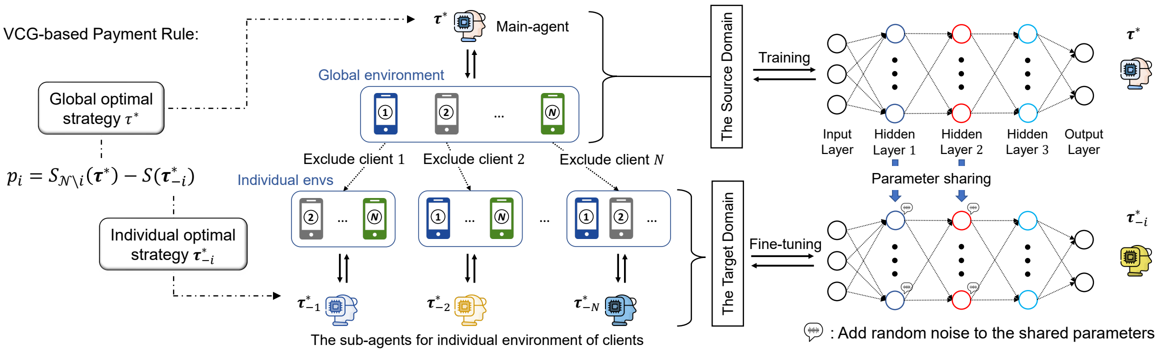
Since $\boldsymbol{\tau}^*$ maximizes the long-term social welfare of FL system, we can obtain $U_i(\boldsymbol{\tau}) - U_i(\tilde{\boldsymbol{\tau}}) \geq 0$, which means client i cannot increase its utility by bidding untruthfully. □

3.5.4 Parameter-based Knowledge Transfer for DRL Training

According to the VCG payment rule presented in equation (3.19) and illustrated in Figure 3.3(a), the computation of payments assigned to each client i entails training a main-agent to acquire the optimal global strategy $\boldsymbol{\tau}^*$ within an environment that incorporates the participation of all clients. This universal global strategy can subsequently be employed to calculate $S_{\mathcal{N}\setminus i}(\boldsymbol{\tau})$ for any client i. Then, to compute $S(\tau^*_{-i})$, a separate agent referred to as a sub-agent must be trained to obtain the optimal individual strategy $\boldsymbol{\tau}^*_{-i}$ within client i's unique environment. This environment consists of all clients except for client i, denoted as $1, \ldots, i-1, i+1, \ldots, N$. Consequently, each distinct client i necessitates the training of an individual sub-agent to derive their respective $\boldsymbol{\tau}^*_{-i}$ using traditional Deep Reinforcement Learning (DRL) methods, given the varying environments across clients. In summary, the computation of payments requires the training of N sub-agents, each dedicated to obtaining the optimal individual strategy $\boldsymbol{\tau}^*_{-i}$ for a specific client. This process becomes more time-consuming as the number of clients increases, scaling linearly with the client count.

Indeed, it is evident that there exist notable similarities between these environments. The global strategy $\boldsymbol{\tau}^*$ is acquired within an environment that encompasses the participation of all clients (referred to as the global environment), while the individual strategy $\boldsymbol{\tau}^*_{-i}$ for client i is obtained within an environment that excludes client i (known as the individual environment). The sole distinction between these two environments lies in the absence of client i, with all other clients remaining identical. In essence, all the individual environments for $\boldsymbol{\tau}^*_{-i,i\in\mathcal{N}}$ can be regarded as subsets of the global environments for $\boldsymbol{\tau}^*$. A visual representation of this relationship is provided in Figure 3.3(b).

Based on the aforementioned similarity between the global and individual environments, we propose the utilization of transfer learning as a technique to expedite the


VCG-based Payment Rule:
Global optimal strategy τ^*
$p_i = S_{\mathcal{N}\setminus i}(\boldsymbol{\tau}^*) - S(\boldsymbol{\tau}^*_{-i})$
Individual optimal strategy $\boldsymbol{\tau}^*_{-i}$
$\boldsymbol{\tau}^*$
Main-agent
Global environment
Exclude client 1
Exclude client 2
Exclude client N
Individual envs
$\boldsymbol{\tau}^*_{-1}$
$\boldsymbol{\tau}^*_{-2}$
$\boldsymbol{\tau}^*_{-N}$
The sub-agents for individual environment of clients
The Source Domain
Training
Input Layer
Hidden Layer 1
Hidden Layer 2
Hidden Layer 3
Output Layer
$\boldsymbol{\tau}^*$
Parameter sharing
The Target Domain
Fine-tuning
$\boldsymbol{\tau}^*_{-i}$
: Add random noise to the shared parameters

(a) The VCG-based payment rule (b) The illustration of environment similarities (c) The illustration of parameter-based knowledge transfer

Figure 3.3: The architecture of parameter-based knowledge transfer for DRL training acceleration to compute the VCG-based payment.

training process of sub-agents in acquiring their optimal strategies. Transfer learning is a methodology that leverages pre-existing knowledge to facilitate the learning of new knowledge [97, 73]. Particularly in the field of machine learning, transfer learning involves applying knowledge gained from a source domain to a distinct but related target domain. In the VCG payment rule, the global environment in which the main-agent obtains the global strategy $\boldsymbol{\tau}^*$ can serve as the source domain, while the individual environments in which sub-agents acquire their individual strategies $\boldsymbol{\tau}^*_{-i,\in\mathcal{N}}$ can be regarded as the target domains. Therefore, by implementing transfer learning, the knowledge acquired by the main-agent from the global environment can be shared with all sub-agents. This enables the sub-agents to benefit from the pre-existing knowledge, leading to a significant reduction in training time compared to the original approach of training each sub-agent from scratch repeatedly.

In DRL, the agent enhances its knowledge by engaging in iterative interactions with the environment. This knowledge is encapsulated within the agent's deep neural network, manifested through various weight parameters. Therefore, we adopt the parameter-based transfer learning approach [87] to facilitate knowledge transfer between agents One straightforward approach for parameter control in transfer learning is to directly share the parameters of the source learner with the target learner. For instance, if we have a neural network model for the source task, we can share (or freeze) most of its layers and only fine-tune the last few layers to create the target network. This parameter sharing technique has been widely employed in network-based transfer learning methods [97, 37, 73, 87]. Furthermore, since the shared parameters have already converged during the previous training, in order to enhance their generalization capabilities and enable adaptation to the new environment, we introduce a small amount of random Gaussian noise denoted as σ to the shared parameters. This noise injection serves to improve the flexibility and adaptability of the parameters when confronted with novel tasks or environments. A detailed representation of our parameter-based knowledge transfer methodology can be found in Figure 3.3(c).

In the final step, our main-agent and sub-agents undergo training through parameter-based knowledge transfer. The main-agent and sub-agents share an identical neural network architecture. The training process is depicted in **Algorithm 1**.

The workflow of parameter-based knowledge transfer for training DRL agents is as follows: First, the network parameters θ of the main-agent are randomly initialized (Line 2). Subsequently, DRL training is conducted on the source domain, which represents the global environment, as shown in Fig 3.3(b). (Line 3). As the training process gradually converges, the network parameters are updated from θ to $\theta(\boldsymbol{\tau}^*)$, enabling the main-agent to acquire the optimal global strategy $\boldsymbol{\tau}^*$ for all clients (Line 4).

For each sub-agent $i \in \mathcal{N}$, a parameter-based knowledge transfer approach is employed to accelerate its initial training. Initially, the network parameters θ_i of sub-agent i are randomly initialized (Line 7). Next, the first two layers of the main-agent's network parameters $\theta(\boldsymbol{\tau}^*)$ are shared with θ_i, effectively replacing the original parameters of θ_i (Line 8). To enhance the generalization performance of these shared parameters, a random Gaussian noise σ is added to them (Line 9). Finally, the sub-agent undergoes a similar training process on the target domain, fine-tuning its parameters θ_i to $\theta(\boldsymbol{\tau}_{-i}^*)$, resulting in the sub-agent i acquiring the optimal individual

Algorithm 1: The workflow of parameter-based knowledge transfer for DRL agent training (same network architecture).

```
1  The training of main-agent:
2  Initialize network parameters $\theta$ of main-agent
3  Perform DRL training on the source domain until convergence
4  The network parameters are updated from $\theta$ to $\theta(\boldsymbol{\tau}^*)$, and the main-agent
     obtains the optimal global strategy $\boldsymbol{\tau}^*$ for all clients
5  The training of $N$ sub-agents:
6  for client $i$ in $1, 2, \cdots, N$ do
7      Initialize the network parameters $\theta_i$ of sub-agent $i$
8      Share the first two layers of main-agent's network parameters $\theta(\boldsymbol{\tau}^*)$ with $\theta_i$
9      Add a random Gaussian noise $\sigma$ to these shared parameters of $\theta_i$
10     Perform DRL training on the target domain until convergence
11     The network parameters are fine-tuned from $\theta_i$ to $\theta(\boldsymbol{\tau}^*_{-i})$, and the sub-agent $i$
         obtains the optimal individual strategy $\boldsymbol{\tau}^*_{-i}$
```

strategy $\boldsymbol{\tau}^*_{-i}$ for client i (Line 10-11).

3.6 Performance Evaluation

3.6.1 Experimental Setup

We utilize the *FedAvg* framework, originally proposed by Google, as the basis for our experiments. These experiments are conducted on various real-world datasets, namely MNIST, Fashion-MNIST, and CIFAR-10. The parameter settings for the neural networks employed in the image classification task for each dataset align with the configurations outlined in [63].

Federated Learning Setup: Regarding the federated learning setup, there are some variations between the datasets. However, certain FL settings remain consistent across all datasets. We evaluate all mechanisms using Non-IID data partitioning, wherein the entire dataset is randomly distributed among the clients, with each client possessing unique data. Different degrees of Non-IID are achieved by adjusting the random seed. In terms of local model training, we employ Stochastic Gradient Descent with a mini-batch size of 20, while the learning rate for local updates is set to 0.01. For energy consumption analysis, we assume that the number of CPU cycles required to process a single sample, denoted as c_i, is 20 cycles/bit. Additionally, the effective capacitance coefficient β_i is randomly distributed within the range of 1×10^{-28} to 2×10^{-28}. To account for the heterogeneity in clients' hardware resources, the CPU-cycle frequency δ_i is randomly distributed within the range of 1 to 2 GHz.

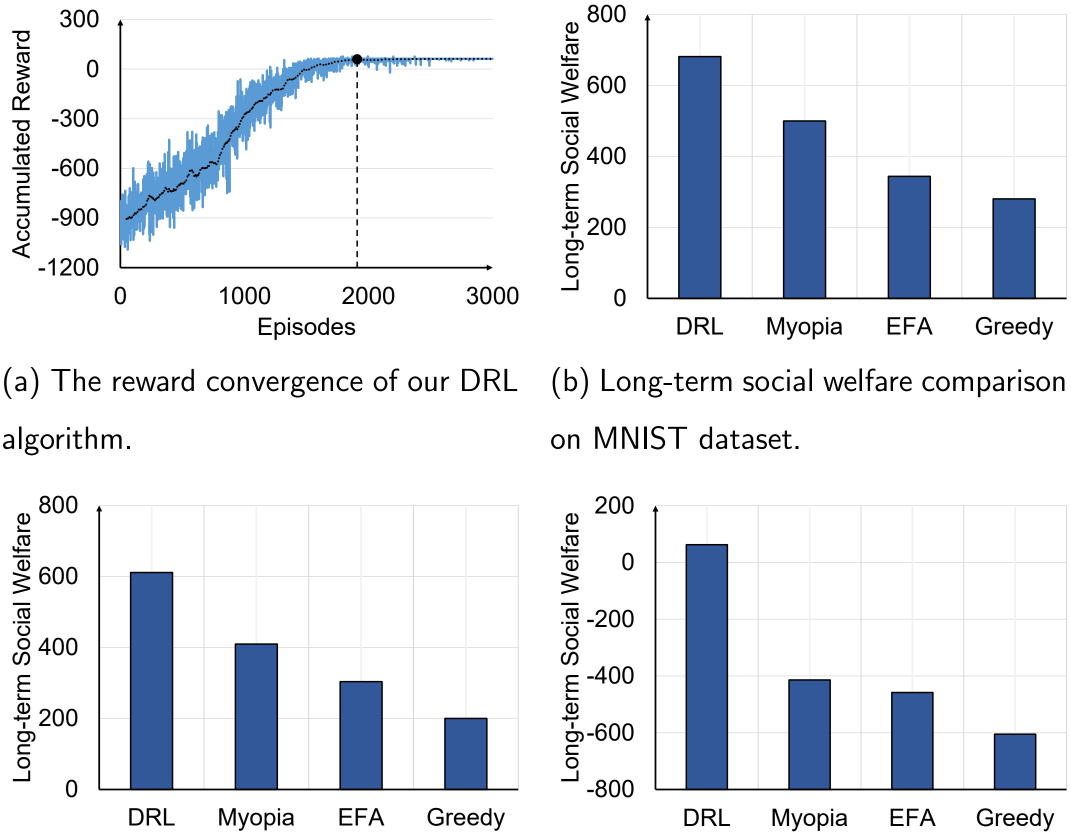
Deep Reinforcement Learning Setup: In the DRL setup, we configure several parameters to facilitate the experience-driven learning process. Specifically, we set the number of episodes, denoted as E, to 3000. Within each episode, the agent takes a fixed number of steps, denoted as K, which is set to 5. Additionally, the learning rates for both the actor and critic networks are set to $lr_a = lr_c = 0.00003$. These learning rates decay by 95% every 40 episodes to facilitate convergence. To optimize the long-term target directly, the update batch size of the agent is set to the same value as the step number K. In terms of reward design, we employ a reward discount factor of $\gamma = 0.95$ to account for future rewards. Furthermore, we introduce a preference adjustment coefficient, denoted as λ, which is set to 1000. In Fig 3.5, a diverse range of results for different λ values is displayed. For the Actor-Critic model utilized in our DRL agent, we adopt the same well-established settings as outlined in [93].

Benchmark mechanism: we assess the effectiveness of the proposed deep reinforcement mechanism by comparing it against several benchmark mechanisms as follows: **1) Myopia**: This mechanism decomposes the problem at hand into a series of one-round problems, akin to the approach used in online auctions in other domains rather than federated learning [81, 16, 103]. Each individual problem is solved optimally using exhaustive search, resulting in a short-term optimal solution. The payment scheme for each round follows the VCG mechanism. **2) Expert-FedAvg (EFA)**: Building upon the FedAvg framework proposed in [63], this mechanism incorporates our historical training experience to manually set the local epoch number for each federated learning communication round. **3) Greedy**: This mechanism adopts a greedy strategy by selecting the strategy with the maximum reward from the experience replay buffer with an 80% probability. With the remaining 20% probability, it generates a random strategy.

3.6.2 Performance Analysis

First, we present the performance of our mechanism in terms of optimizing the long-term social welfare within the FL system, as depicted in Fig. 3.4(a). As the training process of the DRL agent progresses, the long-term social welfare exhibits a consistent upward trend. Notably, the metric reaches convergence at approximately 2000 episodes. This observation suggests that the agent gradually acquires the optimal strategy for the FL online auction over time, resulting in the maximization of social welfare in the long run.

The effectiveness and superiority of our proposed mechanism can be better demonstrated through a comprehensive comparison with other mechanisms, as illustrated in Fig. 3.4. We conducted a series of experiments using real datasets, namely MNIST, Fashion-MNIST, and CIFAR-10. Our deep reinforcement mechanism consistently outperforms all other benchmark mechanisms in terms of long-term social welfare optimization. To further illustrate the advantages of our approach, we focus on the results obtained from the MNIST dataset, as shown in Fig. 3.4(b). While the myopia approach employed in previous works has achieved notable improvements through independent single-round optimization, our method still manages to increase the long-term social welfare by an impressive 36% by incorporating long-term optimization



(a) The reward convergence of our DRL algorithm.

(b) Long-term social welfare comparison on MNIST dataset.

(c) Long-term social welfare comparison on F-MNIST dataset.

(d) Long-term social welfare comparison on CIFAR-10 dataset.

Figure 3.4: The performance of our mechanism and its comparison with other baselines.

considerations. Furthermore, we compare our proposed mechanism with the manually adjusted EFA method, which benefits from the experience accumulated during the historical training process. Although the EFA method surpasses the random sampling-based Greedy method, it falls short when compared against our proposed mechanism. Similar observations can be made when examining the results obtained from the Fashion-MNIST dataset, as depicted in Fig. 3.4(c). Due to the close similarity in dataset features between Fashion-MNIST and MNIST, the performance trends remain consistent, further highlighting the effectiveness of our proposed mechanism. In Fig. 3.4(d), we present the results obtained from the CIFAR-10 dataset. While

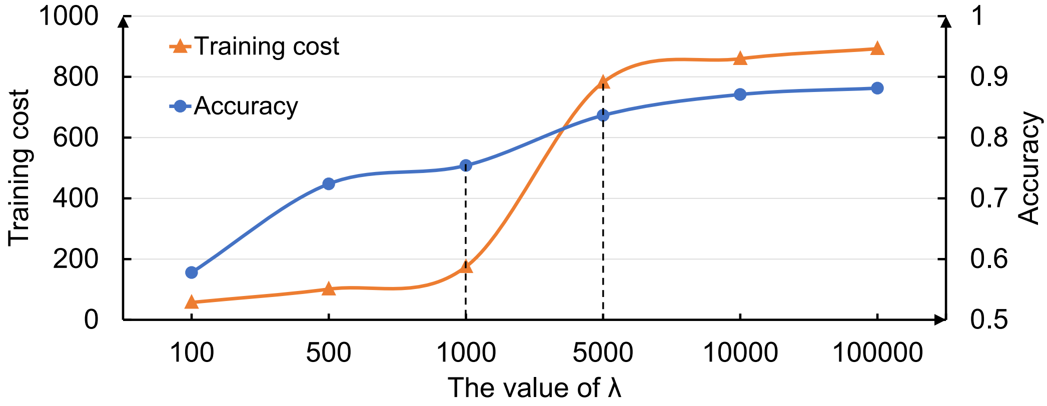
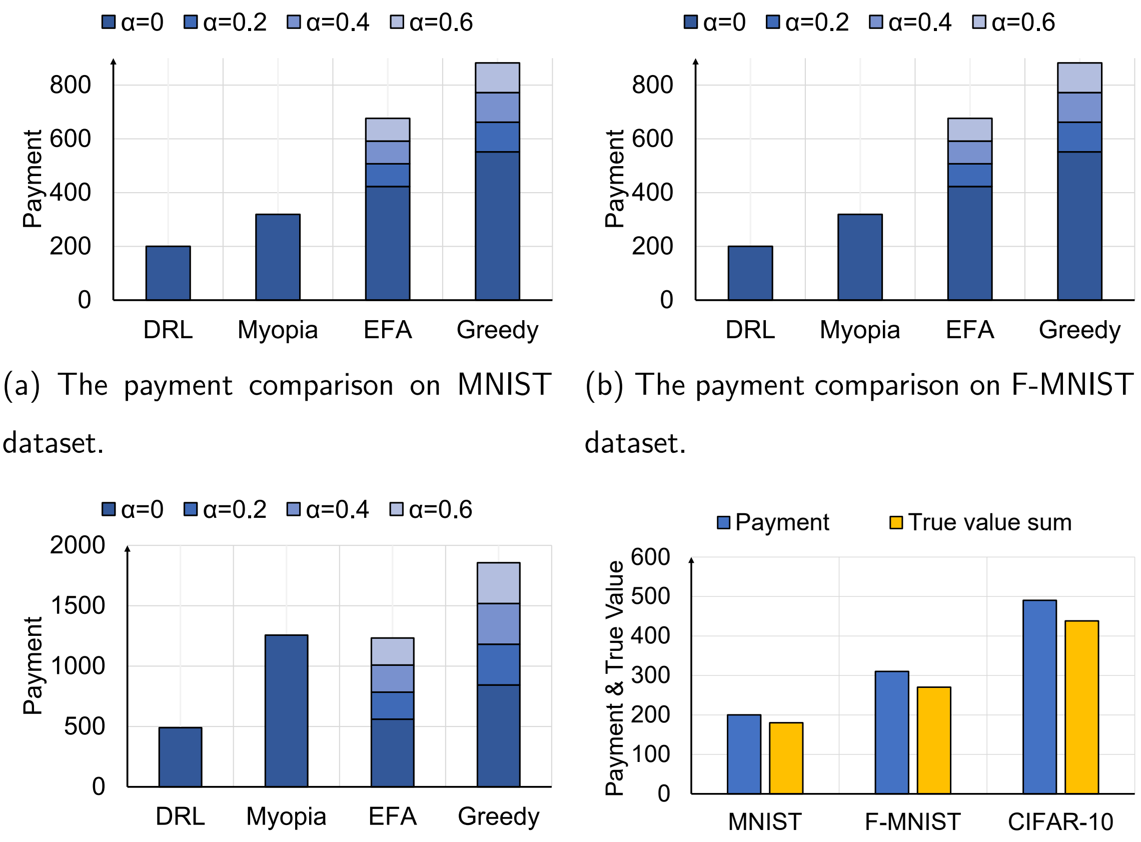


Figure 3.5: The changing trend between accuracy and training cost under different values of λ on MNIST, where λ is a preference adjustment parameter.

the experimental details are largely similar to those of the previous datasets, the higher complexity of CIFAR-10 poses a greater challenge for non-learning benchmark mechanisms to identify optimal strategies. Consequently, our proposed deep reinforcement mechanism achieves a more significant improvement in terms of long-term social welfare optimization on CIFAR-10.

Referring to equations (3.13) and (3.14), it is worth noting that the preference adjustment coefficient λ plays a significant role in balancing the various components of long-term social welfare. A higher value of λ indicates a stronger preference for model accuracy. To investigate the impact of dynamically varying λ on the trade-off between accuracy and training cost, we conducted a series of experiments, as illustrated in Fig.3.5. The experimental results showcase that as the value of λ progressively increases, both the model accuracy and training cost of federated learning (FL) exhibit an upward trend. In particular, the figure indicates two critical junctures at $\lambda = 1000$ and 5000. As λ escalates from 100 to 1000, the accuracy-cost proportion leans towards cost, prompting a transition in our mechanism from an extreme strategy, favoring the selection of the most cost-efficient clients, towards an optimal strategy, which involves a more diversified client selection. Throughout this transition, the associated cost undergoes a modest increase, yet yields a noteworthy



(a) The payment comparison on MNIST dataset.

(b) The payment comparison on F-MNIST dataset.

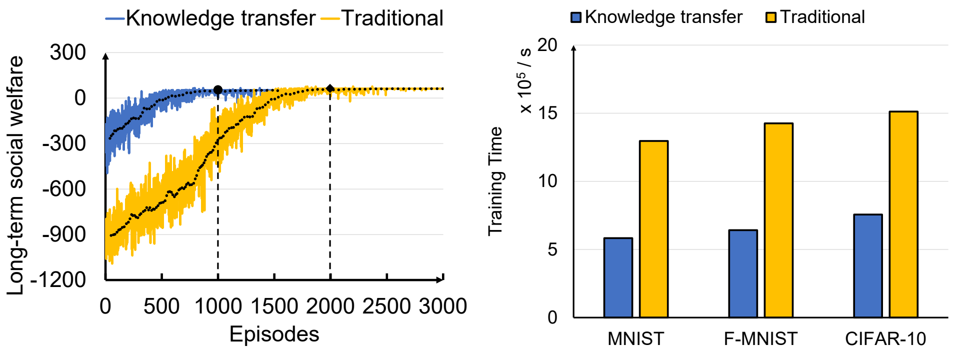
(c) The payment comparison on CIFAR-10 dataset.

(d) The comparison between payment and client true value.

Figure 3.6: The payment comparison on different strategy baselines with different datasets, and the experiment proof of the client long-term individual rationality (IR) property in Eq. (3.18) on all datasets.

enhancement in accuracy. Within the λ range of 1000 to 5000, accuracy and cost proportions align closely, rendering the mechanism sensitive to their trade-off. Notably, accuracy improvements in this range coincide with sharp increases in cost. Upon exceeding $\lambda = 5000$, the mechanism gravitates towards an accuracy-centric approach, albeit with diminishing returns due to convergence saturation. Consequently, further investment in training cost fails to yield significant accuracy improvements.

Fig. 3.6 illustrates the payment comparison across various mechanisms under distinct

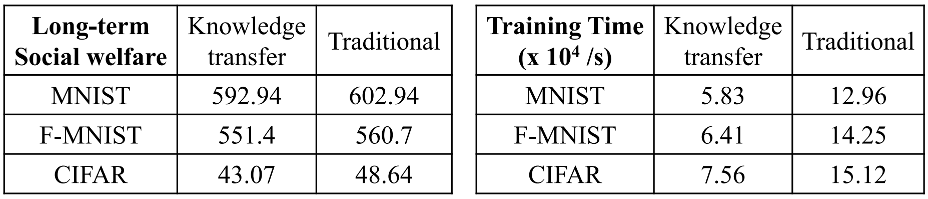


(a) Convergence comparison (b) Training time comparison

Figure 3.7: The performance of parameter-based knowledge transfer approach, where the traditional approach is to retraining DRL agent from scratch.

scenarios. Considering the absence of truthfulness assurance in mechanisms such as EFA and Greedy, wherein clients may misrepresent their costs to maximize payment from the server, a parameter denoted as α is introduced to simulate varying degrees of untruthfulness among clients. Specifically, the payment allotted to each client is computed as the product of $(1 + \alpha)$ and its respective cost. Notably, as reinforcement mechanisms and Myopia ensure truthfulness, their α values are set to 0. The comparative analysis presented in Fig. 3.6 indicates that our mechanism achieves the minimal truthful payment when juxtaposed with Myopia across all datasets, resulting in a reduction of 37% on MNIST, 39% on F-MNIST, and 60% on CIFAR-10, while concurrently enhancing long-term social welfare. Furthermore, to elucidate the concept of long-term individual rationality as delineated in Equation (3.18), we juxtapose the payment against the aggregated true values of clients, corresponding to their costs for participation in FL training. The outcomes depicted in Fig. 3.6(d) affirm our adherence to this property.

Finally, the efficacy of the parameter-based knowledge transfer approach is evidenced through the analysis presented in Figure 3.7. The convergence assessment conducted on CIFAR-10, as depicted in Fig 3.7(a), underscores the comparable efficacy of knowledge transfer relative to conventional methods, alongside a notable 50% reduction in



Long-term Social welfare	Knowledge transfer	Traditional
MNIST	592.94	602.94
F-MNIST	551.4	560.7
CIFAR	43.07	48.64

(a) The comparison of long-term social welfare on different datasets

Training Time (x 10^4 /s)	Knowledge transfer	Traditional
MNIST	5.83	12.96
F-MNIST	6.41	14.25
CIFAR	7.56	15.12

(b) The comparison of training time on different datasets

Figure 3.8: The detailed performance comparison between knowledge transfer and traditional approaches.

required training iterations (from 2000 rounds to 1000 rounds). Fig. 3.7(b) provides a comparative analysis of actual training durations across diverse datasets, revealing a substantial halving of the training period. Furthermore, Fig 3.8 offers a detailed performance juxtaposition of two DRL training strategies across various datasets. It is noteworthy that the knowledge transfer technique incurs only a marginal diminishment in social welfare compared to traditional training-from-scratch methods. This slight loss arises from disparities between the source and target environments, which, despite their similarity, prevent perfect adaptation. However, this diminishment pales in comparison to the considerable reduction in training time achieved through knowledge transfer.

3.7 Remarks

This chapter delves into the incentivization of clients to partake in Federated Learning (FL) training. Recognizing the pivotal importance of long-term social welfare in FL, particularly considering its iterative communication-intensive nature, we propose a reinforcement-empowered online auction mechanism. This mechanism aims to optimize FL's long-term social welfare by integrating Deep Reinforcement Learning (DRL) and Vickrey-Clarke-Groves (VCG) mechanisms. Furthermore, to address the repeated training inherent in VCG payment rules, we introduce a parameter-based knowledge transfer approach to facilitate policy sharing among agents.

The performance evaluation of our proposed mechanism demonstrates a notable 36% enhancement in long-term social welfare and a 37% reduction in payments compared to benchmark mechanisms. Moreover, the knowledge transfer approach significantly mitigates training duration, achieving a 50% reduction while maintaining comparable results.

Additionally, several intriguing avenues warrant further exploration. These include strategies for mitigating potential malicious client behavior in FL environments and devising methodologies for handling multiple heterogeneous FL tasks within the system.

3.8 Discussion

In this chapter, we designed a Vickrey-Clarke-Groves (VCG) auction based incentive mechanism and utilize the Deep Reinforcement Learning (DRL) technique to achieve the long-term optimization for FL system. However, we would like to point out in particular that:

1. The VCG auction is not the only choice for system modeling. In our settings, to satisfy the constraints of three key properties for FL system sustainability, we choose the VCG auction. Once the constraints are changes under other cases, there will be many other potential system modeling methods.

2. DRL technique is not the only solution here. Actually, the traditional Ant Colony Optimization (AOC) can also find the optimal results, but with a huge computational cost. Other method, like distributed optimization, is also a potential solution, where the server and the clients solve the sub-objective respectively.

Chapter 4

Rethinking Personalized Client Collaboration in Federated Learning

Federated Learning (FL) has gained considerable attention recently, as it allows clients to cooperatively train a global machine learning model without sharing raw data. However, its performance can be compromised due to the high heterogeneity in clients' local data distributions, commonly known as Non-IID (non-independent and identically distributed). Moreover, collaboration among highly dissimilar clients exacerbates this performance degradation. Personalized FL seeks to mitigate this by enabling clients to collaborate primarily with others who have similar data characteristics, thereby producing personalized models. We noticed that existing methods for assessing model similarity often do not capture the genuine relevance of client domains. In response, this chapter enhances personalized client collaboration in FL by introducing a metric for domain relevance between clients. Specifically, to facilitate optimal coalition formation, we measure the marginal contributions of client models using coalition game theory, providing a more accurate representation of potential client domain relevance within the FL privacy-preserving framework. Based on this metric, we then adjust each client's coalition membership and implement a personalized FL aggregation algorithm that is robust to Non-IID data domain. We provide a theoretical analysis of the algorithm's convergence and generalization capabilities. Our extensive evaluations on multiple datasets, including MNIST, Fashion-MNIST, CIFAR-10, and CIFAR-100, and under varying Non-IID data distributions (Pathological and Dirichlet), demonstrate that our personalized collaboration approach consistently outperforms contemporary benchmarks in terms of accuracy for individual clients.

4.1 Introduction

With the ongoing advancement of web services, vast amounts of client data are generated daily, that can be immediately exploited through machine learning technology. Indeed, machine learning models, when fueled by such extensive data, have found applications in a myriad of contexts, revolutionizing fields like precision medicine and recommendation systems, to name a few. Within these applications, the precision and generalizability of models are paramount, attributes that are enhanced by training on large data volumes. However, legal constraints, business confidentiality, and individual privacy concerns prevent clients from directly sharing data. This leads to the creation of ”data silos”, limiting the potential enhancement of model capabilities [42].

Federated Learning (FL) is a distributed machine learning approach that enables clients to collaboratively train machine learning models using their local data, without the need to exchange raw data [63]. Instead, by sharing model parameters or intermediate results via a central server, data from different clients can be virtually fused and aligned, enabling clients to collaborate and learn from each other. Importantly, FL strikes a balance between data privacy and data sharing, embodying the principle that while ”data remains unseen, it is still accessible” and ”data stays

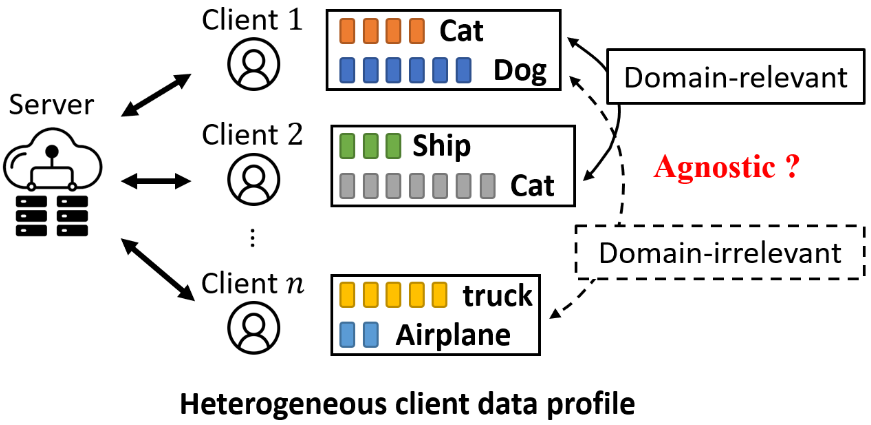


Figure 4.1: Heterogeneous client data domain profiles in an agnostic federated learning system. client 1 and 2 are domain-relevant since they both have '*cat*', while client 1 and n are domain-irrelevant with no label overlap. But these domain relevances are agnostic to clients with the inherent FL privacy protection regulations.

stationary, but models are exchanged."

While Federated Learning (FL) offers potential, its client collaboration often falls short in performance due to the heterogeneous alignment of data domains across clients also known as Non-IID data. Recognizing the needs of the clients, previous studies [80, 110, 39, 18] have investigated the concept of personalized collaboration. Leading methods like FedFomo [110] and FedAMP [39] promote collaboration between client pairs with similar local models. Precisely, FedFomo gauges similarity through loss metrics, while FedAMP utilizes model parameter similarity. These methods operate under the assumption that clients with analogous models share high relevance and should therefore collaborate to enhance performance. However, our experiments indicate that neither loss nor model similarity conclusively indicates domain relevance among clients.

Motivation: We rethink the problem of personalized client collaboration in FL by focusing on measuring domain relevance between clients. To elucidate our motivation, consider the example depicted in Fig. 4.1. While the '*cat*' on client 1 and 2 is domain-

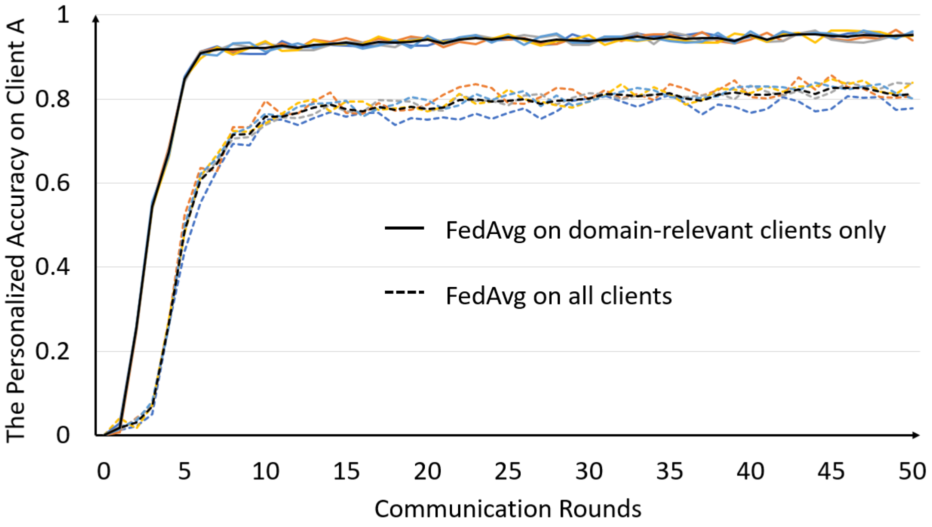


Figure 4.2: The influence of domain relevance on the personalized performance of client A (MNIST). We repeat experiments for 5 times (indicated by different colors) and the black line is their average.

relevant in the data domain, the data domains between client 1 and client n are entirely unrelated. A core insight from our work is that collaboration between domain-relevant clients boosts performance, whereas involving unrelated clients can severely degrade outcomes.

To further certify our above key insight, we conducted a preliminary experiment using the standard FedAvg Algorithm on MNIST with the following settings in Figure 4.2. We configured a setup with a total of 5 clients: A, B, C, D, E, assuming that the personalized task of client A is the even number classification, *i.e.*, $\{0, 2, 4, 6, 8\}$. The label distribution of other clients are: $B : \{0, 2, 4\}$, $C : \{6, 8\}$, $D : \{1, 3, 5\}$ and $E : \{7, 9\}$. It is very clear that class labels owned by client B and C overlap with client A. Thus, they ($B\&C$) are A's domain-relevant clients, while the other two clients ($D\&E$) are domain-irrelevant. Subsequently, we devised a personalized model for client A using the FedAvg algorithm under two distinct scenarios: In scenario (a), we aggregate the models of all 5 clients to generate a personalized model for client A. This scenario encompassed collaborations that intermingled with domain-irrelevant clients. In scenario (b), we only aggregate the models from client A, B, and C to generate a personalized model for client A, concentrating exclusively on collaboration with domain-relevant counterparts. We can observe that the personalized accuracy of user A converges rapidly within a few communication rounds when the collaboration is strictly with domain-relevant clients. Conversely, including domain-irrelevant collaborators in the mix degrades the final personalized accuracy of user A.

Given the privacy protection requirement in the FL system, it's impossible to directly conduct domain relevance analysis between clients on the data level, where the only available medium for information exchange is the model of each client. Therefore, different from the previous simple model similarity perspective, this chapter introduces coalition game theory [19] to perform complex analysis on the model level so that the potential domain relevance at the data level can be accurately reflected. In this way, we can guarantee the domain relevance identification, while strictly adhering to the privacy protection requirement of FL. This theory aids each client in assessing the marginal contributions made by other clients' models to their own personalization process. The calculation of the average marginal contribution of a participating client's model considers all potential combinations of clients within the ongoing personalized coalition. This computation, also referred to as the Shapley Value (SV), encapsulates the collaborative impact of each client's model.

Expanding on this groundwork, we enhance the involvement of individual clients in coalitions and present a personalized FL aggregation algorithm. This algorithm repurposes the SV as aggregation weights, effectively steering the FL training procedure. Notably, this approach showcases robustness even in scenarios with highly Non-IID data distributions. We embark on a theoretical analysis of the convergence and generalization bounds of the proposed algorithm. Additionally, we notice that the local SV evaluation on each client requires them to download the model of others, which raises issues about communication overhead and privacy. Thus, we further utilize a shared feature extractor to reduce communication overhead and differential privacy techniques to protect model privacy.

To the best of our knowledge, this is the first time that coalition game theory has been used as a guiding principle for the personalized collaboration process within FL. In summary, the principal contributions of this chapter are four-fold:

- We revisit the personalized client collaboration problem in FL from the perspective of domain relevance and model this problem as a coalition game.
- We employ the insights from coalition game theory, particularly the Shapley Value (SV), to aid each client in identifying domain-relevant collaborators. This is achieved by assessing the marginal contributions of other clients to their own personalized performance.
- The SV from domain relevance analysis can be reused as aggregation weights to steer the FL training process, which implements a personalized FL aggregation algorithm without any extra information. The convergence and generalization bounds of the algorithm are theoretically analyzed.
- We conduct extensive experiments to validate the performance of our proposed algorithm, pFedSV, on datasets with different non-IID settings. The results show that pFedSV outperforms state-of-the-art baselines.

4.2 Related Work

4.2.1 Personalized Federated Learning

Recently, to address the client data heterogeneity challenge, Personalized Federated Learning (PFL) is proposed by utilizing the knowledge from other clients to customize a unique model for themselves, rather than using the traditional FL method to generate a single global model for all, which can significantly improve the model performance for every client in FL system. Initially, an additional fine-tuning step for the global model on each client's local dataset is a natural strategy for personalization [62, 95], which enables the global model to fit local data domains. Besides, some previous studies also attempted to enhance the robustness of global model under severe data non-IID level. They tried to add regularization term [85] or proximal term [51] to constraint the update of global model, which keeps the robust to all heterogeneous clients However, their methods are all based on the adjustment of a single global model scheme, which cannot satisfy the personalized demand of individual clients at the local data level, as the target distribution of clients in severe data Non-IID setting can be fairly different from the global average aggregation [40]. Therefore, a part of work, such as pFedHN, considers directly generating personalized parameters for each client's model [80]. While most other works try to promote collaboration between different clients to achieve mutual progress, FedFomo [110] and FedAMP [39] follow a similar idea that encourages pairwise collaboration among clients with similar model features, where the former uses loss similarity and the latter adopts parameter similarity. Clients who have higher similarity in these model features will be assigned higher aggregation weights, rather than the previous average. Although pairwise collaboration methods have achieved good results, they still do not capture the essence of PFL: 1) Each client wants to collaborate with others who are truly relevant at the local data level, not model similarity. 2) Model aggregation is a multiwise process, only considering pairwise relationships ignores the intertwined interactions among models. Thus, We introduce SV from coalition game theory to help each client accurately identify their domain-relevant collaborators with privacy guarantee, by complex marginal contribution analysis. Furthermore, the SV can also be reused as personalized model aggregation weights for each client.

4.2.2 Shapley Value for Federated Learning

The conventional FL framework is a multi-party architecture where clients collaboratively train a shared global model with data privacy protection. Considering the heterogeneity of clients in terms of data domain, hardware, resources, etc., the contribution of different clients to the single shared global model varies significantly, which is also very difficult to precisely quantify them. As a fair contribution evaluation metric, the Shapley Value from the cooperation game theory [25] can successfully solve this problem by measuring the marginal contribution of collaborators on the final outcome, where its calculation process considers the final results under various different combinations of collaborators. Therefore, it's widely applied in various multi-party collaboration scenarios, such as FL. Wang *et al.* use SV in FL for various applications: 1) they measure the contribution of different clients for fair credit allocation [92], and 2) they quantify the importance of different features to the final prediction [91]. Song *et al.* achieve a fair profit allocation for clients in FL by using SV as the contribution index [83]. Furthermore, Yu *et al.* also utilizes the fair property of SV to design an incentive mechanism in FL [104]. However, they mainly utilize the desirable properties of SV to ensure the fairness of their contribution evaluation on different clients, but ignore that SV as a meaningful quantitative metric, can also guide the training process of FL. Some other works also notice that SV can be a effective guidance for typical FL training. Nagalapatti *et al.* propose to use SV-based model aggregation on heterogeneous client models to obtain a global model with higher performance [70]. Sun *et al.* present an adaptive SV-based weighting mechanism for the robustness of FL [84]. However, these works cannot be well generalized to the PFL scenario, since their server's general dataset can only enable global SV evaluation. The personalized SV evaluation requires local client data as metric, which is a significant challenge under FL data protection principle. In our work, the SV evaluation of the final model performance can help analyze the underlying data quality of different clients, without disclosing any data privacy. Besides, SV can also be reused as personalized aggregation weight to enhance model robustness against Non-IID data distribution.

4.3 The Essence of PFL Problem

4.3.1 Problem Formulation

The objective of PFL, as described in the literature [42, 113], is to customize personalized models for each client while accommodating their private data distribution through collaboration among a set of clients. In PFL, there are n clients denoted as $C_1, C_2, \ldots, C_n$, and each client has the same model structure $\mathcal{M}$ but with different weights $\theta_1, \theta_2, \ldots, \theta_n$. The personalized models for each client are represented as $\mathcal{M}(\theta_i)$. In contrast to traditional federated learning, each client i has a private dataset $\mathcal{D}_i$ that is sampled from their own distinct data distribution $\mathcal{P}_i$. The loss function for client i is denoted as ℓ_i, and the average loss over the private dataset $\mathcal{D}_i$ is given by $\mathcal{L}_i(\theta_i) = \frac{1}{d_i} \sum_{j \in \mathcal{D}_i} \ell_i(j, \theta_i)$, where d_i represents the size of $\mathcal{D}_i$ and j represents a data sample in $\mathcal{D}_i$. The optimization objective of PFL is to find the optimal set of personalized model parameters $\Theta^* = \arg\min_{\Theta} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_i(\theta_i)$, where Θ is the set of personalized model parameters $\{\theta_i\}_{i=1}^{n}$. In the subsequent analysis, we will investigate the underlying factors contributing to the encountered challenges by conducting a series of pre-experimental analyses. Furthermore, we will propose a collaborative solution using the Shapley value, aiming to mitigate these issues in a multiwise fashion.

4.3.2 Root Causes of PFL Problems

Domain Relevance. According to extensive previous work for data Non-IID problem in FL [113, 51, 52, 44], the model performance degradation is due to the client

Client index	**A**	**B**	**C**	D	E
Label distribution	$\mathbf{[0,1]}$	$\mathbf{[1,2]}$	$\mathbf{[0,3]}$	$[2,7]$	$[3,6]$
Client index	F	G	H	I	J
Label distribution	$[6,7]$	$[4,9]$	$[5,8]$	$[4,8]$	$[5,9]$

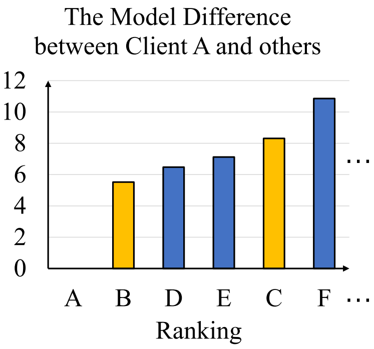


Figure 4.3: The validation of model similarity theory on domain relevance identification, where the table shows the ground truth of client label distribution and the bar chart shows the model difference $||\theta_A - \theta_i||^2$ between client A and other clients.

data domain heterogeneity. However, the inherent data privacy protection of FL makes it difficult to identify other domain-relevant clients, when facing an agnostic system. Since the client models are the only communication intermediary in this situation, previous work directly adopts one-to-one model similarity test to represent domain relevance, i.e., clients with higher model similarity will be regarded as having higher domain relevance. But, there are some flaws lurking behind this theory, which can cause wrong identification. In the table of Fig. 4.3, we show the ground truth of all client label distribution, where the data distribution is pathological Non-IID partition on CIFAR-10 dataset, and numbers $0 \sim 9$ represent the index of different labels. Take client A with labels $[0,1]$ as an example. Client B with labels $[1,2]$ and client C with labels $[0,3]$ are its domain-relevant clients since they both have overlap labels of A. We use Euclidean distance, i.e., $||\theta_A - \theta_i||^2, i \in \{N\}$, to measure the model difference between client A and other clients in Fig. 4.3. If the theory is true, the model differences of B and C should be the smallest among all clients, i.e., $||\theta_A - \theta_B||^2 \approx ||\theta_A - \theta_C||^2 < ||\theta_A - \theta_i||^2, i \in N \setminus \{B, C\}$, while the results in Fig. 4.3 are not consistent with it.

Multiwise Collaboration Weights. Another significant aspect pertains to the aggregation of personalized models within the coalition to generate client-specific

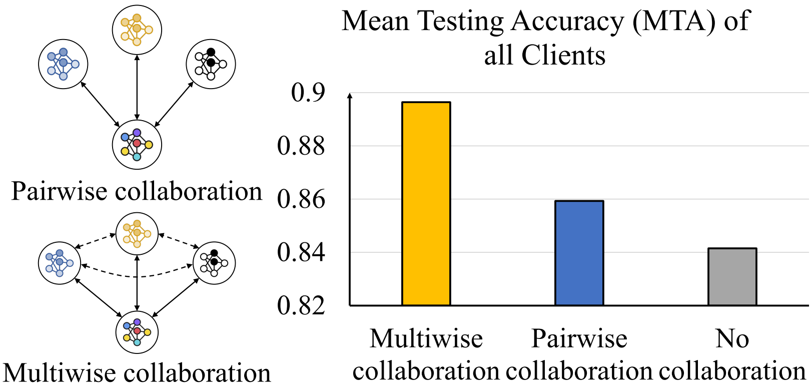


Figure 4.4: The schematic of Multiwise vs. Pairwise collaboration and the experiment results on CIFAR-10 dataset with the pathological Non-IID setting.

models. Previous approaches have predominantly employed pairwise collaboration, involving a comparison of model similarities on a one-to-one basis, and subsequent assignment of aggregation weights proportional to their magnitudes. This methodology is visually depicted in Figure 4.4. However, let us consider a hypothetical scenario where the client's current model is analogous to a carriage, while each of the other clients' models represents a force that propels the carriage towards a specific direction, ultimately reaching the client's optimal personalized model. Evidently, the movement of the carriage is the outcome of a combination of multiple forces. This implies that the collaborative generation of personalized model aggregation weights must take into account the multiwise influences among the collaborators. To investigate this further, we conducted extensive experiments under controlled conditions, wherein each client possessed prior knowledge of domain-relevant clients. The sole variable in these experiments was the method of collaboration employed among the clients for generating aggregated weights. The results depicted in Figure 4.4 indicate the superiority of the multiwise collaboration approach over the pairwise collaboration approach.

4.3.3 Domain-relevant Coalition Formation and Personalized Model Generation

Preliminaries of SV

In the context of the coalition game, each client can be regarded as a player, and the set of all clients is denoted as $N = 1, 2, \ldots, n$. A utility function $v(S) : 2^n \to \mathbb{R}$ is defined, where $S \subseteq N$ represents a coalition of players, and the value $v(S)$ quantifies the overall gain achieved by the coalition. It is conventionally assumed that $v(\emptyset) = 0$, indicating that an empty coalition yields no gain. Formally, let $\pi \in \Pi(N)$ be a permutation of clients in N, and $C_\pi(i) = j \in \pi : \pi(j) < \pi(i)$ represents the coalition consisting of all predecessors of client i in the permutation π. The SV for client i is defined as the average marginal contribution to all possible coalitions $C_\pi(i)$ formed by other clients. It can be calculated using the following formula:

$$\varphi_i(v) = \frac{1}{|N|!} \sum\nolimits_{\pi \in \Pi} [v(C_\pi(i) \cup \{i\}) - v(C_\pi(i))]. \tag{4.1}$$

Alternatively, the SV can be expressed using the following equivalent formulation:

$$\varphi_i(v) = \sum\nolimits_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} [v(S \cup \{i\}) - v(S)]. \tag{4.2}$$

The SV possesses several desirable and unique properties, making it an effective tool for achieving domain-relevant coalition formation for each client and facilitating personalized model generation within the coalition.

SV for Domain Relevance

- *Symmetry*: If two clients, denoted as i and j, make equivalent contributions to any coalition, they should receive the same value. In other words, if for all subsets $S \subseteq N \setminus i, j$ it holds that $v(S \cup i) = v(S \cup j)$, then the Shapley values for clients i and j will be equal, i.e., $\varphi_i = \varphi_j$.

- *Null Player*: when a client makes zero marginal contributions to all possible coalitions. In such cases, the client is considered a null player and receives a Shapley value of zero. Formally, if $v(S \cup i) = 0$ for all subsets $S \subseteq N \setminus i$, then $\varphi_i = 0$.

The *Symmetry* and *Null Player* properties in SV are particularly useful in assisting each client in identifying their domain-relevant counterparts. Clients who do not contribute significantly to any coalition, i.e., *Null Players*, can be identified as irrelevant in the collaborative process. Meanwhile, clients who exhibit similar contributions are treated equally, ensuring fairness and accuracy in the identification of domain-relevant clients.

The process of identifying domain-relevant clients involves a systematic workflow. In each communication round denoted by t, every client i initiates the process by uploading their locally updated model θ_i^t to the server. These individual models collectively constitute a model pool $\theta_{i\,i=1}^{t\,n}$ on the server-side. Subsequently, clients retrieve models belonging to other clients from the model pool, allowing them to construct their own domain-relevant coalition. This coalition formation is based on the downloaded models, which are carefully evaluated for their relevance and compatibility. Following the coalition formation, each client performs personalized model aggregation, leveraging the models obtained from the selected coalition members. It is important to note that in the agnostic federated learning system, the availability of clients is not guaranteed at the outset. Consequently, the composition of the domain-relevant coalition undergoes dynamic reconstruction during the training process to accommodate the evolving presence and relevance of clients.

In order to identify domain-relevant clients, we establish a model download vector for each client based on relevance scores. For client $i \in N$, a relevance vector $\phi^{i,t} = [\phi_1^{i,t}, \cdots, \phi^{i,t} n]$ is generated, where $\phi j^{i,t}$ represents the relevance score of client j to client i in the t-th round. Initially, $\phi^{i,t=0}$ is set to the zero vector. To select which models to download, we consider the top-k clients based on their relevance scores in the vector. (Note: In the first round, k other clients' models are randomly downloaded as the vector is initialized with all zeros). This process allows each client to form a personalized coalition set $S^t_{i,k}$, which includes their own index and the downloaded models. With this coalition set, client i can evaluate its SV using the following coalition game and their local validation dataset $\mathcal{D}_{V_i}$. We define a coalition game $(\theta^t_j j \in Si, k^t, v)$, where v is a utility function that assigns a value to each client subset $X \subseteq S^t_{i,k}$. The value is determined by the performance $\mathcal{A}$ of the model $\theta^t X$ generated from subset X on the validation dataset $\mathcal{D}V_i$, as expressed by the following equations:

$$\theta^t_X = \frac{1}{|X|} \sum\nolimits_{j \in X} \theta^t_j, \text{and } \ v(X, \mathcal{D}_{V_i}) = \mathcal{A}(\theta^t_X, \mathcal{D}_{V_i}). \tag{4.3}$$

By solving the coalition game $(\theta^t_j j \in Si, k^t, v)$ using the Shapley value calculation method, we obtain the Shapley values φ^t_j for all clients $j \in S^t_{i,k}$ within the personalized coalition $S^t_{i,k}$ in the t-th round, as defined by Eq. (4.1). Next, client i updates the relevance scores in their relevance vector to $\phi^{i,t+1}$ using the following formula:

$$\phi^{i,t+1}_j = \alpha \phi^{i,t}_j + (1-\alpha)\varphi^t_j, \quad \forall j \in S^t_{i,k} \tag{4.4}$$

In Eq. (4.4), α is a weight parameter that determines the influence of the previous relevance score $\phi^{i,t}_j$ compared to the obtained Shapley value φ^t_j. This update process enables the relevance scores to be adjusted based on the contributions of the clients in the personalized coalition.

Intuitively, a higher relevance score assigned to client j indicates a greater contribution to the personalized performance of client i. Consequently, it suggests a higher likelihood of client j being considered as a domain-relevant client for client i. Furthermore, it is observed that the relevance vector exhibits instability during the initial rounds, primarily due to the fact that not all models from domain-relevant clients can be downloaded. To address this, the relevance vector undergoes several rounds of iterative updates, employing the top-k scheme for screening. It is worth noting that in the coalition game, if the models of other clients negatively impact the personalized performance, the corresponding Shapley values can be negative. Consequently, the irrelevance of certain clients becomes evident as their scores rapidly decline and may eventually become negative throughout the iterations, leading to their exclusion from consideration. For instance, consider a scenario where client i has a total of 20 clients, with 2 being domain-relevant. Assuming that client i downloads the top-5 models from other clients in each round, it would require a maximum of 5 rounds to identify all the domain-relevant clients. A comprehensive analysis of the convergence of the Dynamic Top-k Download Mechanism is provided in Section 4.4.2.

Dynamic top-k download mechanism: To minimize communication overhead, the download number k in each round can be dynamically adjusted. Through iterative updates, only domain-relevant clients maintain a positive relevance score. Consequently, if the number of clients with a positive score does not match the current value of k, it is dynamically modified to ensure that all downloads are exclusively focused on the necessary domain-relevant clients. This dynamic adjustment guarantees that the communication process is optimized by downloading models only from clients who are deemed relevant, reducing unnecessary data transmission and improving efficiency.

SV for Multiwise Collaboration Weights

- *Group Rationality*: This property ensures that the total gain or value of the entire coalition S is distributed among all the clients in the coalition, i.e., $v(S) = \sum_{i \in S} \varphi_i$.

- *Linearity*: The linearity property states that the values obtained under different utilities can be summed up to obtain the value under a utility that is the sum of all those utilities: $\varphi_i(v) + \varphi_i(u) = \varphi_i(v + u)$. This property is beneficial for personalized model aggregation as it allows for the combination of models trained on different datasets or with different features. This property also ensures that scaling the utility or the contribution of a client by a constant factor results in a proportional scaling of their SV, i.e., for every $i \in N$ and any real number a, it has $\varphi_i(av) = a\varphi_i(v)$.

By satisfying these properties, personalized model aggregation with multiwise collaboration in the coalition adheres to principles of fairness, cooperation, and linearity. These properties provide a solid foundation for effective and equitable collaboration among clients in the coalition.

The computation of the SV in personalized model aggregation, as represented by Eq. (4.1), involves the exhaustive exploration of permutations among clients within the coalition game. Through this process, the intricate multiwise influences that exist among the clients are inherently taken into account, thereby enabling a comprehensive assessment of their respective contributions to the final results. Moreover, the *Group Rationality* property ensures a shared objective among all clients within the coalition. Specifically, their collective aim is to achieve optimal performance for the current client i, thereby necessitating the identification of personalized model parameters that yield the highest performance. Consequently, the distribution of the collective gain or value, as encapsulated by the SV, aligns seamlessly with this common objective, fostering fairness and cooperation within the coalition. The *Linearity* property naturally integrates into the model aggregation process. It captures the relationship between the improvement in personalized accuracy resulting from the aggregation of other client models with one's own. The SV of the model serves as an accurate reflection of this aggregated impact on performance. Notably, a larger positive SV signifies a greater positive contribution to performance improvement, while a larger negative SV indicates a more pronounced detrimental effect. As such, the linearity property facilitates a clear understanding of the relative significance and influence of each client's model within the aggregation process.

Based on the Shapley values (SV) φ_j^t of all clients $j \in S_{i,k}^t$ in Eq. (4.4), the downloaded models are assigned real numbers that quantify their marginal contribution to the personalization of the current client i. A positive number indicates a positive effect, while a negative number indicates a negative effect. To facilitate multiwise collaboration in the current round, it is necessary to exclude models from irrelevant clients with negative SV. Only domain-relevant clients within the coalition are considered for weight computation. The weights w_j^t for these clients are determined as follows:

$$w_j^t = \frac{\max(\varphi_j^t, 0)}{\|\theta_i^t - \theta_j^t\|}, \tag{4.5}$$

Here, the model differences $\|\theta_i^t - \theta_j^t\|$ are incorporated to further refine the resulting weights, taking into account the dissimilarity between the parameters of the current client i and those of the selected clients j. Subsequently, the weights obtained are normalized using 0-1 normalization to obtain personalized aggregation weights w_j^{t*}, satisfying the conditions $w_j^{t*} \in [0, 1]$ and $\sum_j w_j^{t*} = 1$. Finally, the personalized model of client i in the t-th round is generated through the following multiwise collaboration:

$$\theta_i^{t*} = \sum\nolimits_j w_j^{t*} \theta_j^t, \quad \forall j \in S_{i,k}^t. \tag{4.6}$$

It is important to note that SV evaluations are performed in each round to account for small changes in multiwise influences resulting from parameter updates after client local model training. This ensures that the personalized model accurately reflects the evolving collaborative dynamics.

4.4 The pFedSV Algorithm

Based on the above solution frame, we propose our pFedSV Algorithm, where the whole workflow is demonstrated in Algorithm 2 and 3. The procedure begins with each client initializing their model parameters θ_i and the relevance vector ϕ^i (Algorithm 2, Line 1-2). In each round t, the clients update their model parameters to θ_i^t through E

Algorithm 2: Shapley value based Personalized Federated Learning on whole model (pFedSV)

```
Input: $n$, $N$, $\{\theta_i\}_{i=1}^n$, $k$, $E$, $T$, $R$ and $\mathcal{D}_{V_i}$
Output: $\{\theta_i^*\}_{i=1}^n$: clients' personalized model parameters
1  Initialize the clients' model parameters $\{\theta_i\}_{i=1}^n$.
2  Initialize clients' relevence vector: $\phi^{i,t=1} = \vec{0}$, $\forall i \in N$.
3  for client $i = 1, 2, \cdots, n$ do
4      update model parameter to $\theta_i^t$ via $E$ local epochs and upload to the server.
5      download $k$ copies of other clients' model parameters from the server with
         the dynamic top-$k$ download mechanism.
6      $S_{i,k}^t \leftarrow \theta_i^t \cup \{k$ downloaded model parameters$\}$.
7      $\varphi_j^t \Leftarrow$ SV_evaluation$(S_{i,k}^t, \mathcal{D}_{V_i}, R)$, $\forall j \in S_{i,k}^t$.      ▷ Details in Algorithm 2
8      $\phi_j^{i,t+1} = \alpha\phi_j^{i,t} + (1-\alpha)\varphi_j^t, \forall j \in S_{i,k}^t$
9      $w_j^{t*} = \frac{w_j^t}{\sum_j w_j^t} \Leftarrow w_j^t = \frac{\max(\varphi_j^t, 0)}{\|\theta_i^t - \theta_j^t\|}$, $\forall j \in S_{i,k}^t$.
10     $\theta_i^{t*} = \sum_j w_j^{t*}\theta_j^t, \forall j \in S_{i,k}^t$.
```

local epochs of training and subsequently upload them to the server (Algorithm 2, Line 4). Subsequently, the clients download k copies of other clients' model parameters based on a dynamic top-k download mechanism (Algorithm 2, Line 5). At this stage, the fundamental conditions for each client's coalition game, required for their own model personalization, are established. Initially, a coalition game $(\theta_j^t j \in Si, k^t, v)$ is formed, where $S_{i,k}^t$ represents the set of model parameters comprising the downloaded models as well as the client's own model (Algorithm 2, Line 6). Then, the process of evaluating the Shapley values (SV) is performed to obtain the SV for each model parameter in $S_{i,k}^t$ (Algorithm 2, Line 7), which will be further elaborated in Algorithm 3. The obtained SV values serve two purposes: updating the relevance vector of each client to identify their domain-relevant clients (Line 8), and calculating multiwise aggregation weights for model personalization (Algorithm 2, Line 9). Finally, each client performs the respective weighted aggregation to obtain new model parameters, which serve as the starting point for the subsequent round $t+1$.

Algorithm 3: Shapley value evaluation

```
  Input: $S^t_{i,k}$, $\mathcal{D}_{V_i}$, $R$.
  Output: $\varphi^t_j, \forall j \in S^t_{i,k}$.
1 $P \leftarrow$ set of $R$ permutations of $S^t_{i,k}$.
2 for client $j \in S^t_{i,k}$ do
3     for permutation $p \in P$ do
4         $X^t_{p,j} = \{l | l \in S^t_{i,k} \wedge p(l) \leq j\}$
5         $a^p_j \leftarrow v(\{X^t_{p,j} \cup j\}, \mathcal{D}_{V_i}) - v(X^t_{p,j}, \mathcal{D}_{V_i})$
6         $\varphi^t_j \leftarrow \varphi^t_j + \frac{1}{|P|} a^p_j$.
```

To address the exponential time complexity required for accurate evaluation of SV, an approximation algorithm is employed. A widely accepted approach is to utilize Monte Carlo sampling techniques, which treat the computation of SV as an expectation calculation problem [61, 15, 60]. The details of the approximation process are outlined in Algorithm 3. First, a set P is created by randomly sampling R permutations of $S^t_{i,k}$ from the total $|S^t_{i,k}|!$ possible permutations (Algorithm 3, Line 1). For each permutation in P, the algorithm scans the elements from the first to the last and calculates the marginal contribution for each newly added model parameter (Algorithm 3, Line 3-5). This procedure is repeated for all R permutations, and the approximation of the SV is obtained by averaging the calculated marginal contributions (Algorithm 3, Line 6). As the number of samples R increases, the Monte Carlo sampling technique becomes an unbiased estimate of the SV. Therefore, by gradually increasing the number of samples, a more accurate approximation of the SV can be obtained.

4.4.1 Convergence of SV Evaluation Approximation.

The computation complexity for precise SV evaluation is exponential to the number of players. According to Eq. (4.1), the computation process can be viewed as an expectation calculation problem, thus the Monte Carlo sampling technique can be used to approximate the SV. It will converge to an unbiased estimate of the SV as the increasing of sampling number R. It's proved that $R = 3|S_{i,k}^t| \ll |S_{i,k}^t|!$ Monte Carlo sampling number is sufficient for convergence, with a small approximation bound $\epsilon > 0$ [60].

4.4.2 Convergence Analysis of Dynamic Top-k Download Mechanism

Assume that there are total n clients with 100% participation, the local data distributions of these clients follow the pathological data Non-IID setting, where each client is randomly assigned m types of labels. An example of client label distribution on the CIFAR-10 dataset with $m = 2$ is shown in Fig. 4.3 for reference. Domain heterogeneity is defined as each client's label distribution is different, while domain relevance is defined as there are same class labels between different clients. Therefore, we can observe from the ground truth of Fig. 4.3 that each client has m other domain-relevant clients in this setting from an omniscient perspective.

Suppose that the initial model download number for each client is k. Then, we provide the convergence proof of our dynamic top-k download mechanism. Take the personalization process of client A as an example, there are two conditions for the settings of hyperparameters m and k ($m < k$ or $m > k$), and we will explain them one by one.

When $m < k$**:** In the first round, each client will randomly download k copies of other clients' models from the server-side and there are various (C_n^k) possible model combinations.

- **For the best case**, other m domain-related clients' models are all included in the initial k copies, that is for $\forall i \in \{m\}$, we have $i \in \{k\}$. Thus, we can identify all domain-relevant clients of client A in the first round, where the SV of the domain-relevant clients is positive and the domain-irrelevant clients are negative.

- **For the worst case**, none of the m domain-related models is included in the first k copies, that is for $\forall i \in \{m\}$, we have $i \notin \{k\}$. Next, we prove the maximum number of rounds that is required to identify the m domain-relevant clients from all n clients when the worst case occurs in each round. For the first round, since k copies of models are all from the domain-irrelevant clients, their SV will be negative in the evaluation process, which makes their relevance score be negative after updating. Therefore, according to the top-k rule, these clients will not be selected in the next round because the relevant scores of other clients who have never been selected are the initial 0, which is larger than negative scores. The worst case will continue until a certain round t, which satisfies $tk > n - m - 1$ (1 is client A itself). It means that in round t, we have excluded all domain-irrelevant clients with negative SV, and the remaining clients are all domain-relevant clients. Since $k > m$ (they are both integers), we have $(t+1)k = tk + k > n - m - 1 + k > n + (k - m - 1) \geq n$, which means that we must be able to find all domain-relevant clients in the next round $t+1$. Finally, we prove that it takes at most $\lceil \frac{n-m-1}{k} \rceil + 1$ round to identify all other domain-relevant clients.

When $m > k$, following the similar logic as above, we can get the subsequent convergence proof.

- **For the best case**, since $m > k$, we cannot include all m domain-relevant clients in the first round with only k downloaded models. Therefore, the process will continue until all clients are scanned by once. Thus, we need $\lceil \frac{m}{k} \rceil$ round to identify all domain-relevant clients.

- **For the worst case**, we need $\lceil \frac{n-m-1}{k} \rceil$ rounds to exclude all domain-irrelevant clients and then we still need up to $\lceil \frac{m}{k} \rceil$ rounds to identify all domain-relevant clients. Finally, it takes at most $\lceil \frac{n-m-1}{k} \rceil + \lceil \frac{m}{k} \rceil$ rounds.

Normally, to ensure efficient traversal, we will set a large value of k at the beginning. Although a large k leads to a large communication overhead in the beginning, it can help the client rapidly scan all other clients and converge to a specific value $k = m$, which is equal to the number of other domain-relevant clients.

4.4.3 Convergence Analysis of pFedSV.

We prove that pFedSV can assist each client converge to their respective local optimums under the following assumptions: 1) $\mathcal{L}_1, \cdots, \mathcal{L}_n$ are all μ-strongly-convex, 2) $\mathcal{L}_1, \cdots, \mathcal{L}_n$ are all L-smooth, 3) the variance of stochastic gradients in each client is bounded by σ_i^2 and 4) the expected squared norm of stochastic gradients is uniformly bounded by G^2.

Theorem 2. *Let all above assumptions hold and μ, L, σ_i, G are defined therein. Choose $\kappa = \frac{L}{\mu}$, $\gamma = \max\{8\kappa, E\}$ and the learning rate $\eta_t = \frac{2}{\mu(\gamma+t)}$. Then, each client in pFedSV satisfies*

$$\mathbb{E}[\mathcal{L}_i(\theta_i)] - \mathcal{L}_i^* \leq \frac{\kappa}{\gamma + T - 1} \left(\frac{2B}{\mu} + \frac{\mu\gamma}{2} \mathbb{E}\|\theta_i^1 - \theta_i^*\|^2 \right) \tag{4.7}$$

The full version of the convergence analysis of the pFedSV algorithm will be elaborated as follows:

The personalized performance convergence analysis for each client is the same, so we only focus on one client $i \in \{N\}$. Consider a scenario where each client parallel performs E local SGD step to update their own model. Then, they will communicate with the server to download the model for personalized model aggregation, which is denoted as the synchronization step. First, we analyze the case on pFedSV that all other clients (including both domain-relevant and domain-irrelevant clients) participate in the aggregation step to generate the personalized model.

Additional Notation

Let θ_t^k represent the model parameters of the k-th client at the t-th step. The variable E denotes the number of local update epochs, while $\mathcal{I}_E$ represents the set of synchronization steps, defined as $\mathcal{I}_E = \{nE; |; n = 1, 2, \cdots\}$. If $t+1 \in \mathcal{I}_E$, it signifies that the model update involving all participants can be described as follows:

$$\mathrm{v}_{t+1}^k = \theta_t^k - \eta_t \nabla \mathcal{L}_k(\theta_t^k, \xi_t^k) \tag{4.8}$$

$$\theta_{t+1}^k = \begin{cases} \mathrm{v}_{t+1}^k, & if \quad t+1 \notin \mathcal{I}_E \\ \sum_{k=1}^N p_k \mathrm{v}_{t+1}^k, & if \quad t+1 \in \mathcal{I}_E \end{cases} \tag{4.9}$$

In this context, an auxiliary variable v_{t+1}^k is introduced to capture the immediate outcome of a single step of stochastic gradient descent (SGD) applied to θ_{t+1}^k. The parameter θ_{t+1}^k itself corresponds to the model parameter obtained after the communication steps.

In the subsequent analysis, we introduce two virtual sequences: $\bar{\mathrm{v}}_{t,i} = \sum_{k=1}^N p_k \mathrm{v}_{t+1}^k$ and $\bar{\theta}_{t,i} = \sum_{k=1}^N p_k \theta_{t+1}^k$. These sequences serve as useful abstractions for our purposes. The virtual sequence $\bar{\mathrm{v}}_{t+1,i}$ is obtained by performing a single step of stochastic gradient descent (SGD) on $\bar{\theta} t, i$. It is generated as a result of this computation. To facilitate the analysis, we also define $\bar{g}_{t,i} = \sum_{k=1}^N p_k \nabla \mathcal{L} k(\theta_t^k)$ and $g_{t,i} = \sum_{k=1}^N p_k \nabla \mathcal{L} k(\theta_t^k, \xi_t^k)$. Consequently, we establish the relationship $\bar{\mathrm{v}}_{t+1,i} = \bar{\theta}_{t,i} - \eta_t g_{t,i}$, while observing that the expected value of $g_{t,i}$ is equivalent to $\bar{g}_{t,i}$.

Key Lemmas

In order to present a clear proof, it is essential to establish several lemmas prior to stating the main theorem. The detailed proofs of these lemmas can be found in the work by Li et al. [52]. For the purpose of this discussion, we will focus solely on presenting the main theorem.

Lemma 1. *The results of one step SGD. Assume the assumption* 1 *and* 2 *hold. we have*

$$\mathbb{E}||\bar{\mathrm{v}}_{t+1,i}-\theta_i^*|| \leq (1-\eta_t\mu)\mathbb{E}||\bar{\theta}_{t,i}-\theta_i^*||^2+\eta_t^2\mathbb{E}||g_{t,i}-\bar{g}_{t,i}||^2+6L\eta_t^2\Gamma+2\mathbb{E}\sum_{k=1}^{N}p_k||\bar{\theta}_{t,i}-\theta_k^t||^2$$

where $\Gamma = \mathcal{L}_i^* - \sum_{k=1}^N p_k\mathcal{L}_k^* \geq 0$.

Lemma 2. *Bounding the variance. Assume Assumption* 3 *holds. It follows that*

$$\mathbb{E}||g_{t,i}-\bar{g}_{t,i}||^2 \leq \sum_{k=1}^{N} p_k^2\sigma_k^2$$

Lemma 3. *Bounding the divergence of* $\{\theta_{t,i}^k\}$*. Assume Assumption* 4 *holds, that* η_t *is non-increasing and* $\eta_t \leq 2\eta_{t+E}$ *for all* $t \geq 0$*. It follows that*

$$\mathbb{E}\left[\sum_{k=1}^{N} p_k||\bar{\theta}_{t,i}-\theta_k^t||^2\right] \leq 4\eta_t^2(E-1)^2G^2.$$

Full Proof of Theorem 2

Proof. No matter whether $t+1 \in \mathcal{I}_E$ or $t+1 \notin \mathcal{I}_E$, we always have the following equation: $\bar{\theta}_{t+1,i} = \bar{\mathrm{v}}_{t+1,i}$. Let $\Delta_t = \mathbb{E}||\bar{\theta}_{t,i}-\theta_i^*||^2$. From Lemma 1, Lemma 2 and Lemma 3, it follows that

$$\Delta_{t+1} \leq (1-\eta_t\mu)\Delta_t + \eta_t^2 B$$

where

$$B = \sum_{k=1}^{N} p_k^2\sigma_k^2 + 6L\Gamma + 8(E-1)^2G^2$$

For a diminishing stepsize, $\eta_t = \frac{\beta}{t+\gamma}$ for some $\beta > \frac{1}{\mu}$ and $\gamma > 0$ such that $\eta_1 \leq \min\{\frac{1}{\mu}, \frac{1}{4L}\}$ and $\eta_t \leq 2\eta_{t+E}$. We prove that $\Delta_t \leq \frac{v}{\gamma+t}$ where $v = \max\{\frac{\beta^2 B}{\beta\mu-1}, (\gamma+1)\Delta_1\}$. First, the definition of v ensures that it holds for $t = 1$. Assume the conclusion holds for some t, it follows that

$$\begin{aligned}\Delta_{t+1} &\leq (1-\eta_t\mu)\Delta_t + \eta_t^2 B \\ &\leq \left(1 - \frac{\beta\mu}{t+\gamma}\right)\frac{v}{t+\gamma} + \frac{\beta^2 B}{(t+\gamma)^2} \\ &= \frac{t+\gamma-1}{(t+\gamma)^2}v + \left[\frac{\beta^2 B}{(t+\gamma)^2} - \frac{\beta\mu-1}{(t+\gamma)^2}v\right] \\ &\leq \frac{v}{\gamma+t+1}\end{aligned} \tag{4.10}$$

Then,according to the L-smoothness of $\mathcal{L}_i(\cdot)$, we have

$$\mathbb{E}[\mathcal{L}_i(\bar{\theta}_{t,i})] - \mathcal{L}_i^* \leq \frac{L}{2}\Delta_t \leq \frac{L}{2}\frac{v}{\gamma+t}.$$

Specifically, if we choose $\beta = \frac{2}{\mu}$, $\gamma = \max\{8\frac{L}{\mu}, E\} - 1$ and denote $\kappa = \frac{L}{\mu}$, then $\eta_t = \frac{2}{\mu}\frac{1}{\gamma+t}$. One can verify that the choice of $\eta_t \leq 2\eta_{t+E}$ for $t \geq 1$. Then we have

$$v = \max\{\frac{\beta^2 B}{\beta\mu-1}, (\gamma+1)\Delta_1\} \leq \frac{\beta^2 B}{\beta\mu-1} + (\gamma+1)\Delta_1 \leq \frac{4B}{\mu^2} + (\gamma+1)\Delta_1,$$

and

$$\mathbb{E}[\mathcal{L}_i(\bar{\theta}_{t,i})] - \mathcal{L}_i^* \leq \frac{L}{2}\frac{v}{\gamma+t} \leq \frac{\kappa}{\gamma+t}\left(\frac{2B}{\mu} + \frac{\mu(\gamma+1)}{2}\Delta_1\right)$$

□

4.4.4 Generalization Bounds of pFedSV.

In this section, We theoretically prove that the performance of pFedSV can outperform conventional FedAvg algorithm and local training only, through the theorem from domain adaptation [4].

Theorem 3. *For each client $i \in N$, we denote its local distribution and empirical distribution as $\mathcal{D}_i$ and $\hat{\mathcal{D}}_i$. The model parameters learned on $\hat{\mathcal{D}}_i$ is denoted by $\theta_{\hat{\mathcal{D}}_i}$. Then we have*

$$\begin{aligned}\mathcal{L}_{\mathcal{D}_i}\left(\sum\nolimits_j w^{t*}_{\hat{\mathcal{D}}_j}\theta^t_{\hat{\mathcal{D}}_j}\right) &\leq \mathcal{L}_{\mathcal{D}_i}\left(\frac{1}{n}\sum\nolimits_j \theta^t_{\hat{\mathcal{D}}_j}\right)\\ &\leq \mathcal{L}_{\hat{\mathcal{D}}_i}(\theta_{\hat{\mathcal{D}}_i}) + \frac{1}{n}\sum_j\left(\frac{1}{2}d(\mathcal{D}_i,\mathcal{D}_j)+\xi_j\right) + \sqrt{\frac{\log\frac{2n}{\delta}}{2m}}\end{aligned} \tag{4.11}$$

where $w^{t*}_{\hat{\mathcal{D}}_j}$ is the SV-based aggregation weight, $d(\cdot)$ measures the distribution discrepancy between two distributions, m is the number of samples per local distribution and ξ_j is the minimum of the combined loss $\mathcal{L}_{\hat{\mathcal{D}}_i}+\mathcal{L}_{\hat{\mathcal{D}}_j}$. The detailed proof of generalization bounds is elaborated as follows:

Proof. Before the analysis of the generalization bound, we introduce the following notations. In PFL, each client has its own local data distribution $\mathcal{D}_i$ over domain $\Xi := \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \in \mathbb{R}^d$ is the input space and $\mathcal{Y}$ is the output space. For the empirical distribution $\hat{\mathcal{D}}_i$ by the given dataset, we assume that each client local model has access to an equal amount (m) of local data samples. For each client, we assume the local model θ as a mapping $\theta : \mathcal{X} \to \mathcal{Y}$. The cross-entropy loss function of task is defined as $\mathcal{L}(\theta(x), y) = \mathcal{L}(\hat{y}, y)$, where $\hat{y} := \theta(x)$. Note that $\mathcal{L}(\hat{y}, y)$ is convex with respect to $\hat{y}$. We denote $\arg\min_{\theta\in\Theta}\mathcal{L}_{\hat{\mathcal{D}}_i}(\theta)$ by $\theta_{\hat{\mathcal{D}}_i}$

According to the Domain Adaptation theory [4], we utilize the domain measurement tools developed below to analyze the generalization bound of the personalized model that is aggregated from an ensemble of other clients' models.

Theorem 4. *(Domain Adaptation) Considering the distribution $\mathcal{D}_S$ and $\mathcal{D}_T$, for every $\theta \in \Theta$ and any $\delta \in (0, 1)$, with probability at least $1 - \delta$ (over the choice of the samples), there exists:*

$$\mathcal{L}_{\mathcal{D}_T}(\theta) \leq \mathcal{L}_{\mathcal{D}_S}(\theta) + \frac{1}{2}d(\mathcal{D}_S, \mathcal{D}_T) + \lambda, \tag{4.12}$$

where $\lambda = \mathcal{L}_{\mathcal{D}_S}(\theta^*) + \mathcal{L}_{\mathcal{D}_T}(\theta^*)$, and $\theta^* := \arg\min_{\theta\in\Theta}\mathcal{L}_{\mathcal{D}_S}(\theta) + \mathcal{L}_{\mathcal{D}_T}(\theta)$ corresponds to the optimal joint model that minimize the combined loss.

Now we start the proof of Theorem 3 by two parts. 1) we first prove that the personalized model aggregated by the FedAvg algorithm for each client is better than training with their own local data only. 2) Then, we prove that the aggregation by pFedSV only on other domain-relevant clients is better than FedAvg with all clients' participation.

• For the first part, we start with the risk of the personalized model of client i, $\mathcal{L}_{\mathcal{D}_i}(\frac{1}{n}\sum_j \theta_{\hat{\mathcal{D}}_j})$, which is aggregated from FedAvg with the participation of all other clients.

Considering the distance between $\mathcal{L}_{\mathcal{D}_i}(\frac{1}{n}\sum_j \theta_{\hat{\mathcal{D}}_j})$ and $\mathcal{L}_{\hat{\mathcal{D}}_i}(\theta_{\hat{\mathcal{D}}_i})$. By the convexity of $\mathcal{L}$ and Jensen inequality, we have

$$\mathcal{L}_{\mathcal{D}_i}(\frac{1}{n}\sum_j \theta_{\hat{\mathcal{D}}_j}) \leq \frac{1}{n}\sum_j \mathcal{L}_{\mathcal{D}_i}(\theta_{\hat{\mathcal{D}}_j}). \tag{4.13}$$

Using the domain adaptation theory, we transfer from domain $\mathcal{D}_i$ to $\mathcal{D}_j$,

$$\mathcal{L}_{\mathcal{D}_i}(\theta_{\hat{\mathcal{D}}_j}) \leq \mathcal{L}_{\mathcal{D}_j}(\theta_{\hat{\mathcal{D}}_j}) + \frac{1}{2}d(\mathcal{D}_j, \mathcal{D}_i) + \lambda_j, \tag{4.14}$$

where $\lambda_j := \mathcal{L}_{\mathcal{D}_i}(\theta^*) + \mathcal{L}_{\mathcal{D}_j}(\theta^*)$ and $\theta^* := \arg\min_{\theta\in\Theta} \mathcal{L}_{\mathcal{D}_i}(\theta) + \mathcal{L}_{\mathcal{D}_j}(\theta)$.

We can bound the risk with its empirical counterpart through Hoeffding in equality, which gives

$$\Pr\left[\left|\mathcal{L}_{\mathcal{D}_j}(\theta_{\hat{\mathcal{D}}_j}) - \mathcal{L}_{\hat{\mathcal{D}}_j}(\theta_{\hat{\mathcal{D}}_j})\right| \geq \epsilon\right] \leq 2\exp\frac{-2m^2\epsilon^2}{\sum_{k=1}^{m}(b-a)^2}, \tag{4.15}$$

where $[a, b]$ is the range of loss function. In our case, the loss function is bounded in $[0, 1]$ so that $(b-a)^2 \leq 1$. Thus, with the probability at least $1 - \frac{\delta}{n}$, over the draw of m i.i.d. samples S_j from $\mathcal{D}_j$,

$$\mathcal{L}_{\mathcal{D}_j}(\theta_{\hat{\mathcal{D}}_j}) \leq \mathcal{L}_{\hat{\mathcal{D}}_j}(\theta_{\hat{\mathcal{D}}_j}) + \sqrt{\frac{\log\frac{2}{\frac{\delta}{n}}}{2m}}, \tag{4.16}$$

Thus for n sources, we have

$$\begin{aligned}
&\Pr_{S_1 \sim \mathcal{D}_1^m, \cdots, S_n \sim \mathcal{D}_n^m} \left[\bigcap_{j=1}^{n} \left\{ \mathcal{L}_{\mathcal{D}_j}(\theta_{\hat{\mathcal{D}}_j}) \leq \mathcal{L}_{\hat{\mathcal{D}}_j}(\theta_{\hat{\mathcal{D}}_j}) + \sqrt{\frac{\log \frac{2}{\frac{\delta}{n}}}{2m}} \right\} \right] \\
&= 1 - \Pr_{S_1 \sim \mathcal{D}_1^m, \cdots, S_n \sim \mathcal{D}_n^m} \left[\bigcup_{j=1}^{n} \left\{ \mathcal{L}_{\mathcal{D}_j}(\theta_{\hat{\mathcal{D}}_j}) \geq \mathcal{L}_{\hat{\mathcal{D}}_j}(\theta_{\hat{\mathcal{D}}_j}) + \sqrt{\frac{\log \frac{2}{\frac{\delta}{n}}}{2m}} \right\} \right] \\
&\geq 1 - \sum_{j=1}^{n} \Pr_{S_1 \sim \mathcal{D}_1^m, \cdots, S_n \sim \mathcal{D}_n^m} \left[\left\{ \mathcal{L}_{\mathcal{D}_j}(\theta_{\hat{\mathcal{D}}_j}) \geq \mathcal{L}_{\hat{\mathcal{D}}_j}(\theta_{\hat{\mathcal{D}}_j}) + \sqrt{\frac{\log \frac{2}{\frac{\delta}{n}}}{2m}} \right\} \right] \\
&\geq 1 - \delta.
\end{aligned} \tag{4.17}$$

Based on the definition of ERM, we have $\mathcal{L}_{\hat{\mathcal{D}}_j}(\theta_{\hat{\mathcal{D}}_j}) \leq \mathcal{L}_{\hat{\mathcal{D}}_j}(\theta_{\hat{\mathcal{D}}_i})$, where $\theta_{\hat{\mathcal{D}}_i}$ is the personalized model trained on client i. By using the definition of $\hat{\mathcal{D}}_i$ ($\hat{\mathcal{D}}_i = \frac{1}{n}\sum_j \hat{\mathcal{D}}_j$) and the linearity of expectation, we have

$$\frac{1}{n}\sum_j \mathcal{L}_{\hat{\mathcal{D}}_j}(\theta_{\hat{\mathcal{D}}_j}) \leq \frac{1}{n}\sum_j \mathcal{L}_{\hat{\mathcal{D}}_j}(\theta_{\hat{\mathcal{D}}_i}) = \mathcal{L}_{\hat{\mathcal{D}}_i}(\theta_{\hat{\mathcal{D}}_i}). \tag{4.18}$$

Putting these equations together, we have probability of at least $1-\delta$ over $S_1 \sim \mathcal{D}_1^m, \cdots, S_n \sim \mathcal{D}_n^m$ that

$$\begin{aligned}
\mathcal{L}_{\mathcal{D}_i}(\frac{1}{n}\sum_j \theta_{\hat{\mathcal{D}}_j}) &\leq \frac{1}{n}\sum_j \mathcal{L}_{\mathcal{D}_i}(\theta_{\hat{\mathcal{D}}_j}) \\
&\leq \frac{1}{n}\sum_j \left(\mathcal{L}_{\mathcal{D}_j}(\theta_{\hat{\mathcal{D}}_j}) + \frac{1}{2}d(\mathcal{D}_j, \mathcal{D}_i) + \lambda_k \right) \\
&\leq \frac{1}{n}\sum_j \left(\mathcal{L}_{\mathcal{D}_j}(\theta_{\hat{\mathcal{D}}_j}) + \sqrt{\frac{\log \frac{2n}{\delta}}{2m}} + \frac{1}{2}d(\mathcal{D}_j, \mathcal{D}_i) + \lambda_k \right) \\
&\leq \frac{1}{n}\sum_j \mathcal{L}_{\mathcal{D}_j}(\theta_{\hat{\mathcal{D}}_j}) + \sqrt{\frac{\log \frac{2n}{\delta}}{2m}} + \frac{1}{n}\sum_j \left(\frac{1}{2}d(\mathcal{D}_j, \mathcal{D}_i) + \lambda_k \right) \\
&\leq \mathcal{L}_{\mathcal{D}_i}(\theta_{\hat{\mathcal{D}}_i}) + \sqrt{\frac{\log \frac{2n}{\delta}}{2m}} + \frac{1}{n}\sum_j \left(\frac{1}{2}d(\mathcal{D}_j, \mathcal{D}_i) + \lambda_k \right),
\end{aligned} \tag{4.19}$$

where $\lambda_k = \inf_{\theta \in \Theta}(\mathcal{L}_{\mathcal{D}_i}(\theta) + \mathcal{L}_{\mathcal{D}_j}(\theta))$.

• For the second part, we prove that the collaboration with only other domain-relevant clients by pFedSV is better than the collaboration with all clients mixed by FedAvg. According to the domain relevance theory and the personalized performance convergence analysis, the collaboration mixed with domain-irrelevant clients will degrade the personalized model performance. Assume the domain-relevant clients set of client i is $\mathcal{R}$. Thus, we have

$$\mathcal{L}_{\mathcal{D}_i}\left(\frac{1}{n}\sum_{j\in\mathcal{R}}\theta_{\hat{\mathcal{D}}_j}\right) \leq \mathcal{L}_{\mathcal{D}_i}\left(\frac{1}{n}\sum_{j\in\mathcal{R}}\theta_{\hat{\mathcal{D}}_j} + \frac{1}{n}\sum_{j\notin\mathcal{R}}\theta_{\hat{\mathcal{D}}_j}\right) = \mathcal{L}_{\mathcal{D}_i}(\frac{1}{n}\sum_{j}\theta_{\hat{\mathcal{D}}_j}). \tag{4.20}$$

The domain adaptation theory provide insights that for two models ($\theta_{\mathcal{D}_S}$ and $\theta_{\mathcal{D}_{S'}}$) trained on different source domains (S and S'). The higher the relevance between the source and target domains, the better the performance of the models, which means:

$$\mathcal{L}_{\mathcal{D}_T}(\theta_{\mathcal{D}_S}) \leq \mathcal{L}_{\mathcal{D}_T}(\theta_{\mathcal{D}_{S'}}), if\ d(\mathcal{D}_T, \mathcal{D}_S) \leq d(\mathcal{D}_T, \mathcal{D}_{S'})$$

On the other hand, our pFedSV can precisely identify the domain relevance and assign the aggregation weights $w^*_{\hat{\mathcal{D}}_j}$ according the relevance (The higher relevance, the larger weights). Thus, we have

$$d\left(\theta^*_{\mathcal{D}_i}, \sum_{j\in\mathcal{R}} w^*_{\hat{\mathcal{D}}_j}\theta_{\hat{\mathcal{D}}_j}\right) \leq d\left(\theta^*_{\mathcal{D}_i}, \frac{1}{n}\sum_{j\in\mathcal{R}}\theta_{\hat{\mathcal{D}}_j}\right)$$

and then

$$\mathcal{L}_{\mathcal{D}_i}\left(\sum_{j\in\mathcal{R}} w^*_{\hat{\mathcal{D}}_j}\theta_{\hat{\mathcal{D}}_j}\right) \leq \mathcal{L}_{\mathcal{D}_i}\left(\frac{1}{n}\sum_{j\in\mathcal{R}}\theta_{\hat{\mathcal{D}}_j}\right)$$

Finally, we have

$$\mathcal{L}_{\mathcal{D}_i}\left(\sum\nolimits_j w^{t*}_{\hat{\mathcal{D}}_j}\theta^t_{\hat{\mathcal{D}}_j}\right) \leq \mathcal{L}_{\mathcal{D}_i}\left(\frac{1}{n}\sum\nolimits_j \theta^t_{\hat{\mathcal{D}}_j}\right) \leq \mathcal{L}_{\hat{\mathcal{D}}_i}(\theta_{\hat{\mathcal{D}}_i}) + \frac{1}{n}\sum_j\left(\frac{1}{2}d(\mathcal{D}_i, \mathcal{D}_j) + \xi_j\right) + \sqrt{\frac{\log\frac{2n}{\delta}}{2m}} \tag{4.21}$$

4.5 Communication Overhead Reduction & Model Privacy Protection

Since each user needs to download many model copies of other users to perform their local SV evaluation in our pFedSV algorithm, we are also aware of the potential communication overhead increase and model privacy issues arising from this model downloading process, and provide solid solutions to address them in this section.

4.5.1 Communication Overhead Reduction

Except for the top-k dynamic mechanism in 4.3.3, we further exploit the advantage of a global shared feature extractor between users to reduce the communication overhead. Specifically, for different learning tasks (i.e., image classification and next word prediction), the model can be divided into two parts: feature extractor and classifier, where the former has a generic function for all users, and the latter is unique for different user's local data domains [18]. According to the latest research [57], they measure the Centered Kernel Alignment (CKA) similarity between the representations from the same layer of different clients' local models, on standard CNN [50]. The observation is clear: comparing different layers in the local models learned on different clients, the similarity of feature extractors among different client local models is very high, while the classifiers have the lowest similarity

Therefore, for the personalization of each user, the most important thing they need to focus on is the classifier of other users, while the feature extractor part can be shared. Following this insight, each user only needs to download one global shared feature extractor and several classifiers of other users to reduce the communication overhead, not the whole model before. And the whole model of other users can be reconstructed by replacing different classifiers. The modified federated learning workflow is demonstrated in Fig.4.5.

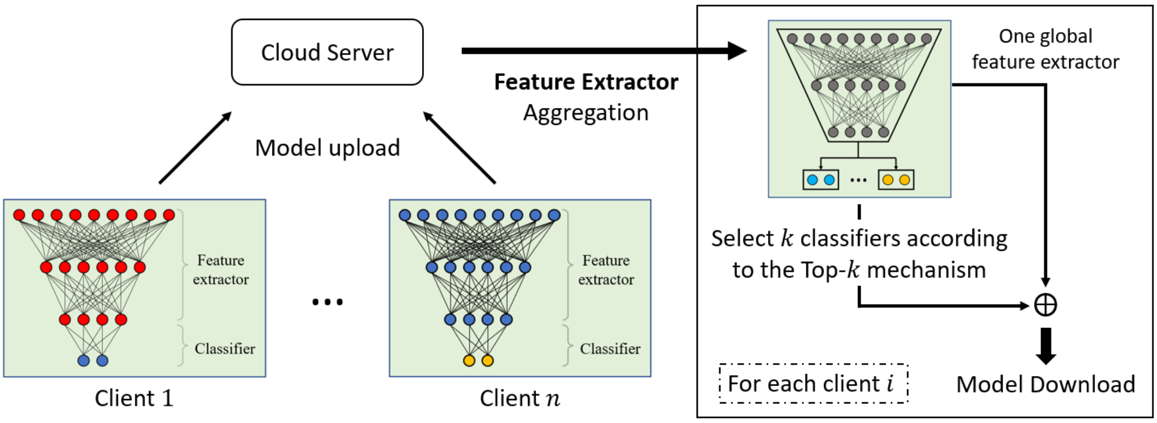


Figure 4.5: The modified federated learning workflow is based on model splitting, where the model of each user is divided into a feature extractor and classifier. The server only generates a global shared feature extractor among different users while maintaining their personalized classifiers.

Nevertheless, since our pFedSV is a general personalization algorithm for different learning tasks, other classical techniques, such as model quantification and compression, can also be applied for further communication reduction.

4.5.2 Model Privacy Protection

To address the issue of model privacy in our proposed framework, we have implemented a two-fold approach. Firstly, we have ensured anonymity by removing any client-specific information from the downloaded models throughout the entire process of pFedSV. This measure ensures that the models themselves do not contain any identifiable information pertaining to individual clients. However, it is important to note that the model parameters themselves may still indirectly reveal sensitive client data. To tackle this concern more effectively, we have incorporated a robust privacy protection mechanism based on (ϵ, δ)-differential privacy (DP) principles, as outlined by Abadi *et al.* [1]. In our approach, we introduce Gaussian noise into the model parameters following each client's local training process. This additional noise injection guarantees that the resulting models satisfy the requirements of differential privacy. By adopting this technique, we provide a stronger level of privacy protection, ensuring that the models do not inadvertently leak sensitive client information. The level of privacy in our framework can be adjusted by introducing more noise into the model parameters. However, it is important to note that increasing the amount of noise comes at the expense of performance. To evaluate the trade-off between privacy and performance, we conducted extensive experiments. The results, which are presented in a later section, demonstrate that our pFedSV algorithm with DP-based Noise Addition can still outperform other personalized baselines while providing an appropriate level of privacy protection (with $\delta = 1$).

4.6 Experiments

4.6.1 Experimental Setup

In this section, we will show all the experiment setups, including hyperparameter settings, datasets, baselines, etc.

Dataset, Model & Machine Configurations. Based on prior work [64, 53], we conduct our experiments on the following datasets: MNIST [50], Fashion-MNIST (F-MNIST) [101], CIFAR-10 [49], and CIFAR-100. For the model structure on different datasets, We use the same CNN architecture as in [64]. All our experiments are run on the following machine configurations: CPU (i9-10900K) and GPU (one RTX 3090).

Baselines & Evaluation Metric. To assess the performance of pFedSV, we conduct a thorough evaluation by comparing it against several state-of-the-art PFL algorithms. These algorithms include pFedMe [85], pFedHN [80], FedFomo [110], and FedAMP [39]. Furthermore, to provide a comprehensive understanding of pFedSV's performance, we compare it with classical single global model methods. These methods include FedAvg [63], FedAvg with fine-tuning (FedAvg+FT), FedProx [51], and a simple separate local training approach denoted as "separate." In the "separate" method, each client independently trains its own model without collaboration. We evaluate the performance of all algorithms using the mean testing accuracy (MTA), which represents the average testing accuracy across all clients. Additionally, we report the error range of the MTA after conducting five repeated experiments, denoted by ±. This evaluation framework allows us to comprehensively assess the performance of pFedSV in comparison to other federated learning approaches.

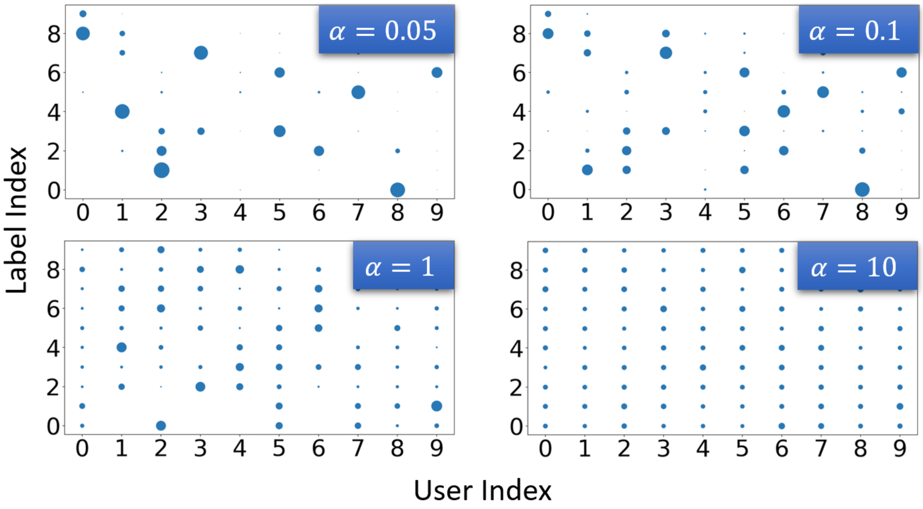


Figure 4.6: The visualization of Dirichlet data Non-IID setting on MNIST, where x-axis indicates the client index, y-axis indicates the label index, and the size of scattered points indicates the number of training samples owned by the client.

Non-IID Data Setting. In our evaluation, we consider two different Non-IID data settings for each dataset. These settings are as follows: 1) Pathological Non-IID data setting: In this setting, each client is randomly assigned two types of labels, and there is no similarity in the private data between any two clients. The characteristics of this setting are illustrated in the table presented in Figure 4.4. 2) Dirichlet Non-IID data setting **Dir**(α): This setting employs different values of α to control the level of data heterogeneity. A small α value indicates high data heterogeneity, resulting in a more biased distribution of labels among the clients. On the other hand, a larger α value reduces the level of heterogeneity and leads to a more balanced label distribution across the clients. The impact of different α values on the clients' data heterogeneity is visually represented in Figure 4.6, providing a clear understanding of this setting. The visualization depicted in Figure 4.6 offers valuable insights into the influence of varying α values on the distribution of labels within the Dirichlet Non-IID data setting, thereby enhancing our understanding of the data heterogeneity present in this setting.

Implementation details. We investigate two federated learning (FL) scenarios with varying client scales: one involving a total of 10 clients with 100% participation and another with a total of 100 clients with 10% participation. In both scenarios, we employ a consistent training configuration, including 5 local epochs, an equal number of communication rounds (20 rounds for the former and 100 rounds for the latter), and specific learning rates (0.01 for MNIST and FMNIST, 0.1 for CIFAR-10) tailored to each dataset. Regarding the hyper-parameters related to the SV (ShareVec) approach, we set the number of Monte Carlo samples as $R = 3|S_{i,k}^t|$, where $|S_{i,k}^t|$ represents the number of model parameters downloaded for each round. Initially, we set $k = 5$ as the number of parameters downloaded. However, it is important to note that the value of k is dynamically adjusted based on the dynamic top-k download mechanism outlined in the "SV for domain relevance" section of our solution (Section 4.3.3).

4.6.2 Performance Analysis

This section aims to present a comprehensive performance evaluation of our proposed pFedSV algorithm, comparing it against state-of-the-art benchmarks and providing a detailed analysis of the experimental results. The objective is to showcase the superior performance and efficacy of pFedSV in PFL.

Table 4.1: The MTA with the pathological Non-IID data setting, where bold indicates the best result among all methods. 10 clients with 100% and 100 clients with 10% participation in each round.

Methods	MNIST		FMNIST		CIFAR-10		CIFAR-100	
	10 clients	100 clients	10 clients	100 clients	10 clients	100 clients	10 clients	100 clients
Seperate	96.11 ± 0.28	93.27 ± 3.68	92.35 ± 0.43	91.42 ± 2.69	84.15 ± 2.13	75.57 ± 4.08	73.57 ± 5.13	68.57 ± 4.25
FedAvg	91.74 ± 1.68	78.46 ± 1.14	90.31 ± 2.49	75.63 ± 4.73	57.67 ± 4.16	44.64 ± 4.75	50.57 ± 3.71	43.16 ± 4.68
FedProx	90.12 ± 0.73	78.45 ± 1.83	90.16 ± 3.05	78.83 ± 3.49	55.68 ± 2.67	45.75 ± 4.39	49.21 ± 3.69	41.08 ± 5.27
IFCA	92.86 ± 1.57	86.73 ± 2.05	90.01 ± 2.38	82.63 ± 3.59	71.69 ± 3.25	60.23 ± 3.94	61.37 ± 4.16	52.44 ± 4.68
FedEM	93.05 ± 1.28	88.53 ± 1.87	91.27 ± 2.68	85.61 ± 4.07	77.38 ± 3.56	65.42 ± 5.07	66.39 ± 5.76	59.84 ± 4.28
FedAvg+FT	94.38 ± 1.06	90.51 ± 1.67	91.18 ± 3.54	89.49 ± 4.51	81.34 ± 3.24	70.13 ± 5.68	71.08 ± 5.14	64.38 ± 4.69
pFedMe	93.75 ± 1.34	86.57 ± 2.61	92.46 ± 1.72	85.39 ± 2.97	80.48 ± 4.59	70.15 ± 5.86	71.56 ± 4.79	60.85 ± 5.26
FedFomo	96.90 ± 0.87	93.71 ± 2.05	94.10 ± 0.65	92.78 ± 1.92	85.93 ± 3.02	74.36 ± 2.15	76.89 ± 3.54	69.21 ± 4.37
FedAMP	95.82 ± 1.37	92.59 ± 1.88	93.26 ± 2.14	91.46 ± 2.04	84.32 ± 3.69	72.91 ± 2.83	75.38 ± 3.19	67.02 ± 4.15
pFedHN	96.53 ± 0.84	94.16 ± 1.38	94.97 ± 0.86	93.69 ± 1.58	86.38 ± 2.72	76.62 ± 3.05	77.24 ± 3.86	70.58 ± 4.57
pFedSV(Ours)	$\mathbf{98.01 \pm 0.83}$	$\mathbf{96.94 \pm 1.75}$	$\mathbf{96.16 \pm 0.58}$	$\mathbf{94.68 \pm 2.36}$	$\mathbf{89.64 \pm 1.88}$	$\mathbf{80.65 \pm 3.78}$	$\mathbf{80.57 \pm 4.37}$	$\mathbf{72.61 \pm 4.73}$

Results on the different Non-IID data setting.

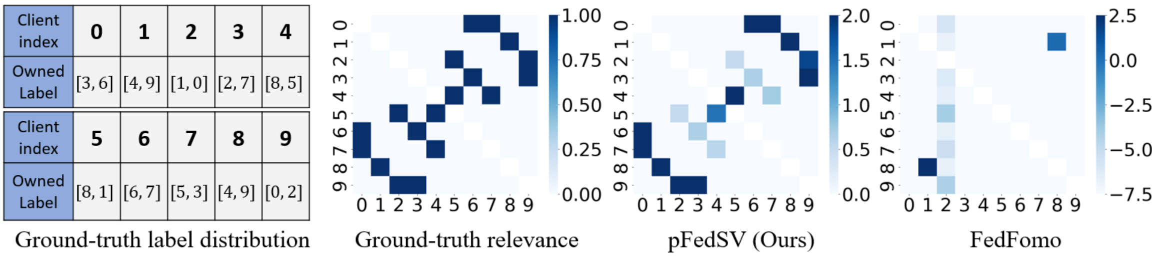
Table 4.1 provides an overview of the MTA achieved by various methods under the pathological Non-IID data setting. In this setting, each client is assigned only two types of labels, thereby simplifying the classification task for individual clients and resulting in high performance for standalone models across all datasets. However, the pathological Non-IID data setting poses a significant challenge for single global model methods. We observe that FedAvg and FedProx experience considerable performance degradation across all datasets. This degradation can be attributed to the inclusion of models from domain-irrelevant clients during global aggregation, leading to instability in the gradient optimization process [111]. On the other hand, the remaining PFL methods, namely FedAvg+FT, pFedMe, FedFomo, FedAMP, pFedHN, and our proposed pFedSV, demonstrate promising performance across all datasets. FedAvg+FT incorporates several local fine-tuning steps to adapt the poor global model to the local Non-IID data distribution. pFedMe introduces novel regularized loss functions based on Moreau envelopes to decouple personalized optimization from global model

Table 4.2: The MTA with the Dirichlet Non-IID data setting ($\alpha = 0.1$) on different datasets, where bold indicates the best result among all methods. 10 clients with 100% and 100 clients with 10% participation in each round.

Methods	MNIST		FMNIST		CIFAR-10		CIFAR-100	
	10 clients	100 clients	10 clients	100 clients	10 clients	100 clients	10 clients	100 clients
Seperate	74.05 ± 2.11	59.81 ± 5.73	60.18 ± 6.42	58.22 ± 6.73	40.53 ± 7.20	36.15 ± 6.88	35.43 ± 3.87	30.05 ± 5.49
FedAvg	43.57 ± 3.75	30.15 ± 4.82	40.58 ± 4.16	36.49 ± 5.07	33.81 ± 5.07	26.82 ± 6.43	26.17 ± 4.27	20.33 ± 5.27
FedProx	47.49 ± 4.18	44.76 ± 5.49	43.09 ± 4.82	40.34 ± 4.72	35.76 ± 5.18	29.91 ± 5.58	29.62 ± 5.13	23.27 ± 4.69
IFCA	58.67 ± 2.69	54.58 ± 3.79	56.29 ± 4.16	51.04 ± 4.38	43.09 ± 4.88	40.67 ± 4.86	39.28 ± 4.11	31.89 ± 4.20
FedEM	66.53 ± 2.74	60.28 ± 4.05	61.79 ± 3.62	57.41 ± 4.28	51.09 ± 4.58	45.82 ± 5.07	41.39 ± 3.76	35.88 ± 4.61
FedAvg+FT	65.72 ± 3.84	55.57 ± 4.26	57.27 ± 4.13	52.83 ± 5.01	43.42 ± 5.29	40.05 ± 5.22	36.33 ± 3.86	32.55 ± 4.37
pFedMe	64.39 ± 4.08	58.02 ± 3.51	60.27 ± 3.59	56.81 ± 4.01	50.73 ± 4.29	44.21 ± 5.09	40.29 ± 3.57	34.94 ± 3.78
FedFomo	72.54 ± 2.18	63.07 ± 2.54	64.75 ± 3.42	60.49 ± 3.72	53.83 ± 4.57	48.35 ± 5.29	45.91 ± 3.06	37.51 ± 3.09
FedAMP	70.15 ± 3.02	60.28 ± 3.11	62.28 ± 2.53	58.94 ± 3.14	51.57 ± 4.03	46.05 ± 4.48	43.67 ± 3.55	36.40 ± 3.76
pFedHN	73.35 ± 2.04	62.57 ± 4.11	62.95 ± 3.44	59.55 ± 4.15	52.82 ± 3.88	47.19 ± 5.83	45.33 ± 3.45	37.38 ± 3.77
pFedSV(Ours)	$\mathbf{78.17 \pm 1.59}$	$\mathbf{70.76 \pm 2.41}$	$\mathbf{71.47 \pm 1.86}$	$\mathbf{66.63 \pm 2.03}$	$\mathbf{61.18 \pm 1.67}$	$\mathbf{56.76 \pm 1.85}$	$\mathbf{50.46 \pm 2.47}$	$\mathbf{42.25 \pm 3.13}$

learning. pFedHN generates personalized parameters for each client's model through an additional hypernetwork. FedFomo and FedAMP achieve good performance by facilitating adaptive pairwise collaboration between clients with similar models to create personalized models. Our pFedSV surpasses all other baseline methods by considering multiwise influences among clients, thereby helping them identify their domain-relevant coalition and generating personalized aggregation weights through multiwise collaboration. This approach enables pFedSV to achieve superior performance in personalized federated learning scenarios.

Table 4.2 showcases the MTA of all methods under the Dirichlet Non-IID data setting with $\alpha = 0.1$. As evident from the visualization presented in Figure 4.6, this setting poses a greater challenge compared to the pathological Non-IID setting, resulting in a significant performance reduction for all methods. However, even in this challenging scenario, our pFedSV consistently outperforms all other baseline methods. It demonstrates its superiority by achieving higher MTA compared to the other algorithms evaluated. It is noteworthy that the lower accuracy observed in Table 4.2 for the case



Client index	0	1	2	3	4
Owned Label	[3, 6]	[4, 9]	[1, 0]	[2, 7]	[8, 5]
Client index	**5**	**6**	**7**	**8**	**9**
Owned Label	[8, 1]	[6, 7]	[5, 3]	[4, 9]	[0, 2]

Ground-truth label distribution

Ground-truth relevance

pFedSV (Ours)

FedFomo

Figure 4.7: The left chart shows the client label distribution obtained from the omniscience perspective. Right side is the visualization of clients' relevance matrix $\phi^i, i \in N$ on different algorithms with the pathological MNIST Non-IID setting after convergence. x-axis and y-axis is the client index.

with 100 clients is due to the fact that only 10% of the clients participate in each round. This limited participation rate contributes to the decreased overall accuracy in the evaluation.

Relevance score & Multiwise collaboration weights.

The superior performance of pFedSV in domain relevance identification can be attributed to the desirable properties of SV. Figure 4.7 provides a visualization of the relevance vector ϕ^i for each client after convergence, demonstrating the impact of different algorithms. For instance, FedFomo utilizes model similarity-based weights to update the relevance vector, whereas pFedSV leverages the computed SV from its local model coalition game. To further illustrate the effectiveness of the pFedSV algorithm, we include a visualization of the ground-truth client relevance obtained from an omniscient perspective, based on the client label distribution table. From the ground-truth visualization, it becomes apparent that symmetry is a crucial property of the client relevance matrix. Our pFedSV algorithm excels at identifying all domain-relevant clients and assigning aggregation weights through multiwise collaboration within the coalition. In contrast, FedFomo does not guarantee precise relevance identification. The visualized results provide compelling evidence of pFedSV's capability to accurately identify domain-relevant clients and leverage multiwise collaboration to assign appropriate aggregation weights. This contributes to its superior performance in domain relevance identification compared to alternative methods like FedFomo.

4.6.3 Communication Overhead & Model Privacy

Communication Overhead Reduction

We have two different mechanisms in this chapter to reduce the communication overhead: dynamic top-k download mechanism and shared common feature extractor. Therefore, to compare the communication overhead under different cases, we adopt the following baselines:

1) **pFedSV (D+C)**: It means we adopt both the **D**ynamic top-k download mechanism and **C**ommon feature extractor in pFedSV to reduce the communication overhead.

2) **pFedSV (D)**: It means we only adopt the **D**ynamic top-k download mechanism in the main content to reduce the communication overhead.

3) **FedFomo**: It downloads the whole model of other clients and performs personalization on the local side of each client [110].

4) **FedAMP**: it performs the personalization on the server side and directly distributes the personalized model to each client, whose communication overhead is equal to FedAvg [39].

5) **FedAvg**: traditional FL algorithm that downloads one global model to each client [63].

All algorithms are implemented with the following setups: total 20 communication rounds, 10 clients with 100% participation in each round, pathological Non-IID data distribution. We use the number of model parameters that are required in upload and download as the measurement metric for communication overhead.

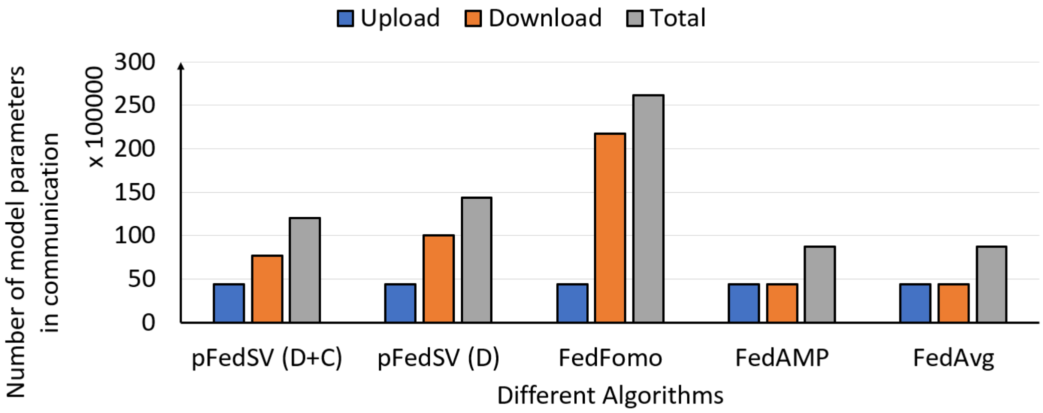


Figure 4.8: LeNet-5: Communication overhead comparison on LeNet-5 with different algorithms. The y-axis indicates the number of model parameters in the communication.

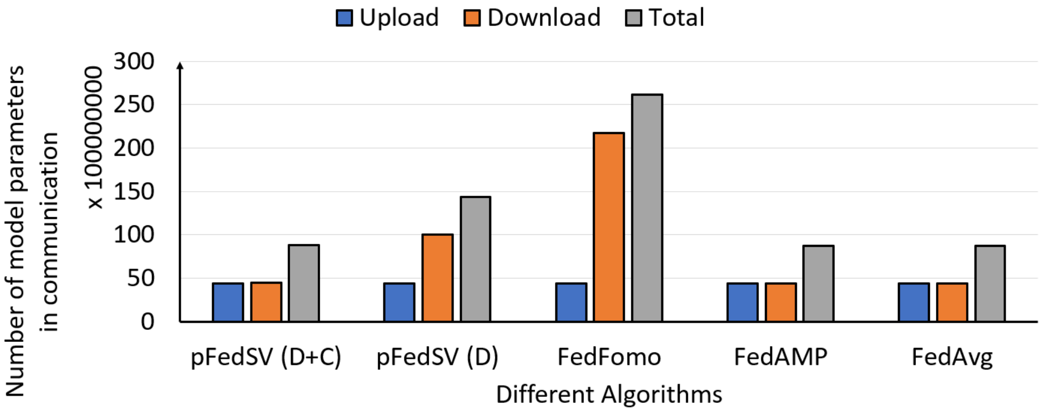


Figure 4.9: ResNet: Communication overhead comparison on ResNet-V1-34-layer(Plain) with different algorithms. The y-axis indicates the number of model parameters in the communication.

In Fig. 4.8, we show the communication overhead comparison of different baselines on the LeNet-5 Model. Besides, to further illustrate the effectiveness of our Top-k

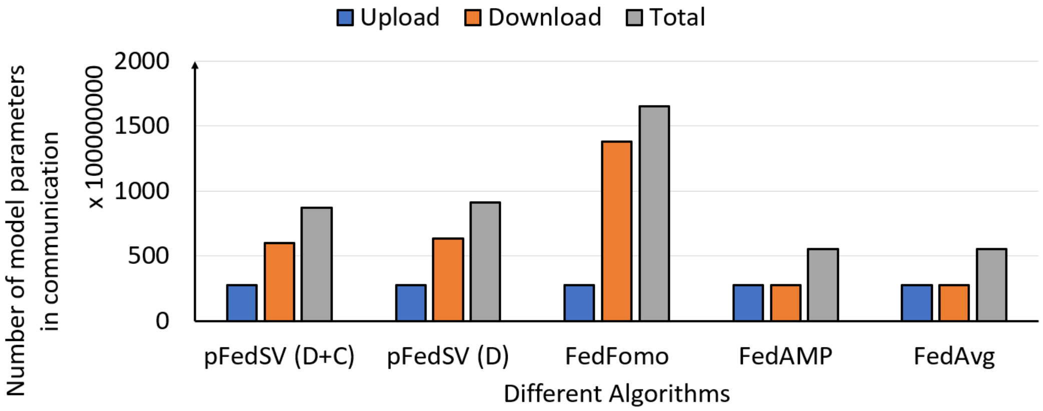


Figure 4.10: VGG-19: Communication overhead comparison on VGG-19 with different algorithms. The y-axis indicates the number of model parameters in the communication.

dynamic download mechanism and shared common feature extractor in communication overhead reduction, we also compute the communication overhead comparison on other different models, including ResNet-V1-34-layer(Plain) in Fig. 4.9 and VGG-19 in Fig. 4.10.

You can find that the communication overhead of pFedSV at ResNet case is almost the same as traditional FedAvg. The reason is that, as a powerful pre-trained model, most model parameters in ResNet are the convolutional layer-based feature extractor, and the classifier-related parameters only account for 2.3% of the overall model parameter number. Thus using a shared common feature extractor can significantly save extensive communication overhead. In contrast, for traditional CNN model such as LeNet-5, the classifier-related parameters can account for 49.57% of the overall model parameter number. Therefore, with the help of shared feature extractor, the additional communication can be significantly reduced in ResNet case. Moreover, the results on VGG-19 are for your additional reference, where the classifier-related parameters can account for 89.74% of the overall model parameter number in VGG-19.

As expected, the communication overhead of FedFomo is much higher than other algorithms. Our dynamic download mechanism can efficiently reduce it by rapidly identifying the domain-relevant clients and adjusting the model download number, which is illustrated in the main content. Besides, the introduced common feature extractor can further reduce the communication overhead in the download part. Finally, FedAMP has the same communication overhead as FedAvg. Although the communication overhead of our pFedSV (D+C) is not the lowest compared to FedAMP, we can achieve higher personalized performance for each client, which is an acceptable trade-off.

Table 4.3: The results of pFedSV with DP, which illustrates that we can maintain the personalized accuracy with a reasonable privacy budget.

Methods	δ	σ	CIFAR-10		CIFAR-100	
			ϵ	Accuracy	ϵ	Accuracy
FedAvg	1×10^{-5}	0	∞	19.68 ± 1.76	∞	5.21 ± 0.41
FedAvg	1×10^{-5}	1	11.28 ± 0.32	17.54 ± 1.37	8.47 ± 0.67	5.03 ± 0.24
FedAvg	1×10^{-5}	2	3.64 ± 0.13	15.97 ± 1.53	2.56 ± 0.19	4.37 ± 0.19
pFedSV	1×10^{-5}	0	∞	84.73 ± 1.67	∞	31.07 ± 1.22
pFedSV	1×10^{-5}	1	**5.97 ± 0.11**	**82.16 ± 1.55**	**8.42 ± 0.71**	**30.59 ± 1.06**
pFedSV	1×10^{-5}	2	1.82 ± 0.05	78.29 ± 1.63	1.80 ± 0.16	23.44 ± 0.89

Model Privacy Protection

In our experiments, we consider a task with the pathological Non-IID data setting on the CIFAR-10 and CIFAR-100 datasets. We utilize 10 clients with 100% participation in each round. The objective is to compare the performance of pFedSV with FedAvg under varying levels of Gaussian noise (σ), while keeping all other parameters fixed. The results presented in Table 4.3 demonstrate that increasing σ enhances privacy (lower ϵ values) at the expense of decreased performance (indicated in bold in the table). Furthermore, the experimental findings in Table 4.3 indicate that introducing aggressive noise leads to a reduction in accuracy (from 84.73% to 78.29%). However, by adopting an appropriate level of noise ($\delta = 1$), model privacy can be

Table 4.4: The MTA comparison of pFedSV with DP-based noise addition and some selected baselines (other omitted baselines can refer to Table 4.2). The experiments are conducted with Dirichlet Non-IID data setting ($\alpha = 0.1$). 10 clients with 100% and 100 clients with 10% participation. We emphasize our pFedSV and the pFedSV+DP in bold.

Methods	MNIST		FMNIST		CIFAR-10		CIFAR-100	
	10 clients	100 clients	10 clients	100 clients	10 clients	100 clients	10 clients	100 clients
Seperate	74.05 ± 2.11	59.81 ± 5.73	60.18 ± 6.42	58.22 ± 6.73	40.53 ± 7.20	36.15 ± 6.88	35.43 ± 3.87	30.05 ± 5.49
FedAvg	43.57 ± 3.75	30.15 ± 4.82	40.58 ± 4.16	36.49 ± 5.07	33.81 ± 5.07	26.82 ± 6.43	26.17 ± 4.27	20.33 ± 5.27
FedFomo	72.54 ± 2.18	63.07 ± 2.54	64.75 ± 3.42	60.49 ± 3.72	53.83 ± 4.57	48.35 ± 5.29	45.91 ± 3.06	37.51 ± 3.09
FedAMP	70.15 ± 3.02	60.28 ± 3.11	62.28 ± 2.53	58.94 ± 3.14	51.57 ± 4.03	46.05 ± 4.48	43.67 ± 3.55	36.40 ± 3.76
pFedSV(Ours)	**78.17 ± 1.59**	**70.76 ± 2.41**	**71.47 ± 1.86**	**66.63 ± 2.03**	**61.18 ± 1.67**	**56.76 ± 1.85**	**50.46 ± 2.47**	**42.25 ± 3.13**
pFedSV+DP	**76.58 ± 1.32**	**68.43 ± 1.86**	**69.24 ± 2.07**	**65.31 ± 1.95**	**57.94 ± 2.53**	**53.28 ± 1.66**	**48.37 ± 2.51**	**40.79 ± 2.84**

protected while only causing a minor impact on accuracy (from 84.73% to 82.16%). To further validate the effectiveness of DP-based methods in addressing privacy concerns, we conduct additional experiments. Specifically, we compare the performance of our pFedSV+DP with other personalized baselines under an appropriate noise level ($\delta = 1$), as shown in Table 4.4. The results reinforce that DP-based methods remain effective in addressing privacy issues, and our pFedSV+DP outperforms other personalized baselines. Overall, the experimental results highlight that increasing the level of noise improves privacy while sacrificing performance. However, by carefully selecting an appropriate noise level, such as $\delta = 1$, model privacy can be protected with only a minimal impact on accuracy. Furthermore, DP-based methods, including our pFedSV+DP, continue to demonstrate their effectiveness in addressing privacy concerns, as validated by numerous studies in the field.

Besides, the key privacy issue concern of our pFedSV algorithm comes from the fact that the local model of each client will be downloaded to other clients, since the recent research [112] on model inversion attacks shows that malicious attackers can recover the raw training data from the model through the gradients only. Therefore, to further demonstrate the effectiveness of our DP-based noise addition on privacy

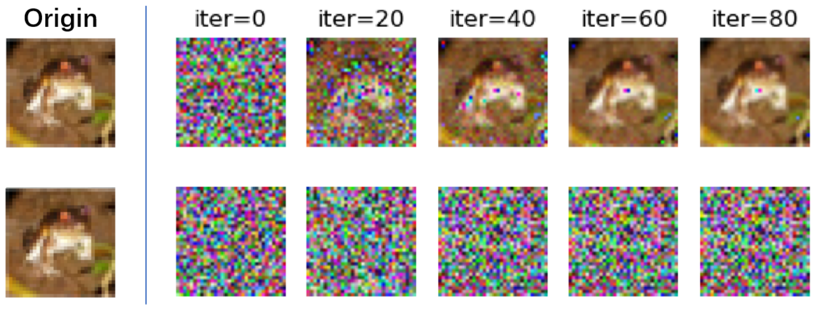


Figure 4.11: The model inversion attack results on the original model and the model with DP noise addition ($\delta = 1$) on the CIFAR-10 dataset. *The top column* is the original model, where the attacker can recover the raw training data with the shared model parameters. *The bottom column* is the model with DP noise addition, where the attack failed.

protection, we conduct an extra experiment with model inversion attack on both the original local model and the model with DP-based noise addition, where the results are illustrated in Fig. 4.11. We can see that the attacker cannot recover the raw training data after we add the DP-based noise into the original model.

4.7 Remarks

In this chapter, we focus on the model personalization of clients with heterogeneous domains in an agnostic federated learning system. we propose pFedSV, a novel personalized FL algorithm that incorporates the Shapley value from coalition game theory to assess intricate, multi-faceted influences by quantifying the individual contributions of each client. We provide a complex analysis by formulating the model aggregation process as a coalition game, which not only helps form the personalized domain-relevant coalition but also serves as personalized aggregation weights for each client. Extensive experiments are conducted to demonstrate the effectiveness of pFedSV and the results empirically illustrate its superiority through the significant improvement on personalized accuracy. Furthermore, regarding the communication overhead and model privacy issues raised by the local model download mechanism in pFedSV, we introduce the shared common feature extractor and the DP-based noise addition, respectively.

4.8 Discussion

In the chapter, we introduce the concept of SV from the cooperation game theory and design a personalized federated learning algorithm based on it. Although we achieve a good personalized model performance, it comes with a relative high computational cost from the calculation of SV. And this computational cost is not negligible after applying the Monte-Carlo sampling technique to reduce the calculation times. Therefore, we expect the following future work can find novel optimization directions from two aspects: 1) propose a more efficient way for SV calculation to reduce the computation cost. 2) using the advanced ML technique to directly predict the SV based on the historical information.

Chapter 5

Federated Unlearning: Guarantee the Right of Clients to Forget

The Right to be Forgotten encompasses the entitlement of a data proprietor to withdraw their data from an entity that stores it. Within the domain of federated learning, adherence to the Right to be Forgotten necessitates the eradication not only of the data itself but also of any influence exerted by said data on the federated learning (FL) model. We refer to this process as federated unlearning. The most direct and legitimate approach to implementing federated unlearning involves the removal of the revoked data, followed by the complete retraining of the FL model. However, the computational and temporal burdens associated with fully retraining FL models can prove to be excessively costly. This research paper represents an initial endeavor towards a comprehensive exploration of the unlearning paradigm within the context of federated learning. Initially, we define the problem of efficient federated unlearning, outlining its challenges and objectives. Additionally, we identify three prevalent types of federated unlearning requests, specifically class unlearning, client unlearning, and sample unlearning. Drawing upon the aforementioned challenges and objectives, we propose a general pipeline for federated unlearning that addresses the aforementioned types of requests. Furthermore, we reexamine the manner in which training data influences the performance of the final FL model. Consequently, we enhance the proposed framework by incorporating reverse stochastic gradient ascent (SGA) and elastic weight consolidation (EWC). To validate the efficacy and efficiency of the proposed method, we conduct various experiments that assess its performance in terms of unlearning. The results obtained from these experiments affirm the effectiveness of the proposed approach. We anticipate that the proposed method will serve as an indispensable component within future machine unlearning systems.

5.1 Introduction

Edge computing is facilitating the convergence of mobile phones and Internet of Things (IoT) devices, leading to a transformation of the prevailing computing platform. This shift is paving the way for the emergence of next-generation intelligent services that rely on machine learning techniques. In this context, Google has recently introduced Federated Learning (FL) [63], an innovative distributed machine learning paradigm. FL encourages clients to collectively train a shared global model, while also leveraging the capabilities of edge devices to address concerns related to privacy, security, and regulatory compliance [113].

Numerous applications of federated learning entail the analysis of data generated by individuals on their respective local devices. This data frequently comprises sensitive information, encompassing but not limited to diagnostic records, bank statements, and facial images. Furthermore, federated learning operates on dynamic local data, which undergoes continuous changes throughout the training process for each client. New data is regularly generated and incrementally utilized to refine existing models, aligning with the principles of lifelong learning [102].

Conversely, there are instances where data may need to be deleted as per requirements. Recent privacy regulations, such as the General Data Protection Regulation (GDPR) implemented by the European Union and the former Right to be Forgotten, grant clients the right to remove specific data from trained models. Apart from compliance with the Right to be Forgotten, data removal from federated learning (FL) models is also advantageous when certain training data becomes obsolete over time. These practical requirements have given rise to the need for efficient techniques that allow FL models to unlearn or forget the knowledge acquired from data that needs to be eliminated. These techniques, known as Machine Unlearning, have gained significant attention both in academic research and industry applications [30, 28, 8].

The most legitimate approach to implementing machine unlearning involves the removal of the requested data and subsequently retraining the federated learning (FL) model from the beginning. However, it is important to note that the computational and time costs associated with fully retraining FL models in response to the erasure of training data can be excessively burdensome and may pose practical challenges.

This chapter delves into the efficient implementation of machine unlearning within the framework of federated learning, which is referred to as *Federated Unlearning*. Initially, the problem of federated unlearning is defined by highlighting the associated challenges and goals. These challenges specifically pertain to iterative learning, stochastic training, and data isolation within the context of federated learning. Based on these challenges, the goals of the federated unlearning problem revolve around accuracy, unlearning privacy, model agnosticism, and unlearning efficiency. Furthermore, three common types of federated unlearning requests are identified in the context of federated learning: class unlearning, client unlearning, and sample unlearning. To address these diverse unlearning requests using a unified framework, we reexamine how training data impacts the performance of the final federated learning model in a conventional training process based on gradient descent. This analysis provides a deeper understanding of the dominant factors involved in the federated unlearning process. Consequently, a general federated unlearning framework based on reverse stochastic gradient ascent (SGA) is proposed to effectively eliminate the influence of specific training data.

The initial version of the reverse stochastic gradient ascent (SGA) demonstrates satisfactory performance when addressing class unlearning requests. However, for client unlearning and sample unlearning requests, we integrate the capabilities of elastic weight consolidation (EWC) with the basic SGA. This integration gives rise to the SGA-EWC-based federated unlearning framework [46]. To evaluate the effectiveness of our proposed method, experiments are conducted using the widely adopted federated learning setting called FedAvg. The results obtained from these experiments indicate that our method achieves strong performance in terms of both unlearning efficacy and efficiency.

The main contributions of this chapter are summarized as:

- We take an early step in thoroughly exploring the machine unlearning paradigm within the context of federated learning. It achieves this by defining the problem of efficient federated unlearning, encompassing the various requests, challenges, and goals involved.

- We propose a general pipeline for federated unlearning that addresses three distinct types of requests: class unlearning, client unlearning, and sample unlearning. By revisiting how training data influences the final federated learning model's performance, the pipeline is designed to effectively tackle these different types of unlearning requests.

- Various experiments are conducted to validate the effectiveness of the proposed method. These experiments assess the unlearning efficacy and efficiency of the method, providing empirical evidence of its performance.

5.2 Background of Machine Unlearning and Federated Learning

5.2.1 Machine Unlearning

Indeed, in various domains such as healthcare, movie recommendation, and other applications, enterprises and organizations collect clients' data to train machine learning models. While these technologies offer convenience and promising outcomes, they also raise concerns about the potential leakage of personal information. As a result, as data privacy gains increased attention, clients often seek to delete or conceal their personal data once the service is no longer required. The most straightforward approach in such cases is to delete the data of specific clients from the dataset and retrain a new model from scratch using the remaining data. However, this approach is impractical due to the high cost associated with retraining models after every deletion request, both in terms of time and resources. To address this issue, machine unlearning has been proposed as a solution. Machine unlearning techniques aim to remove the knowledge acquired from the data that needs to be deleted from the model, without requiring the costly process of retraining the model from scratch. By selectively unlearning specific data, machine unlearning provides an efficient alternative to fully retraining models in response to data deletion requests.

In order to minimize the cost of unlearning and maintain model performance, several strategies have been developed for data deletion from machine learning models. One approach involves post-processing the trained model to ensure that the results of the unlearning algorithm are statistically similar to those of a retrained model [30, 28]. This can be achieved by updating the model to minimize the empirical loss on the remaining data. Another approach is to develop new training algorithms that reduce the need for complete retraining. A critical technique in this regard is ensemble learning, where the entire dataset is divided into multiple shards, and a separate sub-model is trained on each shard [77]. The unlearning process can then be accomplished by retraining these sub-models, including the revoked data, which significantly reduces the computational resource and memory storage requirements [8]. In this chapter, the focus is on extending machine unlearning to distributed scenarios, where the data points that need to be deleted are distributed across multiple devices. This distributed setting introduces additional challenges and considerations for unlearning techniques.

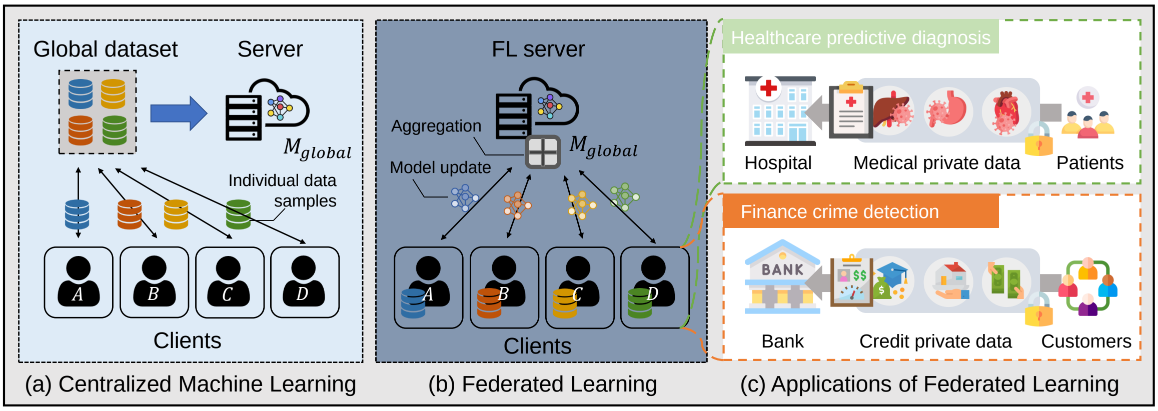


Figure 5.1: Comparison of centralized machine learning and federated learning.

5.2.2 Federated Learning

Federated Learning has emerged as an efficient paradigm for training machine learning models collaboratively among multiple clients while preserving the privacy of their local raw data [63, 109]. In federated learning, the traditional centralized approach of training a machine learning model on a central server is replaced by a distributed approach. Fig. 5.1 provides an illustration of the detailed workflow of federated learning compared to centralized machine learning.

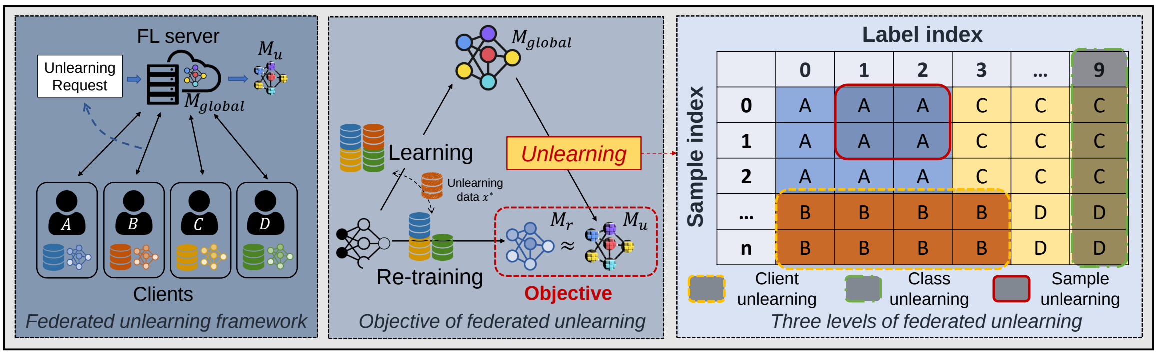
In a typical federated learning (FL) algorithm, such as FedAvg [63], the training process proceeds iteratively in rounds. Initially, the model is initialized on the server. Subsequently, in each communication round, the server randomly selects a subset of clients from the available pool. The latest global model is then distributed to the selected clients. These clients perform local model updates on their respective data using the stochastic gradient descent (SGD) algorithm for multiple epochs. After completing the local update, the clients transmit their local model parameters back to the server. The server aggregates these parameters to obtain a new global model by means of averaging. The updated global model is subsequently distributed to the clients for the next round of training. This iterative process continues until the global model achieves convergence or satisfies other specific requirements.

In contrast to traditional distributed machine learning approaches that involve sharing potentially sensitive client information, Federated learning overcomes data barriers while ensuring privacy preservation. Its ability to amalgamate client contributions has made federated learning widely adopted in diverse real-world applications, such as keyboard prediction [35] and healthcare [10]. However, challenges persist in the context of federated learning, particularly concerning non-independent and identically distributed (Non-IID) data and the significant communication overhead imposed by a massive number of distributed clients. Addressing these challenges and devising effective solutions to enhance client data privacy and enhance training model performance through enabling technologies represent promising directions in distributed scenarios.

5.3 From Machine Unlearning to Federated Unlearning

5.3.1 What is Federated Unlearning?

Federated unlearning focuses on a federated learning (FL) scenario where clients collaborate with an FL server to train and maintain a global model. However, cer-



	0	1	2	3	...	9
0	A	A	A	C	C	C
1	A	A	A	C	C	C
2	A	A	A	C	C	C
...	B	B	B	B	D	D
n	B	B	B	B	D	D

Figure 5.2: Overall architecture of federated unlearning.

tain clients may subsequently request the removal of privacy-sensitive or illegal data contributions from the global model to safeguard privacy or mitigate legal risks. Consequently, the server needs to update the model so that it appears as if the deleted data never participated in the FL training process. For instance, as depicted in Fig. 5.2, four clients with distinct local data (represented by different colors) participate in FL training. Once a well-trained global model, denoted as M_{global}, is obtained, Client B submits an unlearning request to eliminate its data contribution from the global model. To fulfill this request, the FL server must provide the clients with a new model, denoted as M_u, which remains unaffected by any of Client B's data. A straightforward approach would involve deleting the requested data and retraining a new model from scratch using only the remaining data. However, the computational and time overhead associated with retraining can be prohibitively expensive and unacceptable. Hence, there is an urgent need for a precise and efficient method to meet the unlearning objective, as illustrated in the middle of Fig. 5.2. Such a method should enable the removal of specific client data from the global model without necessitating complete retraining, thereby minimizing the computational and time overheads involved.

Federated unlearning holds potential for various FL applications that prioritize data ownership. For instance, in a healthcare application where multiple hospitals collaborate to develop a diagnostic predictive model using FL, a hospital may need to remove specific data from the model if a patient decides to withdraw from data sharing and no longer provides their data. Similarly, in the field of finance, banks can collectively train an FL model for the detection of financial crimes. If a client requests to close their account, a bank may need to remove the corresponding client's data from the model.

5.3.2 Goals of Federated Unlearning

The goals of federated unlearning is described as follows.

Goal 1. Zero contribution. *The primary objective of federated unlearning is to achieve "zero contribution", which denotes that the removal of data should result in no influence on the unlearned model's parameters. Specifically, the deleted data should have no impact on the model's parameters, ensuring that the model's predictions on the deleted data remain consistent with those of a model that was not trained using the deleted data.*

Goal 2. Accuracy. *The concept of "accuracy" in federated unlearning refers to the objective of minimizing the accuracy gap between the unlearned model and the baseline model, regardless of the number of data points that are unlearned. The goal is to ensure that the unlearned model does not experience significant degradation in accuracy when applied to other data, thereby maintaining a high level of predictive performance.*

Goal 3. Unlearning privacy. *The notion of "unlearning privacy" in federated unlearning pertains to the requirement that the unlearning process should not lead to any privacy breaches. Specifically, it ensures that the deleted data of clients remains protected and cannot be recovered by potential attackers, even in the presence of gradient leakage attacks during the unlearning process. This objective aims to safeguard the privacy of client data and prevent any unauthorized access or disclosure of sensitive information.*

Goal 4. Model Agnostic. *The principle of "Model Agnostic" in the context of federated unlearning postulates that the proposed technology should be applicable to various federated learning (FL) models, irrespective of their diverse characteristics and complexities. In essence, it necessitates the capability of the unlearning methodology to be universally employed across FL models, without being constrained by their specific nature or intricacy. This quality ensures the versatility and adaptability of the federated unlearning technology, enabling its seamless integration into different FL scenarios and accommodating the specific requirements and nuances of various FL models.*

Goal 5. Unlearning efficiency. *The concept of "Unlearning efficiency" in federated unlearning refers to the objective of developing a technology that is more efficient than the retraining baseline, regardless of the amount of data that needs to be forgotten. The goal is to minimize the computational resources and time required for the unlearning process, striving to achieve a higher level of efficiency compared to the traditional retraining approach. This efficiency objective is crucial in order to make the federated unlearning technology practical and feasible, enabling seamless and effective removal of data from the model without incurring excessive computational or time costs.*

5.3.3 Why is Federated Unlearning Challenging?

While many of the goals in federated unlearning align with those in traditional machine learning, the existing technologies for machine unlearning cannot be directly applied in the context of federated learning (FL) for three primary reasons, as discussed in Section 5.2.

Iterative Learning. The iterative learning process in FL poses a fundamental challenge for federated unlearning. In FL, each client's initial model for a training round is derived from the aggregation of models from all clients in the previous round. This interdependence and intertwining of clients' information make it difficult to isolate and remove specific data contributions during the unlearning process. The iterative nature of FL necessitates novel approaches and techniques for unlearning that can effectively address this challenge and ensure the successful removal of data while maintaining the integrity and accuracy of the global model.

Stochastic Training. The stochastic nature of FL training poses another significant hurdle for federated unlearning. Unlike centralized machine learning, the FL training process is highly non-deterministic. In each training round, the FL server randomly selects clients for global model aggregation, and each client independently and randomly selects and orders batches of data for local training. This inherent randomness introduces challenges in unlearning as the exact data contributions from specific clients or batches become difficult to trace or isolate. Unlearning techniques in FL must account for this stochasticity and develop strategies to accurately identify and remove the desired data contributions while preserving the integrity and performance of the global model.

Data Isolation. Client data is protected for privacy reasons in FL, and the FL server lacks direct access to individual client data. Each client retains its own data samples and conducts local model training. However, employing a data splitting-based machine unlearning approach poses significant challenges in FL due to the distributed nature of data and privacy concerns. Such an approach would necessitate substantial storage space, placing a considerable burden on all clients within the system [8].

5.3.4 Possible Research Directions of Federated Unlearning

Consider the data samples illustrated in the right portion of Fig. 5.2. The cell contents indicate both the client to which each data sample belongs and the corresponding label. In accordance with the desired level of forgotten data specified by the clients, research directions for federated unlearning can be categorized into the following three levels:

Class Unlearning. Clients aim to eliminate specific classes of data from the trained model. For instance, as depicted in Fig. 5.2, Clients C and D choose to unlearn all data samples associated with the class label 9.

Client Unlearning. A client intends to remove all of its data from the trained model. As shown in Fig. 5.2, Client B decides to unlearn all of its data samples.

Sample Unlearning. A client seeks to remove a subset of its data from the trained model, which is a more fine-grained and challenging task compared to client unlearning. For example, as indicated in Fig. 5.2, Client A selects a subset of its data samples for unlearning.

5.4 A General SGA-based Federated Unlearning Framework for Different Levels

In this paper, we examine the impact of training data on model performance and propose a federated unlearning framework based on reverse stochastic gradient ascent (SGA) to eliminate the influence of specific training data. Our framework effectively counters the effects of individual data points by iteratively adjusting model parameters in the opposite direction of the gradients computed from the data. This unlearning approach improves model generalization and contributes to enhanced privacy, accuracy, and efficiency in federated learning.

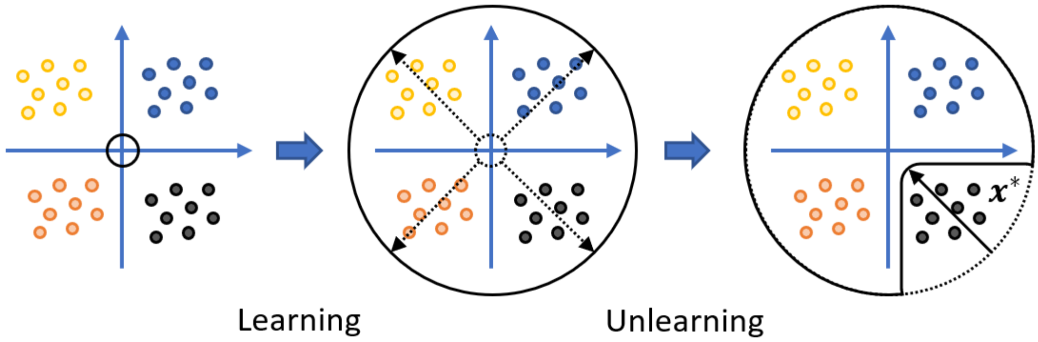


Figure 5.3: The visual representation showcases the alteration of the model's generalization boundary during both the learning and unlearning stages. Distinct colors denote various labeled data points, while black circles delineate the model's generalization boundary, with solid and dashed lines denoting the boundary's state post-update and pre-update, respectively. Additionally, arrows depict the corresponding gradients. Notably, black dots labeled with $\boldsymbol{x}^*$ signify unlearning data points.

5.4.1 Framework Overview

The prevailing model training methods in machine learning, largely rooted in gradient descent, commonly employ mini batch-based stochastic gradient descent (SGD) as the preferred approach [6]. For instance, in the task of image classification using the MNIST dataset, the objective is to develop a model capable of accurately classifying handwritten numeric images ranging from 0 to 9. In this context, gradient descent can be viewed as a learning process where the model progressively acquires knowledge about the shared characteristics within each class and the distinguishing features across different classes present in the training dataset. Consequently, the model's generalization boundary expands to encompass more data, signifying its ability to classify the included instances.

Alternatively, gradient descent can be seen as a form of learning, and reverse gradient ascent can be understood as an opposing process that contracts the generalization boundary, effectively eliminating the model's capacity to classify specific data points.

Figure 1 provides an illustration of the changes in the model's generalization boundary during the SGD learning and SGA unlearning processes. Removing a data point, denoted as $\boldsymbol{x}^*$ (represented by black dots), from the generalization boundary implies that the model discards all prior knowledge of $\boldsymbol{x}^*$ and loses its ability to classify it. This concept of unlearning is central to our proposed federated unlearning framework based on SGA. By performing a few SGA iterations using the unlearning data $\boldsymbol{x}^*$ on the global model M_{global}, our framework efficiently achieves unlearning, yielding an unlearning model M_u in a rapid and effective manner.

Finally, the SGA-based unlearning framework comprises two stages: 1) Learning stage: The standard federated learning (FedAvg) process is executed until model convergence, resulting in a global model denoted as M_{global}. 2) Unlearning stage: SGA-based unlearning is then performed to derive the unlearning model M_u. In the learning stage, FedAvg is employed for model training until convergence. Subsequently, in the unlearning stage, SGA is utilized to eliminate specific training data effects and obtain the unlearning model.

5.4.2 Methodology for Class Unlearning

In class-level federated unlearning, the objective is to exclude a specific class from the model's generalization boundary, thereby rendering the global model distributed to all clients incapable of classifying that particular class. To achieve this goal, a simple SGA-based unlearning approach is sufficient to control the model's generalization boundary. In the traditional federated learning scenario, there is typically a testing dataset available on the server-side, consisting of labeled data from all classes. This testing dataset can be utilized to implement SGA-based unlearning. This assumption is reasonable because, even in the absence of the testing dataset, previous research has demonstrated the generation of synthetic data with similar features on the server-side using techniques like Generative Adversarial Networks (GANs) [29]. In the end, SGA-based unlearning is applied to eliminate the classification capability of the specific class from the global model. The "unlearning data" $\boldsymbol{x}^*$ refers to the data with a specified unlearning label, which can be either real data or synthetic data generated on the server-side.

5.4.3 Methodology for Client Unlearning

In client-level federated unlearning, the objective is to remove the previously acquired knowledge of the global model from a specific client referred to as the "unlearning client" C_u. This entails eliminating the influence of the unlearning client's local data, denoted as the "unlearning data" $\boldsymbol{x}^*$.

Why simple SGA fails in client unlearning. In extreme non-IID cases, where the unlearning client (C_u) exclusively possesses all the data of a specific class, client unlearning can be considered equivalent to class unlearning, and simple SGA (Stochastic Gradient Ascent) is sufficient for achieving the unlearning objective. However, practical scenarios typically involve non-IID data distributions that are not highly skewed. In such scenarios, where multiple clients share a certain class of data, simple SGA-based unlearning is inadequate to meet the requirements of client unlearning. This is due to the following reasons: Firstly, the performance of a well-trained global model relies on its ability to generalize well, irrespective of the source of the data. Consequently, it is not possible to obtain a model that exhibits poor accuracy on label 9 data from client A, while simultaneously achieving good accuracy on the same label 9 data from client B. Such a scenario would indicate an extreme case of overfitting. Secondly, the proportion of data plays a significant role in the context of client unlearning. When the unlearning client (C_u) possesses only a small portion (e.g., 10%) of the label 9 data, excluding this client directly from the federated learning (FL) training process would have a negligible impact on the model's performance for label 9. This implies that the absence of a small portion of data belonging to a specific class does not significantly affect the model's ability to generalize to that class. This observation is supported by a comparison between the retraining baseline and the SGA-based unlearning approach, where the retraining model (M_r) achieves 94.36% accuracy on label 9 data, while the unlearning model (M_u) only achieves 61.53% accuracy. Therefore, employing the simple SGA-based unlearning approach is ineffective in scenarios involving normal non-IID data distributions for client unlearning. This is evident from the observed results in Fig. 5.4, which clearly demonstrate that the approach excessively corrupts the model's generalization boundary, resulting in a rapid reduction in accuracy to 0% within just two steps.

Previous memory protection with continual learning. The preceding analysis highlights the need for a protection strategy to safeguard the model's generalization boundary during the SGA-based unlearning process. Drawing inspiration from studies on continual learning [46, 2], which have identified catastrophic forgetting as a consequence of training models on sequentially different datasets, we can draw parallels between SGA-based unlearning and catastrophic forgetting. In both cases, when a model is trained on new data, it adjusts the parameters learned from previous data to accommodate the new data, resulting in the loss of knowledge acquired from the old data. To address this issue, we can adopt a typical approach from continual learning known as Elastic Weight Consolidation (EWC). The fundamental concept behind EWC is to restrict the magnitude of updates to different parameters through the inclusion of regularization terms. By doing so, parameters that were more important for the previous data are changed minimally during training on new data. In other words, if certain parameters were deemed significant for previous tasks, they should undergo minimal changes during the learning process of new tasks.

Combined EWC-SGA-based unlearning Based on the aforementioned insights, we propose an EWC-SGA-based unlearning framework. The framework consists of the following steps:

1) Calculation of Importance Factors: Firstly, the importance factor of each param-

eter in the model is computed using the Fisher Information matrix. This factor is determined by the squared difference between the local parameter θ_i and the corresponding global parameter $\theta_{global,i}$. Mathematically, it can be expressed as:

$$F_i(\theta_i - \theta_{global,i})^2 \tag{5.1}$$

2) Incorporation of Importance Factors: The importance factors are then integrated into the traditional cross-entropy loss as a regularization term. This additional term restricts the magnitude of parameter updates during the unlearning process. Parameters with higher importance factors are more resistant to updates. The resulting loss, termed the "unlearning loss," is given by:

$$L_u(\theta) = L_{ce}(\theta) + \frac{\lambda}{2} \sum_i F_i(\theta_i - \theta_{global,i})^2 \tag{5.2}$$

, where λ represents the strength of the regularization and $L_{ce}(\cdot)$ denotes the cross-entropy loss.

3) SGA-Based Unlearning: Finally, the SGA-based unlearning algorithm is applied using the unlearning loss. This approach effectively controls the model's generalization boundary and safeguards the remaining clients from being adversely affected during the unlearning process.

5.4.4 Methodology for Sample Unlearning

Sample-level federated unlearning aims to remove the knowledge acquired from a specific portion of client data from the global model, while client unlearning involves eliminating knowledge from the entire client data. In this context, the EWC-SGA-based unlearning framework remains effective for sample unlearning. Furthermore, federated learning inherently ensures data privacy, resulting in the unlearning data ($\boldsymbol{x}^*$) for both client-level and sample-level residing solely on the local client-side. Consequently, the unlearning process is executed exclusively on the client-side. To accomplish the objective, the current global model is initially downloaded to the designated unlearning client. Subsequently, the EWC-SGA-based framework is applied to derive the unlearning model (M_u). Ultimately, the unlearning model is uploaded to the server-side, replacing the previous global model, denoted as M_{global}^{new}.

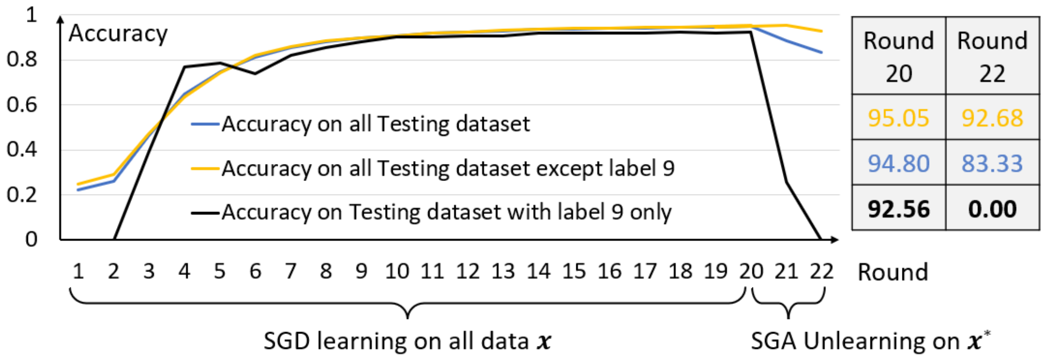
5.5 Performance Evaluation

In this section, a series of experiments is performed to validate the efficacy of the SGA-based unlearning framework and its integration with the Elastic Weight Consolidation (EWC) technique. To provide a comprehensive comparison, the baseline approach of *retraining* is employed, wherein the model is trained anew using only the remaining client data ($\boldsymbol{x}/\boldsymbol{x}^*$). This approach serves as an exact unlearning method as it entirely excludes the unlearning data ($\boldsymbol{x}^*$) from the training process. By including this baseline, we aim to highlight the advantages of our unlearning framework in terms of knowledge elimination.

5.5.1 Results on class unlearning

In order to demonstrate the efficacy of the SGA approach in controlling the model's generalization boundary, an experiment is conducted in the context of traditional FL using the FedAvg algorithm with the MNIST dataset. In this experiment, the label ”9” is designated as the unlearning class, and the data associated with this label in the server-side is considered the unlearning data ($\boldsymbol{x}^*$). These unlearning data can also be synthetic data generated using Generative Adversarial Networks (GANs). The training process is divided into a ”learning stage” consisting of 20 rounds and an ”unlearning stage” comprising 2 rounds.

As depicted in Fig. 5.4, the global model accuracy steadily increases during the normal training process and converges during the ”learning stage.” Subsequently,



Round 20	Round 22
95.05	92.68
94.80	83.33
92.56	**0.00**

Figure 5.4: Experiment of SGA-based class unlearning on MNIST dataset.

during the ”unlearning stage,” the model accuracy on the data with the label ”9” (represented by the black line) experiences a sharp decline, reaching 0% accuracy in the following 2 rounds. This indicates that the model successfully eliminates its classification capability for the label ”9” that was learned previously. Simultaneously, the accuracy of the model on the remaining labels (represented by the golden line) remains largely unaffected, with a slight decrease from 95.05% to 92.68%. This result illustrates that SGA-based unlearning can achieve the target of eliminating model knowledge and controlling model generalization boundary for specified classes, which is consistent with the demonstration in Fig. 5.3. The observed results validate that the SGA-based unlearning approach effectively achieves the objective of eliminating specific class knowledge from the model and controlling the model's generalization boundary, which align with the demonstrated outcomes in Fig. 5.3.

5.5.2 Results on client unlearning

To investigate client-level unlearning, a series of experiments is conducted on a total of 10 clients using pathological data with a non-IID distribution. Each client is randomly assigned three classes out of a total of 10 classes [64]. The 10-th client is designated as the unlearning client, and its local data corresponds to the unlearning data ($\boldsymbol{x}_{10} = \boldsymbol{x}^*$). Since achieving convergence in federated learning with pathological data and non-IID

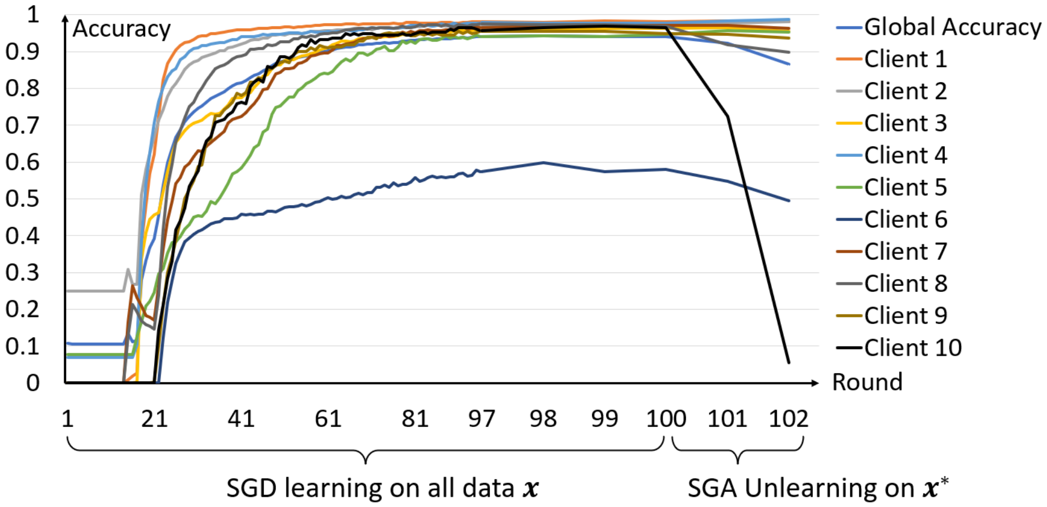


Figure 5.5: Experiment of EWC-SGA-based client unlearning on MNIST dataset.

distributions can be challenging, the training process in the "learning stage" is set to 100 rounds, followed by 2 rounds in the subsequent "unlearning stage." It is important to note that the client unlearning phase incorporates the EWC-SGA-based approach, rather than a simplified method. Furthermore, experiments are conducted using different datasets, and the results obtained from the MNIST dataset are presented in Fig. 5.5. During the "learning stage," the model performance for all clients exhibits a steady increase until convergence. However, in the subsequent "unlearning stage" utilizing the EWC-SGA-based approach, the model performance on the unlearning client (represented by the black line) experiences a significant decline from96.61% to 3.81%. Conversely, the model performance on the remaining clients (represented by the other colorful lines) remains relatively unaffected, with an average decrease of only 2.34%. To provide a more detailed visualization of the unlearning process, the image from the 97-th to the 102-nd round, corresponding to the "unlearning stage," is magnified. These results demonstrate the effectiveness of our methods in achieving the objective of eliminating the model's classification capability for the specified 10-th client while maintaining the model performance on the remaining clients.

5.5.3 Results on Sample unlearning

The experimental setup for sample unlearning is identical to that of class unlearning. However, in sample unlearning, the unlearning client is required to eliminate only a portion of its data contribution to the global model. This is achieved by introducing a hyperparameter called the "unlearning data proportion," denoted as $\alpha = \frac{\boldsymbol{x}^*}{\boldsymbol{x}_i} \in (0, 1]$. Here, $\alpha = 1$ corresponds to client unlearning, and in our experiments, we set $\alpha = 0.3$.

The results of the sample unlearning experiments align with the conclusions drawn from the client unlearning experiments. Our methods effectively eliminate the knowledge contributed by the unlearning data $\boldsymbol{x}^*$ from the global model. Moreover, since only a portion of data is unlearned from the unlearning client, and the remaining data $\boldsymbol{x}_i/\boldsymbol{x}^*$ still exist in the federated learning system, the model's performance on the unlearning client does not drop to almost 0% as observed previously. Instead, it retains some level of classification capability during subsequent training.

5.5.4 Efficiency and Convergence Analysis

As depicted in Figure 5.5, the efficiency and convergence of our SGA-based framework are analyzed. The unlearning process using our framework requires only 2 rounds, while the retraining baseline takes approximately 100 rounds to achieve convergence, which is comparable to the previous training process. This substantial difference, nearly 50 times faster, demonstrates the significant efficiency of our framework. Moreover, considering that the learning task in this study employs the simple MNIST dataset, the efficiency gap is expected to beeven wider for more challenging tasks. Additionally, the experiment results presented in both Figure 5.4 and Figure 5.5 illustrate that our method successfully achieves the unlearning objective within a short period of approximately 2 rounds. Furthermore, we conduct tests under various experimental conditions listed in Table. 5.1, and in all cases, the methods empirically converge.

Table 5.1: The experiments summary of different levels unlearning. **SGA** and **RE** is the *(EWC-)SGA-based unlearning framework* and the *retraining baseline*, respectively. **G-/U-/R-Acc** denote the accuracy of **G**lobal model, **U**nlearning client and **R**emaining clients average, respectively. **LS/US** denote **L**earning **S**tage and **U**nlearning **S**tage, respectively.

Level		Class Unlearning				Client Unlearning				Sample Unlearning			
Dataset		MNIST		CIFAR-10		MNIST		CIFAR-10		MNIST		CIFAR-10	
Stage		LS	US	LS	US	LS	US	LS	US	LS	US	LS	US
SGA	G-Acc	94.80	83.33	67.83	59.07	94.18	86.56	58.43	51.76	95.63	91.44	58.36	52.81
	U-Acc	92.56	0.00	65.54	0.00	93.37	5.01	57.29	4.38	94.71	63.54	58.21	39.67
	R-Acc	95.05	92.68	68.77	64.14	96.61	90.45	59.69	54.27	95.87	92.17	59.57	56.79
RE	G-Acc	94.80	87.64	67.83	61.06	94.18	88.75	58.43	53.06	95.63	91.03	58.36	53.69
	U-Acc	92.56	0.00	65.54	0.00	93.37	8.65	57.29	7.53	94.71	65.27	58.21	41.05
	R-Acc	95.05	92.84	68.77	63.31	96.61	91.87	59.69	55.73	95.87	93.16	59.57	57.24

5.5.5 Table Summary of All Experiments

The effectiveness of the SGA-based unlearning framework in different stages of the dynamic model performance change is demonstrated in Fig. 5.4 and 5.5. To comprehensively evaluate our approach, extensive experiments are conducted on various datasets, comparing the results with the retraining baseline. A summary of all experiment results is presented in Table. 5.1. For instance, let's consider the experiment conducted on the MNIST dataset with client unlearning. Firstly, the typical FL algorithm is employed to train a well-trained global model, which represents the learning stage (LS) in our study. The model's performance is evaluated by measuring the global accuracy (G-Acc) and the local accuracy of all clients. Specifically, the local accuracy of the targeted unlearning client is denoted as U-Acc in the table, while the average local accuracy of the remaining clients (excluding the unlearning client) is represented as R-Acc. Secondly, the unlearning algorithm is applied to the well-trained global model, representing the unlearning stage (US). The resulting global accuracy, unlearning client accuracy, and remaining client accuracy after the unlearning stage are denoted as G-Acc, U-Acc, and R-Acc, respectively. The SGA and RE notations indicate different unlearning approaches, with RE representing the retraining baseline.

5.6 Remarks

This charpter delves into the concept of machine unlearning within the context of federated learning, elucidating its architecture, challenges, and objectives. We explore the impact of training data on the performance of federated learning models and propose a comprehensive framework to address unlearning requests across multiple levels—class, client, and sample. Through extensive experiments conducted on diverse datasets, we validate the efficacy of our approach.

Furthermore, we identify several intriguing open issues in this domain:

- Knowledge Identification: In federated learning, knowledge accumulation occurs implicitly through the model aggregation process. Effectively discerning and attributing knowledge contributions from different clients remains a challenge.
- Feature-Level Unlearning: In vertical federated learning, clients possess distinct features of the same data sample. Extending the unlearning process to the feature level poses a significant research question.
- Evaluation Metrics: Presently, the assessment of unlearning effectiveness primarily relies on indirect comparisons with retraining. Designing comprehensive evaluation criteria to provide a clearer assessment of the unlearning effect is imperative.

These open issues serve as promising directions for further research and development in the realm of federated unlearning.

5.7 Discussion

In this chapter, we design a general federated unlearning framework based on SGA and achieve good unlearning performance. However, our work is in the very early beginnings of the current federated unlearning field. We expect there will be more promising follow-up works in the future. Besides, we also find that the unlearning metric is an open challenge in this field, i.e., how to evaluate whether the unlearning algorithm truly remove the specific target knowledge while keeping the knowledge of remaining data. We believe that this issue will become a hot research point in the near future.

Chapter 6

Conclusions and Future Work

6.1 Conclusion

This thesis explores the complex cooperation and competition relationships that coexist among heterogeneous clients in the FL system and then utilize these relationships to design effective mechanisms to improve the performance of FL in different aspects. First, we consider the competition relationship among heterogeneous clients in the FL system, where the server only has limited rewards to motivate client participation. To achieve the long-term sustainability of FL, we extend the crucial economic properties of mechanism to a long-term form to fit the successive FL process. Then, we present a long-term online VCG auction mechanism for FL that employs an experience-driven deep reinforcement learning algorithm to directly obtain the long-term optimal strategy. Second, we rethink the nature of client collaboration in FL, where the local model aggregation process is essentially a coalition game among clients. Thus, we introduce the "Shapley Value" concept from the coalition game theory, which can accurately identify the domain relevance among heterogeneous clients. The domain relevance identification promote each client to form their respective personalized coalitions, while the Shapley Value can further serve as personalized aggregation weights to achieve model customization for each client. Third, we take the first step to derive the "Machine Unlearning" concept to the FL context, which is referred to as "Federated Unlearning" (FU), and define different types of unlearning requests in FU. Then, we propose a general pipeline to enable model knowledge unlearning with the client collaboration in FL, where we revisit the nature of how the training data affects the FL model during the learning process and thereby empower the proposed pipeline with the reverse stochastic gradient ascent (SGA).

6.2 Future Work

This thesis is an initial exploration of the cooperation and competition relationships that coexist among heterogeneous clients in the current FL systems. Our work demonstrates that these complex relationships are a double-edged sword. While they pose various challenges for FL systems, they can also be well utilized to design more effective mechanisms to improve the performance of FL in various aspects. In the future, we can further develop our research from the following potential perspectives. First, we will continue to investigate and design in-depth incentive mechanisms that further take into account the dynamic nature of clients in FL systems. Besides, the current designs rarely consider the local data value of client as a factor, which is actually very important in FL incentive mechanism designs. How to accurately define the data value for each client regarding the FL global learning task is a potential research direction. Second, we will investigate more general and efficient personalized FL algorithms. While the current pFedSV shows significant performance improvement, it raises challenges on the computation and communication overhead from the SV estimation. Moreover, how to generate personalized models for those "unseen" clients that newly participate in the FL system is also a critical future direction. Third, we will keep designing more efficient federated unlearning algorithms and taking good utilization of client collaboration. Since the current SGA-based federated unlearning algorithm still requires some computation to generate reverse gradient for knowledge unlearning, a potential research direction is to adapt the data distillation techniques that can reproduce the gradient with a tiny amount of distilled data. Furthermore, the current learning and unlearning techniques in FL are independently developed, how to design a general model knowledge editing technique that can simultaneously achieve learning and unlearning is a potential research direction.

Chapter 7

List of Publications

* indicates equal contribution (co-first authors)

7.1 Published

1. **Leijie Wu**, Yaohong Ding, Akash Dhasade, Martijn De Vos, Anne-Marie Kermarrec, Song Guo. "*QuickDrop: Efficient Federated Unlearning via Synthetic Data Generation*". The 25th ACM/IFIP International Middleware Conference, 2024.

2. **Leijie Wu**, Song Guo, Yaohong Ding, Yufeng Zhan, and Jie Zhang. "*Rethinking Personalized User Collaboration when Facing An Agnostic Federated Learning System*". IEEE Transactions on Mobile Computing (TMC) (CCF-A), 2024.

3. **Leijie Wu**, Song Guo, Yi Liu, Zicong Hong, Yufeng Zhan, and Wenchao Xu. "*Long-term Adaptive VCG Auction Mechanism for Sustainable Federated Learning with Periodical Client Shifting*". IEEE Transactions on Mobile Computing (TMC) (CCF-A), 2023.

4. **Leijie Wu**, Song Guo, Yi Liu, Zicong Hong, Yufen Zhan, and Wenchao Xu.

"*Sustainable Federated Learning with Long-term Online VCG Auction Mechanism*". IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)(CCF-B), 2022.

5. **Leijie Wu**, Song Guo, Junxiao Wang, Zicong Hong, Jie Zhang, and Yaohong Ding. "*Federated Unlearning: Guarantee the Right of Clients to Forget*". IEEE Network (JCR-Q1), 2022.

6. Yi Liu*, **Leijie Wu***, Yufeng Zhan, Song Guo, and Zicong Hong (*Co-first author). "*Incentive-Driven Long-term Optimization for Edge Learning by Hierarchical Reinforcement Mechanism*". IEEE 41st International Conference on Distributed Computing Systems (ICDCS) (CCF-B), 2021.

7. Yi Liu, Song Guo, Yufeng Zhan, **Leijie Wu**, Zicong Hong, and Qihua Zhou. "*Chiron: A Robustness-aware Incentive Scheme for Edge Learning via Hierarchical Reinforcement Learning*". IEEE Transactions on Mobile Computing (TMC) (CCF-A), 2023.

8. Yufeng Zhan, Peng Li, **Leijie Wu**, and Song Guo. "*L4L: Experience-driven computational resource control in federated learning*". IEEE Transactions on Computers (TC)(CCF-A), 2021.

9. Yufeng Zhan, Jie Zhang, Zicong Hong, **Leijie Wu**, Peng Li, and Song Guo, "*A Survey of Incentive Mechanism Design for Federated Learning*". IEEE Transactions on Emerging Topics in Computing (TETC) (JCR Q1), 2021.

7.2 Under Review

1. **Leijie Wu**, Song Guo, Yaohong Ding, Junxiao Wang, Wenchao Xu, Jie Zhang, and Richard Yida Xu. "*Demystify Self-Attention in Vision Transformers from a Semantic Perspective: Analysis and Application*". submitted and under review.

2. **Leijie Wu**, Song Guo, Yaohong Ding, and Junxiao Wang. "*Waiting for Opportunity: Online Lazy Machine Unlearning by Uncertainty-based OOD Detection*". submitted and under review.

3. **Leijie Wu**, Song Guo, Junxiao Wang, Zicong Hong, Jie Zhang, and Jingren Zhou. "*On Knowledge Editing in Federated Learning: Perspectives, Challenges, and Future Directions*". submitted and under review.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[2] Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[3] Geir B Asheim, Tapan Mitra, and Bertil Tungodden. Sustainable recursive social welfare functions. In *The Economics of the Global Environment*, pages 165–190. Springer, 2016.

[4] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1):151–175, 2010.

[5] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysaw Dbiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[6] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of International Conference on Computational Statistics (COMPSTAT)*, 2010.

[7] Jean-Philippe Bouchaud, J Doyne Farmer, and Fabrizio Lillo. How markets slowly digest changes in supply and demand. In *Handbook of financial markets: dynamics and evolution*, pages 57–160. Elsevier, 2009.

[8] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *Proceedings of IEEE Symposium on Security and Privacy (SP)*, 2021.

[9] Eric Breck, Neoklis Polyzotis, Sudip Roy, Steven Whang, and Martin Zinkevich. Data validation for machine learning. In *MLSys*, 2019.

[10] Theodora S Brisimi, Ruidi Chen, Theofanie Mela, Alex Olshevsky, Ioannis Ch Paschalidis, and Wei Shi. Federated learning of predictive models from federated electronic health records. *International Journal of Medical Informatics*, 112:59–67, 2018.

[11] Jonathan Brophy and Daniel Lowd. Machine unlearning for random forests. In *International Conference on Machine Learning*, pages 1092–1104. PMLR, 2021.

[12] Thomas D Burd and Robert W Brodersen. Processor design for portable systems. *Journal of VLSI signal processing systems for signal, image and video technology*, 13(2-3):203–221, 1996.

[13] California State Legislature. California consumer privacy act of 2018. California Legislative Information, 2018. Available online: `https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=201720180AB375`.

[14] Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE symposium on security and privacy*, pages 463–480. IEEE, 2015.

[15] Javier Castro, Daniel Gómez, and Juan Tejada. Polynomial calculation of the shapley value based on sampling. *Computers & Operations Research*, 36(5):1726–1730, 2009.

[16] S. Chen, L. Jiao, L. Wang, and F. Liu. An online market mechanism for edge emergency demand response via cloudlet control. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, pages 2566–2574, 2019.

[17] Vikram S Chundawat, Ayush K Tarun, Murari Mandal, and Mohan Kankanhalli. Zero-shot machine unlearning. *IEEE Transactions on Information Forensics and Security*, 2023.

[18] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. *arXiv preprint arXiv:2102.07078*, 2021.

[19] Kate Donahue and Jon Kleinberg. Model-sharing games: Analyzing federated learning under voluntary participation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 5303–5311, 2021.

[20] Min Du, Zhi Chen, Chang Liu, Rajvardhan Oak, and Dawn Song. Lifelong anomaly detection through unlearning. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 1283–1297, 2019.

[21] Paul Dütting, Zhe Feng, Harikrishna Narasimhan, David Parkes, and Sai Srivatsa Ravindranath. Optimal auctions through deep learning. In *International Conference on Machine Learning*, pages 1706–1715. PMLR, 2019.

[22] European Union. Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation). Official Journal of the European Union, 2018. OJ L 119, 4.5.2016, p. 1–88.

[23] Zhe Feng, Harikrishna Narasimhan, and David C. Parkes. Deep learning for revenue-optimal auctions with budgets. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, page 354–362, Richland, SC, 2018. International Foundation for Autonomous Agents and Multiagent Systems.

[24] Marshall L Fisher, Janice H Hammond, Walter R Obermeyer, and Ananth Raman. Making supply meet demand in an uncertain world. *Harvard business review*, 72:83–83, 1994.

[25] Drew Fudenberg and Jean Tirole. *Game theory.* MIT press, 1991.

[26] Xiangshan Gao, Xingjun Ma, Jingyi Wang, Youcheng Sun, Bo Li, Shouling Ji, Peng Cheng, and Jiming Chen. Verifi: Towards verifiable federated unlearning. *arXiv preprint arXiv:2205.12709*, 2022.

[27] Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning*, pages 2242–2251. PMLR, 2019.

[28] Antonio Ginart, Melody Guan, Gregory Valiant, and James Y Zou. Making ai forget you: Data deletion in machine learning. In *Proceedings of Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[29] Jack Goetz and Ambuj Tewari. Federated learning via synthetic data. *arXiv preprint arXiv:2008.04489*, 2020.

[30] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[31] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Forgetting outside the box: Scrubbing deep networks of information accessible from input-output observations. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, pages 383–398. Springer, 2020.

[32] Noah Golowich, Harikrishna Narasimhan, and David C. Parkes. Deep learning for multi-facility location mechanism design. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 261–267. International Joint Conferences on Artificial Intelligence Organization, 7 2018.

[33] Laura Graves, Vineel Nagisetty, and Vijay Ganesh. Amnesiac machine learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11516–11524, 2021.

[34] Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*, 2019.

[35] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

[36] Werner Hediger. Sustainable development and social welfare. *Ecological economics*, 32(3):481–492, 2000.

[37] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[38] Junxian Huang, Feng Qian, Yihua Guo, Yuanyuan Zhou, Qiang Xu, Z Morley Mao, Subhabrata Sen, and Oliver Spatscheck. An in-depth study of lte: Effect of network protocol and application behavior on performance. *ACM SIGCOMM Computer Communication Review*, 43(4):363–374, 2013.

[39] Yutao Huang, Lingyang Chu, Zirui Zhou, Lanjun Wang, Jiangchuan Liu, Jian Pei, and Yong Zhang. Personalized cross-silo federated learning on non-iid data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7865–7873, 2021.

[40] Yihan Jiang, Jakub Konečnỳ, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019.

[41] Yutao Jiao, Ping Wang, Dusit Niyato, Bin Lin, and Dong In Kim. Toward an automated auction framework for wireless federated learning services market. *IEEE Transactions on Mobile Computing*, 2020.

[42] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[43] Jiawen Kang, Zehui Xiong, Dusit Niyato, Shengli Xie, and Junshan Zhang. Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory. *IEEE Internet of Things Journal*, 6(6):10700–10714, 2019.

[44] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR, 2020.

[45] Latif U Khan, Shashi Raj Pandey, Nguyen H Tran, Walid Saad, Zhu Han, Minh NH Nguyen, and Choong Seon Hong. Federated learning for edge networks: Resource optimization and incentive mechanism. *IEEE Communications Magazine*, 58(10):88–93, 2020.

[46] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[47] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[48] Vijay Krishna. *Auction Theory.* Academic Press, second edition, 2010.

[49] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[50] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[51] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020.

[52] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*, 2019.

[53] Paul Pu Liang, Terrance Liu, Liu Ziyin, Nicholas B Allen, Randy P Auerbach, David Brent, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523*, 2020.

[54] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[55] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3):2031–2063, 2020.

[56] Gaoyang Liu, Xiaoqiang Ma, Yang Yang, Chen Wang, and Jiangchuan Liu. Federaser: Enabling efficient client-level data removal from federated learning models. In *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, pages 1–10. IEEE, 2021.

[57] Mi Luo, Fei Chen, Dapeng Hu, Yifan Zhang, Jian Liang, and Jiashi Feng. No fear of heterogeneity: Classifier calibration for federated learning with non-iid data. *arXiv preprint arXiv:2106.05001*, 2021.

[58] Nguyen Cong Luong, Zehui Xiong, Ping Wang, and Dusit Niyato. Optimal auction for edge computing resource management in mobile blockchain networks: A deep learning approach. In *2018 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2018.

[59] Pavel Mach and Zdenek Becvar. Mobile edge computing: A survey on architecture and computation offloading. *IEEE communications surveys & tutorials*, 19(3):1628–1656, 2017.

[60] Sasan Maleki, Long Tran-Thanh, Greg Hines, Talal Rahwan, and Alex Rogers. Bounding the estimation error of sampling-based shapley value approximation. *arXiv preprint arXiv:1306.4265*, 2013.

[61] Irwin Mann and Lloyd S Shapley. Values of large games. 6: Evaluating the electoral college exactly. Technical report, RAND CORP SANTA MONICA CA, 1962.

[62] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

[63] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.

[64] H Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. Federated learning of deep networks using model averaging. *arXiv preprint arXiv:1602.05629*, 2016.

[65] James Midgley. *Social development: The developmental perspective in social welfare*. Sage, 1995.

[66] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

[67] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[68] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[69] Roger B Myerson. *Game theory*. Harvard university press, 2013.

[70] Lokesh Nagalapatti and Ramasuri Narayanam. Game of gradients: Mitigating irrelevant clients in federated learning. *arXiv preprint arXiv:2110.12257*, 2021.

[71] Quoc Phong Nguyen, Bryan Kian Hsiang Low, and Patrick Jaillet. Variational bayesian unlearning. *Advances in Neural Information Processing Systems*, 33:16025–16036, 2020.

[72] Solmaz Niknam, Harpreet S Dhillon, and Jeffrey H Reed. Federated learning for wireless communications: Motivation, opportunities, and challenges. *IEEE Communications Magazine*, 58(6):46–51, 2020.

[73] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[74] Shashi Raj Pandey, Nguyen H Tran, Mehdi Bennis, Yan Kyaw Tun, Aunas Manzoor, and Choong Seon Hong. A crowdsourcing framework for on-device federated learning. *IEEE Transactions on Wireless Communications*, 2020, in Press.

[75] Nicolò Romandini, Alessio Mora, Carlo Mazzocca, Rebecca Montanari, and Paolo Bellavista. Federated unlearning: A survey on methods, design guidelines, and evaluation metrics. *arXiv preprint arXiv:2401.05146*, 2024.

[76] Gavin A Rummery and Mahesan Niranjan. *On-line Q-learning using Connectionist Systems*, volume 37. University of Cambridge, Department of Engineering Cambridge, UK, 1994.

[77] Omer Sagi and Lior Rokach. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1249, 2018.

[78] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

[79] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[80] Aviv Shamsian, Aviv Navon, Ethan Fetaya, and Gal Chechik. Personalized federated learning using hypernetworks. *arXiv preprint arXiv:2103.04628*, 2021.

[81] Weijie Shi, Linquan Zhang, Chuan Wu, Zongpeng Li, and Francis C.M. Lau. An online auction framework for dynamic resource provisioning in cloud computing. *SIGMETRICS Perform. Eval. Rev.*, 42(1):71–83, June 2014.

[82] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge computing: Vision and challenges. *IEEE internet of things journal*, 3(5):637–646, 2016.

[83] Tianshu Song, Yongxin Tong, and Shuyue Wei. Profit allocation for federated learning. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 2577–2586. IEEE, 2019.

[84] Qiheng Sun, Xiang Li, Jiayao Zhang, Li Xiong, Weiran Liu, Jinfei Liu, Zhan Qin, and Kui Ren. Shapleyfl: Robust federated learning based on shapley value. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2096–2108, 2023.

[85] Canh T Dinh, Nguyen Tran, and Tuan Dung Nguyen. Personalized federated learning with moreau envelopes. *Advances in Neural Information Processing Systems*, 33, 2020.

[86] Siyi Tang, Amirata Ghorbani, Rikiya Yamashita, Sameer Rehman, Jared A Dunnmon, James Zou, and Daniel L Rubin. Data valuation for medical imaging using shapley value and application to a large-scale chest x-ray dataset. *Scientific reports*, 11(1):1–9, 2021.

[87] Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(7), 2009.

[88] Huangshi Tian, Minchen Yu, and Wei Wang. Continuum: A platform for cost-aware, low-latency continual learning. In *Proceedings of the ACM Symposium on Cloud Computing*, pages 26–40, 2018.

[89] Praneeth Vepakomma, Tristan Swedish, Ramesh Raskar, Otkrist Gupta, and Abhimanyu Dubey. No peek: A survey of private distributed deep learning. *arXiv preprint arXiv:1812.03288*, 2018.

[90] William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16(1):8–37, 1961.

[91] Guan Wang. Interpret federated learning with shapley values. *arXiv preprint arXiv:1905.04519*, 2019.

[92] Guan Wang, Charlie Xiaoqian Dang, and Ziye Zhou. Measure contribution of participants in federated learning. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 2597–2604. IEEE, 2019.

[93] Hao Wang, Zakhary Kaplan, Di Niu, and Baochun Li. Optimizing federated learning on non-iid data with reinforcement learning. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 1698–1707. IEEE, 2020.

[94] Junxiao Wang, Song Guo, Xin Xie, and Heng Qi. Federated unlearning via class-discriminative pruning. In *Proceedings of the ACM Web Conference 2022*, pages 622–632, 2022.

[95] Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Françoise Beaufays, and Daniel Ramage. Federated evaluation of on-device personalization. *arXiv preprint arXiv:1910.10252*, 2019.

[96] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3-4):279–292, 1992.

[97] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.

[98] Jiasi Weng, Jian Weng, Jilian Zhang, Ming Li, Yue Zhang, and Weiqi Luo. Deepchain: Auditable and privacy-preserving deep learning with blockchain-based incentive. *IEEE Transactions on Dependable and Secure Computing*, 2019.

[99] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[100] Leijie Wu, Song Guo, Junxiao Wang, Zicong Hong, Jie Zhang, and Yaohong Ding. Federated unlearning: Guarantee the right of clients to forget. *IEEE Network*, 36(5):129–135, 2022.

[101] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[102] Jaehong Yoon, Wonyong Jeong, Giwoong Lee, Eunho Yang, and Sung Ju Hwang. Federated continual learning with weighted inter-client transfer. In *Proceedings of International Conference on Machine Learning (ICML)*, 2021.

[103] W. You, L. Jiao, J. Li, and R. Zhou. Scheduling ddos cloud scrubbing in isp networks via randomized online auctions. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pages 1658–1667, 2020.

[104] Han Yu, Zelei Liu, Yang Liu, Tianjian Chen, Mingshu Cong, Xi Weng, Dusit Niyato, and Qiang Yang. A fairness-aware incentive scheme for federated learning. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 393–399, 2020.

[105] Yulan Yuan, Lei Jiao, Konglin Zhu, and Lin Zhang. Incentivizing federated learning under long-term energy constraint via online randomized auctions. *IEEE Transactions on Wireless Communications*, 2021.

[106] Rongfei Zeng, Shixun Zhang, Jiaqi Wang, and Xiaowen Chu. Fmore: An incentive scheme of multi-dimensional auction for federated learning in mec. In *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, 2020.

[107] Yufeng Zhan, Peng Li, Zhihao Qu, Deze Zeng, and Song Guo. A learning-based incentive mechanism for federated learning. *IEEE Internet of Things Journal*, 2020.

[108] Yufeng Zhan and Jiang Zhang. An incentive mechanism for efficient edge learning by deep reinforcement learning approach. In *Proc. of IEEE INFOCOM*, pages 1–10, 2020.

[109] Jie Zhang, Zhihao Qu, Chenxi Chen, Haozhao Wang, Yufeng Zhan, Baoliu Ye, and Song Guo. Edge learning: The enabling technology for distributed big data analytics in the edge. *ACM Computing Surveys (CSUR)*, 54(7):1–36, 2021.

[110] Michael Zhang, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M Alvarez. Personalized federated learning with first order model optimization. In *International Conference on Learning Representations*, 2020.

[111] Xinwei Zhang, Mingyi Hong, Sairaj Dhople, Wotao Yin, and Yang Liu. Fedpd: A federated learning framework with optimal rates and adaptivity to non-iid data. *arXiv preprint arXiv:2005.11418*, 2020.

[112] Yuheng Zhang, Ruoxi Jia, Hengzhi Pei, Wenxiao Wang, Bo Li, and Dawn Song. The secret revealer: Generative model-inversion attacks against deep neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 253–261, 2020.

[113] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[114] Zhi Zhou, Xu Chen, En Li, Liekang Zeng, Ke Luo, and Junshan Zhang. Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proceedings of the IEEE*, 107(8):1738–1762, 2019.