Personalized User Models in a Real-world Edge Computing Environment: A Peer-to-peer Federated Learning Framework

Xiangchi Song*, Zhaoyan Wang, KyeongDeok Baek and In-Young Ko

Korea Advanced Institute of Science and Technology, Daejeon, Republic of Korea
E-mail: xcsong@kaist.ac.kr; zhaoyan123@kaist.ac.kr; kyeongdeok.baek@kaist.ac.kr; iko@kaist.ac.kr
**Corresponding Author*

Received 05 September 2024; Accepted 09 November 2024

Abstract

As the number of IoT devices and the volume of data increase, distributed computing systems have become the primary deployment solution for large-scale Internet of Things (IoT) environments. Federated learning (FL) is a collaborative machine learning framework that allows for model training using data from all participants while protecting their privacy. However, traditional FL suffers from low computational and communication efficiency in large-scale hierarchical cloud-edge collaborative IoT systems. Additionally, due to heterogeneity issues, not all IoT devices necessarily benefit from the global model of traditional FL, but instead require the maintenance of personalized levels in the global training process. Therefore we extend FL into a horizontal peer-to-peer (P2P) structure and introduce our P2PFL framework: efficient peer-to-peer federated learning for users (EPFLU). EPFLU transitions the paradigms from vertical FL to a horizontal P2P structure from the user perspective and incorporates personalized enhancement techniques using private information. Through horizontal consensus

Journal of Web Engineering, Vol. 23_8, 1057–1084.
doi: 10.13052/jwe1540-9589.2381

information aggregation and private information supplementation, EPFLU solves the weakness of traditional FL that dilutes the characteristics of individual client data and leads to model deviation. This structural transformation also significantly alleviates the original communication issues. Additionally, EPFLU has a customized simulation evaluation framework, and uses the EUA dataset containing real-world edge server distribution, making it more suitable for real-world large-scale IoT. Within this framework, we design two extreme data distribution scenarios and conduct detailed experiments of EPFLU and selected baselines on the MNIST and CIFAR-10 datasets. The results demonstrate that the robust and adaptive EPFLU framework can consistently converge to optimal performance even under challenging data distribution scenarios. Compared with the traditional FL and selected P2PFL methods, EPFLU achieves communication time improvements of 39% and 16% respectively.

Keywords: Peer-to-peer federated learning, personalized federated learning, hierarchical edge computing, edge-cloud environment.

1 Introduction

With the rapid advancement of Internet of Things (IoT) technology, an increasing number of devices are being connected to the Internet, leading to the generation of massive volumes of data. However, privacy concerns and data transmission limitations have made centralized cloud processing impractical. Federated learning (FL) [17], as a solution, allows participants to collaboratively train a shared model without sharing data directly.

Although traditional FL has had a revolutionary impact on the IoT, it encounters challenges in computational and communication efficiency within hierarchical collaborative systems. Lian et al.'s case study highlighted the central server as a bottleneck in distributed parallel systems with a large number of clients [14]. Therefore, in large-scale IoT scenarios involving a large number of devices, when these devices generate and simultaneously upload substantial amounts of data traffic, the quality of vertical links (essentially the upload phase) often becomes the primary source of resource consumption. Existing works attempt to reduce the communication overheads of FL by distributing computation [18] or reducing aggregation frequencies [15]. However, these methods are still based on traditional vertical aggregation, and thus the primary cost issues persist. Therefore, traditional FL is not suitable for current large-scale IoT and hierarchical computing environments.

Furthermore, the design of hierarchical collaboration systems not only aims to optimize the system's interaction and communication capabilities but also considers enhancing model performance and serviceability from the user perspective. Users are currently facing a diversity of tasks and data characteristics, so they have an urgent need for personalized models, requiring the system to train models according to personal needs to achieve the best service experience. Although the FL approach excels in privacy protection by aggregating all user data, it tends to promote a solution that is universally effective for all participants.In this method it is difficult to fully utilize the unique information of local data when dealing with highly heterogeneous data and tasks, thereby leading to insufficient satisfaction of personalized demands. The aggregated model may perform poorly for some clients because the comprehensive model actually dilutes the characteristics of individual models. This phenomenon is called the *weakening effect* [5]. Therefore, in order to avoid the weakening effect caused by heterogeneity, it is necessary for the system to introduce personalization technology, retain private information in the model, and enhance model usability from the user perspective.

Given these issues, researchers have begun to explore solutions more suited to the current environment. Peer-to-peer federated learning (P2PFL) emerges as a potential solution, significantly mitigating the traditional FL bottleneck of central server dependency and facilitating the direct sharing of resources and services [2]. As P2PFL promotes direct interactions among edge devices, this horizontal data exchange paradigm significantly alleviates the inevitable communication issues of traditional vertical FL designs. With the incorporation of personalization technologies, the P2PFL framework enables each client not only to participate in the training of the global model but also to adjust and optimize their model based on the characteristics of their local data. This framework not only improves computational and communication efficiency but also provides a flexible solution for the personalized needs of users in IoT environments. Because the P2PFL system is based on direct information exchange between clients, it lacks comprehensive oversight of global information. Consequently, it is challenging to address special data distributions in local clients that can affect system training. Developing personalized models that are both accurate and representative of the local datasets is particularly difficult under these conditions.

As an extension of [25], this study proposes a framework, efficient peer-to-peer federated learning for users (**EPFLU**), a P2PFL approach that considers performance enhancement from the user perspective, to transition

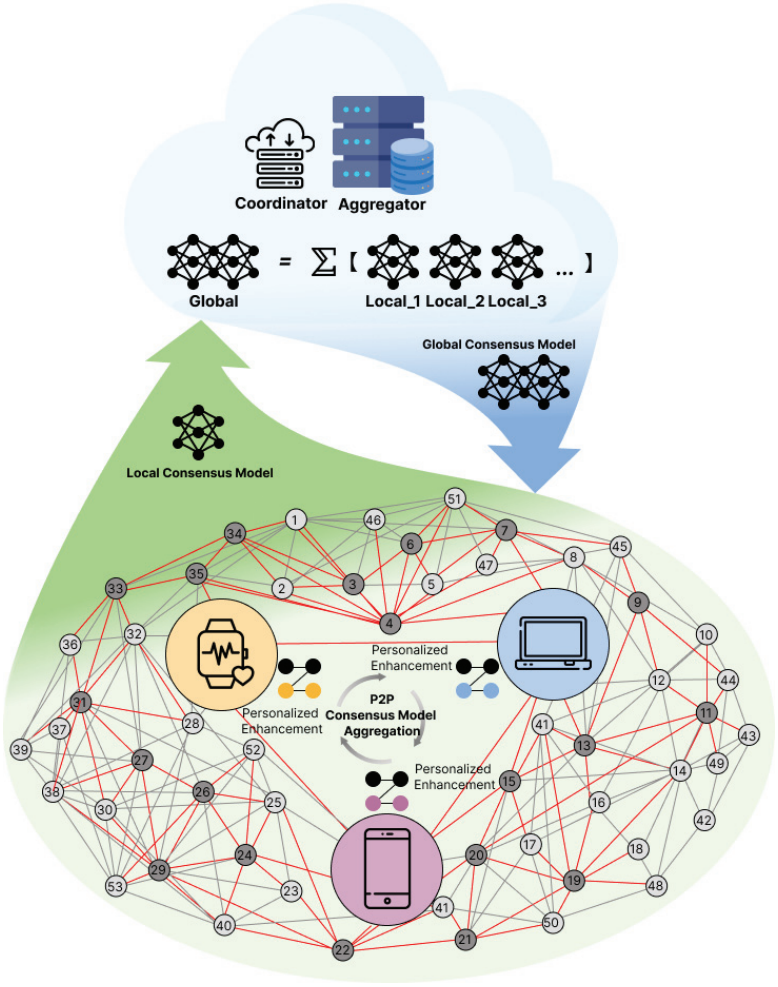


Figure 1 EPFLU: An efficient P2PFL framework from the user perspective.

the traditional vertical FL to a horizontal P2P structure. This approach, by supplementing private information after aggregating consensus models among clients within horizontal networks, offers a more efficient and personalized method for model training in large-scale IoT from the user perspective, as shown in Figure 1. An aggregator responsible for model aggregation on the cloud is maintained, providing foundational FL functions for P2PFL deployment. EPFLU establishes P2P connections among edge clients, forming aggregation paths based on a predefined topological strategy,

thus executing the horizontal transmission and aggregation of consensus models. Once the global consensus model updates the participating clients, clients can use private information to enhance the personalization of their models. We established a comprehensive evaluation simulation framework and used the EUA dataset, which is based on real-world data sources for edge server locations, to evaluate both the baselines and our proposed method. The main contributions of this paper are as follows:

- We propose EPFLU, a robust personalized adaptive method based on cloud-edge collaboration technology, which improves the personalized model from the user perspective and is more efficient in training and communication than traditional FL.
- We adopt a realistic heterogeneous data distribution compared to traditional data partitioning methods. Then we use the EUA dataset, which is based on real-world data sources for edge server distribution. These processes better simulate large-scale IoT scenarios in the real world, thereby demonstrating the applicability of EPFLU to real environmental conditions.
- We design a more complex and comprehensive simulation framework for EPFLU, which considers the communication consumption over actual distances in addition to model computation. The outstanding performance of EPFLU is demonstrated through detailed experiments from the user perspective.

The rest of the paper is organized as follows: Section 2 presents the related work of this paper, Section 3 introduces the problems and design goal, Section 4 describes the system design for the EPFLU-P2PFL framework, Section 5 states the experimental environment, including data preparation and experimental settings. Section 6 analyzes and discusses the experimental results, and Section 7 concludes this paper's work.

2 Related Work

2.1 Federated Learning and Hierarchical Distributed System

The concept of FL has been driving advancements in machine learning, networking, and communication since its introduction in 2017 [17]. In recent years, considerable progress has been made in various domains, including speech processing [31], multimodal learning [6], and AI-driven IoT [1]. FL has become popular in edge intelligence for its ability to enhance training effectiveness and privacy among users and clients. FL optimizes resource

and communication efficiency in cloud-edge systems, particularly in hierarchical federated architectures. Mills et al. utilized model compression techniques to further reduce communication resource usage in cloud-edge systems [18]. Liu et al. studied the joint communication and computation strategy for IoT devices in hierarchical edge computing systems using game theory, aiming to enhance system interaction efficiency [16]. Liu et al. also proposed a hierarchical FL architecture supported by collaborative training algorithms [15]. However since these optimizations are still completed within a hierarchical structure, communication issues remain unavoidable. EPFLU significantly alleviates the overhead associated with vertical communication by transitioning traditional structures to P2P transmission.

2.2 Personalized Federated Learning in Edge Computing

In the domain of intelligent IoT applications, personalized federated learning has become a focal point of research. In this context, various techniques have emerged, including federated transfer learning [4], federated multi-task learning [24], and federated model distillation [13]. These techniques provide abundant choices for implementing personalized federated learning in IoT environments. Wu et al. introduced numerous technologies and provided a detailed analysis of achieving personalized federated learning goals in IoT cloud-edge environments [30]. Wang et al. addressed model heterogeneity by introducing strategies involving common base layers and jointly personalized layers to enhance model accuracy and speed up training [28]. Mills et al. utilized trainable parameters in private layers to accelerate model convergence and achieve satisfactory personalized results [19]. Li et al. achieved edge device personalization by enabling clients to learn personalized models instead of sharing models, as in classical federated learning [12].

2.3 Distributed P2PFL Architecture

In decentralized environments, traditional FL encounters limitations such as constrained communication and the unavailability of a fully connected central server, particularly in P2P networks. Chen et al. introduced a decentralized global model training protocol [2] and subsequently addressed issues such as high bandwidth, data privacy, and single point of failure by proposing the first fine-grained global model training protocol for FL in P2P networks [3]. Sharma et al. presented a compromise solution for extreme FL privacy infringement and maintaining system integrity and privacy in a completely

decentralized topology [23]. Roy et al. proposed a concept that enables direct interaction among participants in a highly dynamic P2P environment without relying on traditional servers [22]. Most of the previous works focused on critical issues such as privacy and fault tolerance in P2PFL systems. However, model performance is significantly impacted when dealing with heterogeneous data. Similar to prior research, our EPFLU framework ensures stability in the system's transmission and training processes, enhancing convergence efficiency and reducing communication costs. Additionally, EPFLU enhances the personalization level of user models based on different users' private data, thereby utilizing local data more efficiently from the user perspective.

3 Problem and Design Goal

This paper primarily focuses on the challenges of communication and model personalization within large-scale IoT in real-life scenarios. It emphasizes the growing need to reduce system communication latency and costs as the deployment of IoT devices and data accumulation continue to increase. Simultaneously, the increase in the number of users has led to a growing demand for personalized solutions. Under the premise of privacy protection, we identify three core issues: (1) Vertical FL contradiction has efficient data utilization but leads to inevitable communication delays and losses; (2) horizontal P2P structure contradiction enables direct data and knowledge exchange among devices to reduce system communication pressure but lacks effective data utilization from all global participants; (3) personalization deficiency allows the heterogeneous data to be held by participants, but is insufficient to meet their needs for personalized solutions.

Based on these issues, we consider how to build an efficient and secure aggregation system in large-scale IoT from the perspective of user benefits. The traditional FL-based cloud-edge hierarchical structure distributes data and model training tasks across edge devices and aggregates them in the cloud. We aim to extend the original FL to a horizontal P2P structure to reduce the communication cost of EPFLU, and further improve the personalization level through model personalization technology. We consider these to design EPFLU:

(1) **Taking advantage of FL and P2P.** When building EPFLU, it is necessary to devise a flexible and effective aggregation method, ensuring secure transmission and orderly aggregation of models and parameters across different levels, effectively leveraging the advantages of FL and P2P structure in both vertical and horizontal dimensions.

(2) **Personalized model enhancement.** The personalization techniques employed in EPFLU are designed to better meet users' needs for personalized solutions. These techniques ensure that the model can fully utilize data from all participants while providing personalized model enhancements for users in heterogeneous data environments.
(3) **Privacy protection.** User privacy is crucial; the aggregation method in EPFLU must ensure the privacy and security of user data. In particular, in the P2P transmission process, both parties in a state of complete trust will encounter challenges such as data integrity and privacy leakage during data transmission.

4 System Design for EPFLU – A P2PFL Framework

This section primarily outlines the work we have undertaken to implement EPFLU. We based our approach on the traditional FL framework and migrated it to a P2P structure. By separating private parameters and enhancing model personalization after aggregation, EPFLU achieves horizontal personalized aggregation and improves communication efficiency.

4.1 Cloud-side and Aggregation Topology Generation Strategy

Based on the vertical FL system, we use the global topology to connect all participating clients on the edge-side networks. EPFLU applies the model aggregation and distribution process in a horizontal topological network, connecting and computing models through predefined strategies, as shown in Figure 2. The topological information, EPFLU's strategies, sampling records, and distance matrix are saved in the coordinator on the cloud. The operation and execution process of the entire system are controlled by the cloud coordinator. Since the P2P structure is decentralized, the coordinator is only responsible for recording and global control according to policies we made ahead of time, and does not have any computing or aggregating functions. All calculations will be completed independently by the client model controlled by the user.

In EPFLU, the system will start running from Algorithm 1, aiming to build an adaptive and optimal aggregation path. As shown in the P2P section of the cloud partition in Figure 2, all information recording and system control during the P2P process are managed by the cloud coordinator, in addition to the system's aggregation and personalization technologies. Specifically, there are four parts: EPFLU will generate a *distance matrix* based on the

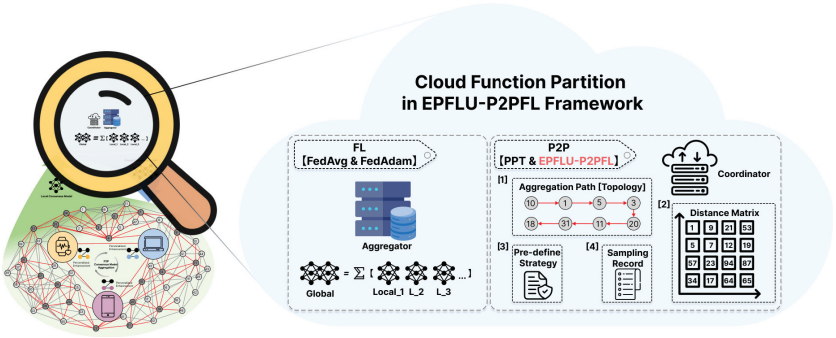


Figure 2 Functional structure of the cloud-side in the EPFLU-P2PFL framework.

dataset collected from real-world data sources, perform sampling, and generate the aggregation topology according to the *pre-define strategy*, store the results in the *aggregation path* to determine the subsequent aggregation order, and simultaneously create a *sampling record* to document the clients sampled for the current round.

To simulate client distribution, we generate a distribution matrix `generationMatrix` based on the distance matrix for all clients to represent their relationships, serving as a reference for subsequent transmission paths. A `startingClient` will be randomly selected from the center area within the client distribution matrix as the starting node of this round of aggregation (line 2). Then, it randomly samples a specified number of clients as `sampledClients` for the current aggregation process (line 3). This sampled number is within the pre-defined strategy. After selecting the clients, EPFLU establishes connections between the sampled clients. The connection principle, based on the `generationMatrix`, employs a traversal strategy that combines depth-first and breadth-first approaches, prioritizing connections with closer clients relative to the current client. Once all connections are established and the topology is generated, it is stored as the `aggregationPath` (line 4). It is worth noting that, after the aggregation path is generated, EPFLU will optimize the generated topology. If a newly sampled client appears in the direct connection between two sampled clients, the direct connection between the two original clients will be canceled, and it will be switched to a one-hop connection through the newly sampled client serving as an intermediate bridging client (line 5). Then the `sampledClients` and the optimized `aggregationPath` is returned (line 6). In this way, EPFLU can ensure a comprehensive and adaptive strategy for client selection and aggregation path.

Algorithm 1: Random client sampling and topology generation strategy.

Input : totalClients, sampleSize, generationMatrix
Output: sampledClients, aggregationPath
Procedure *Sample Clients and Generate the Aggregation Path*
  startingClient ← `selectStartingClient` (generationMatrix)
  sampledClients ← `sampleClient` (totalClients, sampleSize)
  aggregationPath ← `genTopo`(startingClient, sampledClients, generationMatrix)
  aggregationPath ← `topoOpt`(sampledClients, generationMatrix)
  return sampledClients, aggregationPath
end

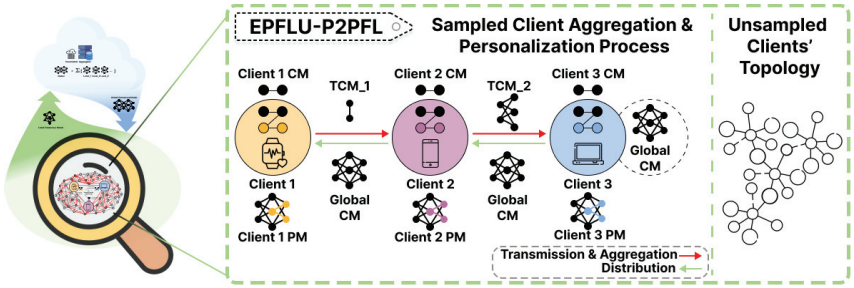


Figure 3 The edge-side structure of P2P transmission and aggregation with personalization processes in the EPFLU-P2PFL framework. CM: consensus model, TCM: temporary aggregated consensus model, and PM: personalized model.

4.2 Edge-side and Personalized Aggregation

Once the aggregation path is constructed, the next step is to implement the model aggregation and personalized processes within the topology. Regarding the part of model personalization enhancement, we have drawn inspiration from the approaches of [19] and [20], utilizing the batch normalization (BN) layer as private model parameters to store private information. EPFLU maintains the level of personalization of the model as it is aggregated by setting private statistics and using private trainable parameters `privateParameters`. Due to the difference in aggregation methods from the vertical FL, where clients upload in parallel and download after aggregation, the aggregation of the global model in the horizontal P2P structure occurs sequentially. Consequently, the integration method for the BN layer also differs. In EPFLU, clients that complete their download in advance during the aggregation and aggregated model deployment to global clients process

are enabled to execute the personalization enhancement earlier, as shown in the process of the sampled list in Figure 3. This is achieved by utilizing privatized information stored within the privacy model parameters, thereby the reducing waiting time from the perspective of users. We extend the separation and supplementary of privacy model parameters into the horizontal P2P propagation process and have devised a personalization strategy tailored to EPFLU.

Algorithm 2: Model aggregation and personalized enhancement based on P2P transmission.

```
   Input : sampledClients, aggregationPath, userModel
   Output: updatedUserModel
1  Procedure Local Training and Separate Private Info
2      for each client in sampledClients do
3          localTraining(userModel)
4          userModel', privateParameters ←
            separatePrivateParams(userModel)
5      end
6  end
7  Procedure Aggregation and Transmission
8      for each client in aggregationPath do
9          consensusModel ← Aggregate(userModel', aggregationPath)
10     end
11 end
12 Procedure Distribute Consensus Model
13     for each client in sampledClients do
14         userModel ← modelCoverage(consensusModel, aggregationPath)
15     end
16 end
17 Procedure User Model Enhancement
18     for each client in sampledClients do
19         updatedUserModel ← enhance(userModel, privateParameters)
20     end
21 end
22 return updatedUserModel
```

Algorithm 2 presents a procedure for personalized model implementation in EPFLU. The process in Figure 3 shows the detail of how the personalized information is integrated into the P2P horizontal aggregation structure. Starting with local training by `sampledClients`, the algorithm then separates `privateParameters` for local storage, preparing a local consensus model *Client CM* for aggregation (lines 1–6). The `sampledClients` and

the `aggregationPath` are used to aggregate and transfer the consensus models (lines 7–11), generating a global consensus model *Global CM* that includes only consensus information from `sampledClients`. The model cumulatively aggregated on the transmission path is a temporary aggregated consensus model *TCM* with partial client information. This *Global CM* is then deployed back to the `sampledClients` (lines 12–16). Finally, each client enhances this consensus model with its own private information stored by `privateParameters` (lines 17–21), resulting in a personalized model *Client PM*, and this enhanced *Client PM* will be used as the new local model for the next round of local training (line 22). The costs of local training, model transfer, and model aggregation processes are recorded and accumulated for subsequent use in our simulation evaluation system (see Section 5.2).

5 Experimental Environment

5.1 Real-world EUA Dataset Processing

We utilized a set of EUA datasets derived from real-world data sources [10], which includes the geographical locations of all cellular base stations in Australia. In our study, we used the server distribution data from the dataset and subsequently deployed the data and models at these server locations. We briefly illustrate the server distribution of the EUA dataset and annotate the aggregation methods of different algorithms in Figure 4.

To simulate a realistic edge computing distribution environment and control the sampling, we randomly selected a geographical range with a latitude range of -33.5 to -33 and a longitude range of 140 to 150. Within this range,

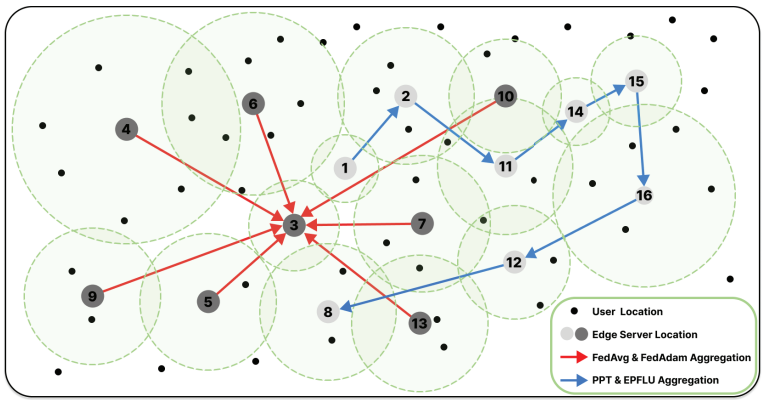


Figure 4 Example of EUA dataset edge server distribution and algorithms.

there are a total of 664 client nodes, slightly exceeding our target number of sampled clients of 500. It is a reasonable sampling range that prevents issues resulting from a region that is too large, which could cause excessive distances between some clients, and from a region that is too small, which could lead to redundant sampling. It is important to note that the EUA dataset itself does not distinguish between cloud and edge servers. In traditional FL methods, the cloud server's role is to be the aggregator, aggregating sub-models from clients. Therefore, we selected the server closest to the center of this range as the aggregator, as shown on the left side of the cloud partition in Figure 2. We assume this server does not participate in training but only in the aggregation process, acting as the cloud server for selected FL algorithms. For the sake of logical consistency, we will continue to use the terms cloud server and edge clients in subsequent discussions.

Through the dataset processing, we obtained the distance matrix based on real-world data sources for subsequent use, representing the actual distribution of cloud server and edge clients, as mentioned in Section 4. This distance matrix provides reliable foundational data for the performance test of EPFLU, enhancing the authenticity of our experiments.

5.2 Simulation Framework Suitable for an EPFLU-P2PFL System

To make EPFLU more realistic, we developed a communication model for detailed simulation of real-world scenarios, enabling accurate assessment of the communication overhead. Our simulation framework considers the specific conditions of wireless communication environments [27] and factors such as bandwidth, channel gain, transmitter power, and noise power present in existing hierarchical FL systems [15]. Additionally, we paid particular attention to the path loss of wireless signals in urban environments. Previous research mainly focused on computational delay and cost without considering the propagation and attenuation of wireless signals. In real-world scenarios, this is critically important because the communication relationships between the cloud and the edge, as well as between edge devices, are non-negligible. Therefore, we adopt the Hata model suitable for urban areas [8] and the free space path loss (FSPL) model [7] to estimate communication losses. In EPFLU, a simplified urban environment path loss model is defined by the following equation:

$$PL_d = 20 \cdot \log_{10} f + 20 \cdot \log_{10}(4\pi/c), \tag{1}$$

$$PL = PL_d + 10 \cdot n \cdot \log_{10}(d). \tag{2}$$

Here, $PL_{d(dB)}$ is the reference path loss at distance $d_{(m)}$, which is calculated based on the FSPL formula, $f_{(Hz)}$ denotes the frequency of the signal, and $c_{(m/s)}$ typically refers to the speed of signal propagation in free space. PL represents the general path loss, which is determined by the reference path loss at $d_{(m)}$ and the path loss index n in different environments.

To align with real-world scenarios and adapt to different communication environments, especially in cloud-edge communication scenarios, we introduce a dynamic multiplier D_{multi} to adjust the communication latency to the cloud and edge in EPFLU. When it comes to interactions between servers, we use the term $Latency_{(s)}$ which is a part of the overall communication time that needs to be evaluated later. In our simulations, we choose to base D_{multi} on the average distance between edge nodes D_{avg} to evaluate the communication latency to the cloud:

$$D_{multi} = Base_{multi} \cdot \left(\frac{D_{avg}}{Ref_{distance}} \right), \tag{3}$$

where $Base_{multi}$ is the base value of the dynamic multiplier, adjusted according to the communication situation between the cloud and edge under different environments, and $Ref_{distance}$ is a reference distance used to standardize the distance ratio.

Integrating path loss and the dynamic multiplier, the communication latency $L_{com(s)}$ and energy consumption $E_{com(J)}$ can be calculated through the following equations:

$$G = g_0 \times 10^{-\frac{PL}{10}} + EL, \tag{4}$$

$$L_{com} = D_{multi} \cdot \frac{S}{B \cdot \log_2(1 + \frac{G \cdot P}{N})}, \tag{5}$$

$$E_{com} = P \cdot L_{com}. \tag{6}$$

Here, G represents the adjusted channel gain, calculated by converting the path loss PL from decibel units back to linear units using the reference gain g_0. To simulate the impact of distance on loss, EPFLU adds an extra loss value extra loss (EL) and introduces a minimum gain value to prevent G from being too small to be calculated. $S_{(bits)}$ represents the size of the model being uploaded, $B_{(Hz)}$ represents the communication bandwidth, $P_{(Watt)}$ is the transmitter power, and $N_{(Watt)}$ is the noise power.

In addition to the design for communication time and energy consumption, EPFLU also accounts for the aspects of local computation time and

energy consumption [15], enabling a comprehensive evaluation of EPFLU's overall performance and energy efficiency. The local computation time is calculated using $time_{local} = \frac{c_{cpu} \cdot D_{local}}{f_{cpu}}$, where $D_{local(bits)}$ represents the amount of data processed in a single local iteration, $c_{cpu(cycles/bit)}$ denotes the number of CPU cycles required to process data, and $f_{cpu(GHz)}$ is the CPU cycle frequency. The calculation for local computation energy is given by $energy_{local} = \frac{\alpha}{2} \cdot (c_{cpu} \cdot D_{local} \cdot f_{cpu}^2)$, with α indicating the effective capacitance, which is the energy required per CPU cycle. In addition, since the size of the models and parameters changes before and after aggregation, we added a re-measurement of the updated parts on top of the original communication transmission considerations in EPFLU. EPFLU recalculates the size of the models and parameters after aggregation and uses the updated sizes when the models return to the clients (both the complete model for FL and the temporary model for P2PFL) to ensure the accuracy of our experiment. Thus, by integrating both communication and local computation in terms of time and energy consumption calculations, it can provide a comprehensive evaluation framework for the performance and energy efficiency of IoT devices when performing tasks.

5.3 Experimental Settings

The system environment of all simulations[1] is Windows 11, Intel(R) Core(TM) i7-9700K CPU @ 3.60GHz, 32.0GB of RAM, NVIDIA GeForce RTX 2070. The deep learning environment configuration is PyTorch1.7.0-cuda11.0.

5.3.1 Datasets and model settings

EPFLU can be applied to various tasks and scenarios, but to better control variables, we use the most typical classification task in this experiment to better align with the baselines we select. EPFLU employs two datasets, namely MNIST [11] and CIFAR-10 [9]. Given that CIFAR-10 is more suited to machine learning tasks, EPFLU will utilize it to evaluate model performance and convergence. Performance evaluation involves comparing the final personalized performance of the model once it has reached stable convergence. For the performance evaluation, we select a simple CNN structure. Given that our simulation framework considers not only the time and

[1]For detailed model and parameter setting, data distribution operations, and complete experimental records, please visit https://github.com/XiangchiSong/JWE_EPFLU.

energy consumption of local data computation but also the time and energy consumption of the communication process, EPFLU uses models with simpler structures to reduce the cost of the transmission process and to facilitate this process. Since MNIST has a simpler structure and lower computational requirements, it will be used to evaluate communication and computational consumption. Here, the evaluation metric compares the communication and computational consumption when reaching a preset test accuracy threshold. For the evaluation of communication and computational consumption, we utilize a classification structure comprising a fully connected (FC) layer, a batch normalization (BN) layer which will keep the private parameters, another FC layer, and a Softmax activation output.

Since most real-world scenarios are heterogeneous, the basic independent and identically distributed (IID) and non-IID distributions are rarely encountered. We consider the extreme data distribution scenarios that better reflect the actual situation: imbalanced-non-IID and imbalanced-mixed-IID. Initially, we sort the labels and assign them identical indices to ensure the consistency of data classes between the training sets and test sets for subsequent personalized accuracy testing. The data distribution in EPFLU is based on the unbalanced setting because the amount of data owned by users in real scenarios varies. This is achieved by dividing the dataset into shards and then randomly selecting at least 1 shard and at most 2 shards for each client. In EPFLU, non-IID conditions are achieved by splitting the dataset and randomly allocating different classes to each client. Mixed-IID conditions are established by setting a ratio of IID clients, which have access to all data classes, and to non-IID clients, with a predefined ratio of 1:1 in EPFLU. It is noteworthy that in the mixed-IID setting, the number of shards selected for clients is not controlled for IID or non-IID types, thus both IID and non-IID clients will have different amounts of data.

5.3.2 Evaluation metrics

In our evaluation of EPFLU, we focus on two main aspects: task performance, and computational and communication consumption.

Performance evaluations are centered on the user perspective, hence we use private test sets available on user clients to ensure user-centric accuracy. We ensure that each user u has its own private test set $Test_u$ with the same data distribution as the training set. For each user u, test accuracy is calculated using $Test_u$ after personalized enhancements. The final user model $Test\ Accuracy$ for each iteration is determined by averaging the test

accuracies across all participating users (N). The formula is as follows:

$$Accuracy_u = \frac{\text{Number of correct predictions by the model in } Test_u}{\text{Total number of samples in } Test_u}, \quad (7)$$

$$Test\ Accuracy = \frac{1}{N}\sum_{u=1}^{N} Accuracy_u. \quad (8)$$

By establishing this metric, the aforementioned evaluation metrics can be represented by the user model accuracy: the result of the performance evaluation is the accuracy of the user model after personalized enhancement, and the result of the computational and communication consumption evaluation is the time and energy consumption of the system when the user model accuracy in the task reaches the preset test accuracy threshold.

5.3.3 Baseline selection

EPFLU evaluates its approach against three baselines, including the basic version of FL, the FL method with personalized processing, and a P2PFL algorithm.

The first is vertical-FL (FedAvg), a classic FL architecture [17] that facilitates model sharing among different clients through vertical aggregation, without the direct exchange of data. The process of system operation can refer to the partition part of FL in Figure 2. The model transmitted here is the complete model of the client. We use this most basic method for subsequent convergence and accuracy testing of EPFLU.

The second is vertical-adaptive federated optimization (FedAdam), which combines the original Adam optimization [21] and MTFL(FedAdam) [19] vertical model aggregation algorithms. FedAdam achieves personalization of user models through adaptive parameters. The process of system operation can refer to the partition part of FL in Figure 2. However only the local consensus model is aggregated and distributed between the edge and cloud, and all private parameters are retained locally. Since both algorithms use personalized processing, the effect of the horizontal P2P structure is tested by comparing with this approach.

The third is horizontal-privacy-preserving training (PPT), a P2PFL protocol [2] that transmits and updates local model parameters using a single-hop P2P approach. We have omitted the original use of symmetric cryptographic systems in PPT, which ensures secure communication between network nodes, to maintain the accuracy of the comparative experiments. The P2P partition of Figure 2 shows the structure of PPT. All clients conduct local

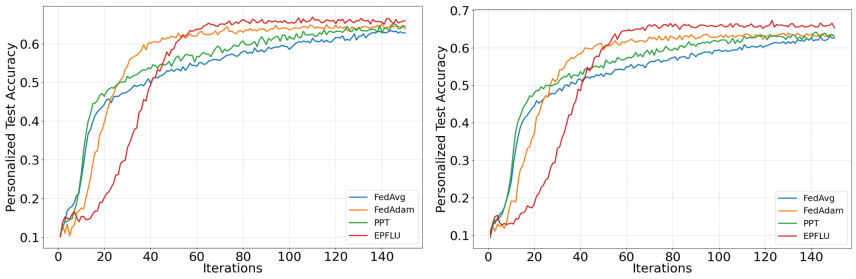
training in each round, but only a subset of clients sampled by the coordinator participate in the aggregation for this round (as the topology indicated by the red lines in Figure 1). Subsequently, the model accumulated at the last client on the route serves as the global model for the next round. Since both algorithms are horizontal structures, EPFLU is compared with this to reflect the effectiveness of retaining personalized parameters and performing personalized enhancement.

6 Experiment Result, Analysis and Discussion

To mitigate the randomness of client sampling we conducted independent repeated experiments. We performed random sampling and conducted five times, taking the average of these results as the final outcome. Additionally, by controlling the seed, we could ensure that the sampling behavior for different algorithms within each sub-experiment was consistent, while client sampling differed across sub-experiments. This process guarantees that all variables, except for algorithm behavior, remain consistent in each experiment, and that the sampling behavior is not repeated across independent experiments.

6.1 Performance

We analyzed the performance of various models based on the CIFAR-10 dataset, divided into imbalanced-non-IID and imbalanced-mixed-IID data distribution scenarios, as shown in Figure 5. After fine-tuning the hyperparameters under this setting, all four algorithms finished convergence and



(a) Imbalanced-Non-IID (b) Imbalanced-Mixed-IID

Figure 5 A performance comparison of models based on the CIFAR-10 dataset. The number of clients is 500, the sampling rate is 0.3, and iterations are intercepted to the stable performance position of 150 rounds.

Table 1 Comparison of different Algs for personalized test accuracy (%)

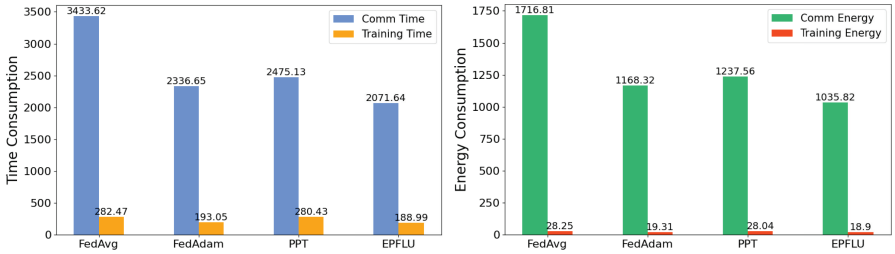
Methods	Imbalanced-non-IID			Imbalanced-mixed-IID		
	$C = 0.3$	$C = 0.5$	$C = 0.7$	$C = 0.3$	$C = 0.5$	$C = 0.7$
FedAvg	62.8	63.1	63.3	62.5	62.4	63.1
FedAdam	63.9	63.3	62.9	63.1	63.5	**63.9**
PPT	64.4	62.8	63.1	63.1	64.0	63.4
EPFLU	**65.9**	**64.7**	**64.0**	**65.3**	**64.4**	63.3

achieved an accuracy of over 60%. FedAvg performs aggregation after all models are trained and uploaded, achieving the fastest initial convergence speed. Along with PPT, both methods stabilize at around 20 iterations and gradually reach their peak. In PPT, since there is no central server, aggregation starts with training and there is no need to wait for all models to be uploaded. Although only clients are sampled for aggregation, all clients must participate during training, so the convergence speed is faster. FedAdam introduces a personalized process based on FedAvg, and ultimately achieves higher performance than FedAvg. However, since EPFLU adopts a P2PFL approach based on personalized information supplementation, the convergence speed is initially slower than that of centralized federated aggregation. Under these two extreme data heterogeneity scenarios, EPFLU performs the best. EPFLU is about 3% higher than traditional FedAvg in both scenarios, and 3% higher than all baselines in Figure 5b. From the results, we found that the personalization techniques used by FedAdam and EPFLU did not directly accelerate model convergence. On the contrary, the negative effect was that the separation and supplementation of personalized parameters slightly slowed down the convergence speed. However, this ultimately benefited the performance of the users' personalized models.

We also tested the impact of different sampling rates ($C = 0.3, 0.5, 0.7$) on personalized accuracy in large-scale distributed environments. As shown in Table 1, EPFLU mostly achieved the best performance in both data distribution scenarios. However, FedAdam slightly outperformed EPFLU with a sampling rate of $C = 0.7$ in the imbalanced-mixed-IID scenario, demonstrating FedAdam's excellent performance in large-scale distributed systems [19].

6.2 Communication and Computational Consumption

For the comparison of communication and computational consumption, we selected the imbalanced-mixed-IID scenario, which is more challenging



(a) Time Consumption (b) Energy Consumption

Figure 6 A comparison of communication and computational consumption based on the MNIST dataset in the imbalanced-mixed-IID scenario. The units of communication and computation are seconds and Joules respectively. The accuracy threshold to stop accumulation is set to 95% and other settings are the same as before.

since it has more complex data heterogeneity than imbalanced-non-IID. We consider the aggregation process complete from the time the initial model preparation begins until the accuracy is above the threshold. These results in Figure 6 are the cumulative communication and computational consumption. In Figure 6a and Figure 6b, it is not surprising that the basic FedAvg has the highest communication and computational consumption. FedAdam achieves relatively satisfactory results, saving 32% compared to the traditional FL. Furthermore, FedAdam even exhibits lower consumption than PPT. The reason is that although PPT has lower communication costs between edge clients, FedAdam's superior learning performance results in faster convergence, achieving the required accuracy in fewer iterations. This indicates that the benefits of reducing the number of iterations outweigh the communication savings brought by the different aggregation algorithms. However, EPFLU demonstrates its superiority by applying the personalization process to the horizontal P2P structure. EPFLU mitigates model degradation caused by data heterogeneity through the transfer of consensus models by P2P topology implemented among edge clients, ultimately achieving rapid convergence and efficient communication. This leads to substantial savings in communication resources and time. From the results in Figure 6a compared to the selected P2PFL method (PPT), EPFLU reduces the communication time to reach model personalization convergence by 16%. Furthermore, compared to the traditional FL method (FedAvg) and advanced FL method (FedAdam), the horizontal P2P transmission strategy provides EPFLU with a communication time improvement of 39% and 11%, respectively.

6.3 Limitation Discussion

The main limitation of this work is the performance fluctuations caused by data heterogeneity. Due to EPFLU's use of random sampling and sequential horizontal aggregation, it is difficult to predict the data distribution of the next client. Therefore, when aggregating with clients with abnormal data distribution, the convergence process may pause and even have a negative impact on the final model. Since personalized accuracy evaluation occurs on the user client, when the model is aggregated with updates from the next client with extreme data distribution, the presence of this kind of client that is unknown to the previous clients will significantly increase the training and testing losses of the user personalized model, leading to stagnation or even degradation in the convergence process. A potential solution is to improve sampling behavior, such as adjusting the client selection strategy based on learned client correlations [26] or dynamically altering the selection probability of specific clients according to the aggregation progress [29], to address the convergence problem in heterogeneous scenarios. Given our limited knowledge of the characteristics of local data stored on edge devices in real-world scenarios, it is a challenging task to design a dynamic adjustment strategy that maximizes the use of client data without compromising privacy, while balancing various influencing factors, and making it suitable for real-world applications.

7 Conclusion

This paper introduces EPFLU, a P2PFL framework, as a robust solution to enhance communication and computational efficiency in large-scale hierarchical IoT systems within edge-cloud environments. By transitioning from traditional vertical FL to the horizontal P2P structure, EPFLU not only avoids the communication bottlenecks of traditional FL but also incorporates personalized enhancement processes supplemented by private information, designed to meet the model needs of clients from the user perspective. We conducted extensive evaluations on the MNIST and CIFAR-10 datasets using the EUA dataset, which is based on real-world edge server locations. The results indicate that EPFLU achieves personalized model accuracy that mostly outperformed the baselines, while significantly excelling in communication and computational consumption, especially under challenging data distribution scenarios. We also found that the communication process is the bottleneck

of large-scale IoT systems, and EPFLU also has limitations in this regard. In the future, we will focus on adapting the network topology according to changes in network conditions, ensuring P2P interactions between clients with minimal communication consumption, and considering improving the client sampling problem mentioned above.

Acknowledgements

This research was partly supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2024-2020-0-01795) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation) and IITP grant funded by the Korea government (MSIT) (No. RS-2024-00406245, Development of Software-Defined Infrastructure Technologies for Future Mobility).

References

[1] Samiul Alam, Tuo Zhang, Tiantian Feng, Hui Shen, Zhichao Cao, Dong Zhao, JeongGil Ko, Kiran Somasundaram, Shrikanth S Narayanan, Salman Avestimehr, et al. Fedaiot: A federated learning benchmark for artificial intelligence of things. *arXiv preprint arXiv:2310.00109*, 2023.

[2] Qian Chen, Zilong Wang, Wenjing Zhang, and Xiaodong Lin. Ppt: A privacy-preserving global model training protocol for federated learning in p2p networks. *Computers & Security*, 124:102966, 2023.

[3] Qian Chen, Zilong Wang, Yilin Zhou, Jiawei Chen, Dan Xiao, and Xiaodong Lin. Cfl: Cluster federated learning in large-scale peer-to-peer networks. In *International Conference on Information Security*, pages 464–472. Springer, 2022.

[4] Yiqiang Chen, Xin Qin, Jindong Wang, Chaohui Yu, and Wen Gao. Fedhealth: A federated transfer learning framework for wearable healthcare. *IEEE Intelligent Systems*, 35(4):83–93, 2020.

[5] Qianlong Dang, Guanghui Zhang, Ling Wang, Shuai Yang, and Tao Zhan. Hybrid iot device selection with knowledge transfer for federated learning. *IEEE Internet of Things Journal*, 2023.

[6] Tiantian Feng, Digbalay Bose, Tuo Zhang, Rajat Hebbar, Anil Ramakrishna, Rahul Gupta, Mi Zhang, Salman Avestimehr, and Shrikanth Narayanan. Fedmultimodal: A benchmark for multimodal federated

learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4035–4045, 2023.

[7] Harald T Friis. A note on a simple transmission formula. *Proceedings of the IRE*, 34(5):254–256, 1946.

[8] Masaharu Hata. Empirical formula for propagation loss in land mobile radio services. *IEEE transactions on Vehicular Technology*, 29(3):317–325, 1980.

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

[10] Phu Lai, Qiang He, Mohamed Abdelrazek, Feifei Chen, John Hosking, John Grundy, and Yun Yang. Optimal edge user allocation in edge computing with variable sized vector bin packing. In *Service-Oriented Computing: 16th International Conference, ICSOC 2018, Hangzhou, China, November 12-15, 2018, Proceedings 16*, pages 230–245. Springer, 2018.

[11] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[12] Ang Li, Jingwei Sun, Binghui Wang, Lin Duan, Sicheng Li, Yiran Chen, and Hai Li. Lotteryfl: Empower edge intelligence with personalized and communication-efficient federated learning. In *2021 IEEE/ACM Symposium on Edge Computing (SEC)*, pages 68–79. IEEE, 2021.

[13] Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019.

[14] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *Advances in neural information processing systems*, 30, 2017.

[15] Lumin Liu, Jun Zhang, SH Song, and Khaled B Letaief. Client-edge-cloud hierarchical federated learning. In *ICC 2020-2020 IEEE international conference on communications (ICC)*, pages 1–6. IEEE, 2020.

[16] Tianyi Liu, Ruyu Luo, Fangmin Xu, Chaoqiong Fan, and Chenglin Zhao. Distributed learning based joint communication and computation strategy of iot devices in smart cities. *Sensors*, 20(4):973, 2020.

[17] 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.

[18] Jed Mills, Jia Hu, and Geyong Min. Communication-efficient federated learning for wireless edge intelligence in iot. *IEEE Internet of Things Journal*, 7(7):5986–5994, 2019.

[19] Jed Mills, Jia Hu, and Geyong Min. Multi-task federated learning for personalised deep neural networks in edge computing. *IEEE Transactions on Parallel and Distributed Systems*, 33(3):630–641, 2021.

[20] Pramod Kaushik Mudrakarta, Mark Sandler, Andrey Zhmoginov, and Andrew Howard. K for the price of 1: Parameter-efficient multi-task and transfer learning. *arXiv preprint arXiv:1810.10703*, 2018.

[21] SashankJ. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and H.Brendan McMahan. Adaptive federated optimization. *arXiv: Learning,arXiv: Learning*, Feb 2020.

[22] Abhijit Guha Roy, Shayan Siddiqui, Sebastian Pölsterl, Nassir Navab, and Christian Wachinger. Braintorrent: A peer-to-peer environment for decentralized federated learning. *arXiv preprint arXiv:1905.06731*, 2019.

[23] Atul Sharma, Joshua C Zhao, Wei Chen, Qiang Qiu, Saurabh Bagchi, and Somali Chaterji. How to learn collaboratively-federated learning to peer-to-peer learning and what's at stake. In *2023 53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks-Supplemental Volume (DSN-S)*, pages 122–126. IEEE, 2023.

[24] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. *Advances in neural information processing systems*, 30, 2017.

[25] Xiangchi Song, Zhaoyan Wang, KyeongDeok Baek, and In-Young Ko. Epflu: Efficient peer-to-peer federated learning for personalized user models in edge-cloud environments. In *ICWE 2024 International Workshops, BECS and WALS, Tampere, Finland*, June 2024.

[26] Minxue Tang, Xuefei Ning, Yitu Wang, Yu Wang, and Yiran Chen. Fedgp: Correlation-based active client selection strategy for heterogeneous federated learning. *arXiv preprint arXiv:2103.13822*, 2021.

[27] Nguyen H Tran, Wei Bao, Albert Zomaya, Minh NH Nguyen, and Choong Seon Hong. Federated learning over wireless networks: Optimization model design and analysis. In *IEEE INFOCOM 2019-IEEE conference on computer communications*, pages 1387–1395. IEEE, 2019.

[28] Kaibin Wang, Qiang He, Feifei Chen, Chunyang Chen, Faliang Huang, Hai Jin, and Yun Yang. Flexifed: Personalized federated learning for edge clients with heterogeneous model architectures. In *Proceedings of the ACM Web Conference 2023*, pages 2979–2990, 2023.

[29] Hongda Wu and Ping Wang. Node selection toward faster convergence for federated learning on non-iid data. *IEEE Transactions on Network Science and Engineering*, 9(5):3099–3111, 2022.

[30] Qiong Wu, Kaiwen He, and Xu Chen. Personalized federated learning for intelligent iot applications: A cloud-edge based framework. *IEEE Open Journal of the Computer Society*, 1:35–44, 2020.

[31] Tuo Zhang, Tiantian Feng, Samiul Alam, Sunwoo Lee, Mi Zhang, Shrikanth S Narayanan, and Salman Avestimehr. Fedaudio: A federated learning benchmark for audio tasks. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

Biographies

Xiangchi Song is enrolled in the integrated program for Master's and Ph.D. degrees at the Web Engineering and Service Computing Lab, School of Computing, Korea Advanced Institute of Science and Technology (KAIST). His research interests include federated learning, P2P learning, hierarchical edge computing and edge cloud environments.

Zhaoyan Wang is enrolled in the Master's program at the Web Engineering and Service Computing Lab, School of Computing, Korea Advanced Institute of Science and Technology (KAIST). His research interests include service computing, spatial-temporal data analysis, and edge-cloud collaboration.

KyeongDeok Baek is a post-doctoral researcher at the School of Computing at the Korea Advanced Institute of Science and Technology (KAIST) in Daejeon, Korea. He received his Ph.D. in Computer Science from the School of Computing at KAIST in 2024. His recent research focuses on realizing interactive Internet of Things services in public spaces by extending multi-agent reinforcement learning and human–computer interaction approaches.

In-Young Ko is a professor at the School of Computing at the Korea Advanced Institute of Science and Technology (KAIST) in Daejeon, Korea. He received his Ph.D. in computer science from the University of Southern California (USC) in 2003. His research interests include services computing, web engineering, and software engineering. His recent research focuses on service-oriented software development in large-scale and distributed system environments such as the web, Internet of Things (IoT), and edge cloud environments. He is a member of the IEEE.