Received 1 December 2023, accepted 9 January 2024, date of publication 17 January 2024, date of current version 23 January 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3353617

RESEARCH ARTICLE

pFedLN: Personalized Federated Learning Framework With Layer-Wised and Neighbor-Based Aggregation for QoS Prediction

WEIWEI SHE, JIANLONG XU, (Member, IEEE), MENGQING JIN, AND YUELONG LIU
Department of Computer Science and Technology, Shantou University, Shantou 515063, China

Corresponding author: Jianlong Xu (xujianlong@stu.edu.cn)

This research was financially supported by Guangdong Basic and Applied Basic Research Foundation (No. 2023A1515010707, 2021A1515012527), Special Projects in Key Fields of Guangdong Universities (No. 2022ZDZX1008, 2020ZDZX3073), Guangdong Province Special Fund for Science and Technology (''major special projects + task list'') Project (No. STKJ202209017), and in part by the Key Scientific Research Project of Universities in Guangdong Province (No. 2020ZDZX3028).

ABSTRACT In the era of a more advanced and intelligent Internet, the highly sophisticated service-oriented internet provides users with a diverse array of similar services. Accurate Quality of Service (QoS) prediction plays a pivotal role in helping users choose the optimal service from a multitude of available options. Traditional federated learning models offer a secure method for multiple clients to collaborate on QoS predictions. However, these models still employ a uniform approach that overlooks the unique requirements of individual clients. In order to meet the different needs of a wide range of customers for models, we propose an innovative personalized federated learning framework with layer-wised and neighbor-based aggregation for QoS prediction (pFedLN). In the proposed framework, we consider the privacy and functional disparities among layers in neural network models and employ diverse aggregation strategies for layers serving different functions. In addition, the similarity between neighbors will be taken into account during the aggregation process. This results in the creation of personalized models for each client that better align with their specific requirements. Sufficient experiments are conducted on a real-world dataset and the results indicate that our approach have a clear advantage in improving the effectiveness of personalization compared to existing approaches.

INDEX TERMS Internet of service, personalized federated learning, QoS prediction.

I. INTRODUCTION

With the continuous development of intelligent service technologies, the concepts of the ''Internet of Things'' and ''Everything as a Service'' are reaching new levels of maturity. The growing interconnectivity between numerous services has given rise to the Internet of Services (IoS) [1]. In the IoS environment, the interoperability among a wide range of services, domains, and business processes and rules has resulted in the emergence of a highly complex service ecosystem [2]. However, effectively identifying and extracting high-quality services can be a challenging task since the the diversity of user demands and the variability of service quality available in the service ecosystem. To address this challenge, utilizing QoS metrics is the most widely accepted approach. The QoS metrics describe the non-functional attributes of a service, such as response time (RT) and throughput, and are regarded as crucial indicators of service quality [3], [4].

Studies [5] and [6] have focused on analyzing historical QoS data and predicting the unknown QoS values to provide clients with personalized recommendations for high-quality

The associate editor coordinating the review of this manuscript and approving it for publication was Claudio Agostino Ardagna.

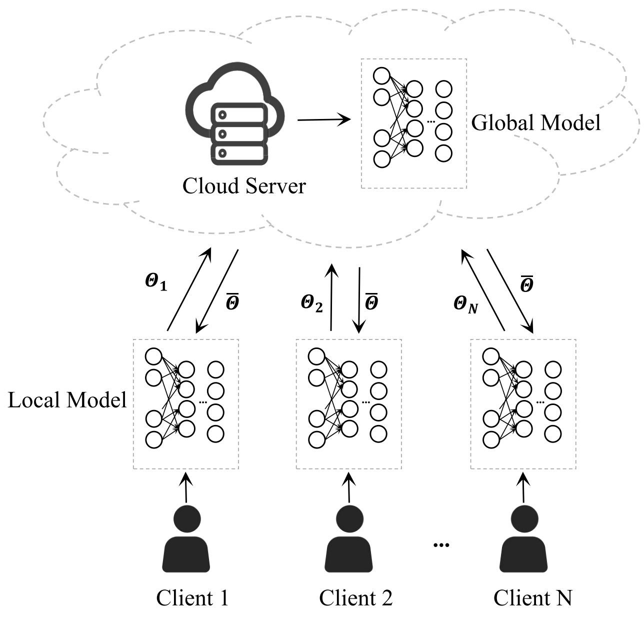


FIGURE 1. Pictorial view of client/server architecture in federated learning system.

services. They rely on the centralized collection of extensive client data, which provides a substantial foundation for data analysis and model training, significantly enhancing prediction efficiency. But while we enjoy the convenience brought by centralized data collection, we are also increasingly aware of the significant challenges in data security and privacy protection.

Federated learning provides an effective solution. As a distributed machine learning paradigm, federated learning is gaining widespread attention. The core idea of federated learning is to share model parameters rather than private raw datasets for collaborative training, so as to protect user privacy [7]. In a typical federated learning system, clients retain their raw data locally and conduct local model training. Additionally, clients can share their model parameters with a cloud server to participate in collaborative training. The details of client/server architecture in federated learning system is shown in Figure 1. The model training will contains three parts: (1) clients are allowed to retain the raw data locally and perform model training locally; (2) client i can share the model parameters Θ_i to the cloud server and participate into the collaborative training; (3) the cloud server will aggregate the parameters received from clients and generate a set of global parameters $\bar{\Theta}$ for clients. This three parts will be conducted for iterations until the model become convergence. It not only greatly reduces the pressure on data security, but also provides an effective way to handle distributed data.

It is worth noting that the aggregation way has a significant impact on the performance of federated learning. The traditional approach for aggregation in federated learning is known as FedAvg [7]. This method achieves model aggregation by directly computing the average of parameters. Under the FedAvg framework, local parameters are transmitted to the server for aggregation after several rounds of local model training, which represents an improvement over the previous approach of uploading parameters after each individual round. The FedAvg algorithm not only prioritizes the privacy of the participants, but also reduces the computational burden on the server by somewhat reducing the data transfer between the client and the server. This positions federated learning with FedAvg as highly advantageous for managing distributed learning scenarios, especially those involving sensitive data. However, despite its widespread practical use, FedAvg does suffer from two notable drawbacks:

A. FAILING TO MEET PERSONALIZED NEEDS

The FedAvg approach uses average aggregation to construct a global model, which will dilute the variability of the clients. In real-world application scenarios, sharing a global model for all clients may lead to slow convergence or poor inference performance because the needs of different clients are different [8]. As a result, the approach of using an unified model for multiple clients is impractical and it does not satisfy the individual needs of the clients.

B. NOT SUITABLE FOR NON-IID DATA

The FedAvg method assumes that the data distributions of all participants are the same. However, a characteristic of federated learning is that client data tends to exhibit non-identical and independently distributed (non-IID) patterns [7]. In non-IID environments, the distribution of data among participants can vary significantly. As a result, the average aggregation method, which assumes equal contributions from all local models, becomes inapplicable, and its use may result in a degradation of model performance [9].

In response to the challenges posed by personalized requirements and non-IID data, personalized federated learning has emerged as an effective solution. Personalized methods in federated learning perform differentiated aggregation strategies by taking into account differences between participants. They can balance the diverse needs of participants, ultimately yielding more precise prediction models.

We aim to leverage the advantages of personalized federated learning to construct an efficient QoS (Quality of Service) neural network prediction model. Within personalized federated learning approaches, we have discovered an effective parameter aggregation strategy, which involves aggregating only a specific portion of the network layers instead of aggregating the entire model. This strategy primarily from the fact that different layers of a neural network can have different utilities, e.g., the shallow layers focus more on local feature extraction, while the deeper layers are for extracting global features [10]. Therefore, we can adopt a new aggregation method, integrate the idea based on the difference between layers, and adopt a partial aggregation strategy for our personalized model. In this strategy, the shallow network responsible for capturing fundamental, generic features is shared and aggregated, while the more personalized deeper layers are retained and updated locally. This strategy strikes a balance in amalgamating global and local knowledge, ultimately providing more precise personalized recommendations to users.

In this paper, we propose an innovative personalized federated learning framework for QoS prediction, named layer-wised and neighbor-based aggregation for personalized federated learning (pFedLN). We combine the distinctions in layer functionalities and the interrelation among clients, resulting in highly personalized predictions. Additionally, we have achieved significant advancements in data privacy protection and learning performance. This paper highlights our improved ability to personalised training for federated learning model, emphasising substantial benefits in the area of QoS prediction. Our solution can be integrated into the federated learning application scenario to address the challenge of personalized training across multiple clients. The main contributions of the paper are as follows:

1) We propose a innovative layer-wised and neighbor-based aggregation scheme for personalized federated learning model. Our scheme takes into account the privacy and functionality of neural network layers and proposes a novel hierarchical approach, innovatively employing different aggregation strategies for layers with distinct functionalities.
2) We perform parameter aggregation based on neighbor similarity, using contextual similarity to create the adjacency matrix A. This similarity is then employed to assign weights to client aggregation, thereby enhancing the personalization effect.
3) We verify the effectiveness of pFedLN through extensive experiments. And the results show that our method not only improves the personalized prediction effect of the model, but also greatly improves the convergence speed.

The paper is organized as follows. In Section II, we provide a review of the current related work. In Section III, we provide a detailed description of our proposed solution. The performance of our scheme under different settings is evaluated in Section IV. Finally, in Section V, we conclude the paper.

II. RELATED WORK

In this section, we focus on aggregation methods in personalized federated learning. Traditional federated learning methods typically employ FedAvg aggregation. However, this does not apply to scenarios where clients have non-independent co-distribution and different requirements between clients. In order to meet the specific needs of each participating client, client personalization has become a core topic in the federal learning field.

In the field of personalized federated learning, numerous studies concentrate on factors such as client variability, heterogeneous data and models, and the enhancement of privacy protection. Researchers are actively working to achieve highly personalized learning performance by making adjustments to the framework, improving optimization algorithms, and implementing additional privacy safeguards. Based on the nature of the problems they address and the methodologies employed, these efforts can be broadly categorized into two main groups: methods for heterogeneous data problems and methods based on model framework adjustment.

A. METHODS DESIGNED FOR HETEROGENEOUS DATA PROBLEMS

Duan et al. [11] proposed a self-balancing federated learning framework called Astraea. The primary objective is to mitigate training bias resulting from imbalanced data distributions. The core concept of Astraea involves adopting a novel role mediation approach to align the client's data distribution with a standardized data distribution. However, this method require the federated learning server to access statistical information regarding the client's local data distribution (e.g., class size, mean, and standard deviation). This requirement raises potential concerns regarding privacy policy compliance.

B. METHODS BASED ON MODEL FRAMEWORK ADJUSTMENT

Federated learning based on model framework adjustment can be divided into single-model and multiple-model approaches according to aggregation results.

In single-model approach, the optimization combines both client model optimization and global model optimization, resulting in a unified global model. This kind of approaches extend from traditional FL algorithms such as FedAvg [7] and include four different methods: local fine-tuning [12], [13], regularization [14], meta-learning [15], and parameter decomposition [16].

In contrast to the single-model approach, the multi-model approach adopts the idea that different clients maintain their own training models instead of a unified global model. It takes the different needs of the clients into account in order to achieve personalization. The multi-model approach includes two strategies: 1) Clients are segmented into multiple groups [9]. For each group, the server maintains an aggregate model and performing model optimization tailored to group members. 2) Each client holds a unique model for individualized requirements, and the server only aggregates the select parameters.

Inspired by prior research and considering the diversity in client data, varying requirements in federated learning, and the efficiency of information transfer, we embrace the concept of multiple models, where each user possesses a unique model. We apply this approach to neural network-based personalized federated learning. Huang et al. [17] computed the similarity between client models and use this similarity to aggregate similar client models. However, in scenarios with a large number of users, calculating model similarity can become time-consuming. Moreover, common methods for measuring model similarity often involve assessing model distances. Yet, due to functional differences among neural network layers, these distance-based methods may overlook inter-layer distinctions, potentially resulting in inaccuracies in personalization.

Each layer in a neural network model has a different function. Shallow networks are usually responsible for extracting basic, generalized features, while deeper networks focus on task-specific and individual feature learning. Based on this idea, Arivazhagan et al. [12] proposed a base + personalization layer approach named FedPer, for federated training of deep feed forward neural networks. They directly divided the neural network into the base layers and the personalization layers. They train the base layer locally and perform global average aggregation on the server, while the personalization layer is updated only locally. It is to be noted that this approach requires explicit planning of the base and personalization layers at the initial stage. In a different vein, Ma et al. [8] presented pFedLA, a method that incorporates a dedicated supernetwork on the server side for each client. This supernetwork computes client similarity, facilitating the learning of weights for cross-client layers during personalized federated learning (pFL) training. This enhances personalization for non-IID datasets. Unlike FedPer, pFedLA eliminates the need for manual parameter configuration, making its supernetwork more adaptable to complex network parameter variations. Chen et al. [18] proposed a structured federated learning framework named SFL, which enhances the knowledge sharing in pFL by leveraging the graph-based structural information among clients. They also proposed generating a GCN network from the server side to extend and learn hidden relationships between clients. This approach inspired us that we can further enhance the personalization effect of the model by considering the relationship between clients.

Inspired by FedPer, pFedLA, and SFL, we consider the functional distinctions among each neural network layer and the similarity relationships between clients, motivating the development of an efficient personalized federated learning model for QoS prediction.

III. pFedLN FRAMEWORK

In this section, we introduce the design of our pFedLN framework, which employs layer-wise and neighbor-based aggregation for federated learning in the context of QoS prediction. To provide a comprehensive understanding of our approach, we begin with the presentation of relevant definitions for the problem in the first subsection. Following that, in the second subsection, we provide a detailed description of the method's implementation.

TABLE 1. Main notations and description.

Symbol	Description
D	A target raw dataset for analysis.
N	The amount of clients.
i	The index of client.
j	The index of service.
$r_{i,j}$	The qos value of client i and client j.
$r'_{i,j}$	The predicted qos value of client i and client j.
m_i	The services set which the client i have invoked.
Θ_i	The model parameters of client i.
$\bar{\Theta}_i$	The aggregated parameter from the cloud server.
$\mathcal{L}_i$	The training loss of client i.
O	Observed records in the training dataset.
O_i	The gathered invocations matrix of client i.
$\|\mathrm{O}_i\|$	The number of valid qos values in O_i.
k	The k-th layer in the neural network model.
$\theta_i^{l_k}$	The k-th layer parameters of client i.
l_{base}	The base layers of neural network model.
l_{per}	The personalized layers of neural network model.
$\theta_i^{l_k,t}$	The k-th layer parameters of client i at t-th iteration.
$\bar{\theta}_i^{l_k}$	The k-th layer parameters of client i aggregated by the server.
$\bar{\theta}_i^{l_k,t}$	The k-th layer parameters of client i aggregated by the server at t-th iteration.
Θ_i^t	The model parameters of client i at t-th iteration.
A	The adjacency matrix consisting of all clients.
$A_{i,j}$	The similarity between client i and client j.

A. PROBLEM FORMULATION

In this subsection, we will state the relevant definitions of the QoS prediction problem. Considering N clients with non-IID datasets, let $D_i = \{i, j, r_{i,j}\}_{j=1}^{m_i} (1 \le i \le N)$ be the dataset on the i-th client, where $r_{i,j}$ is the QoS value of the i-th client and j-th service, m_i is the services set that the i-th client have invoked. Let Θ_i represent the model parameters of the i-th client, the objective of pFedLN can be formulated as:

$$\arg\min_{\Theta_i} \sum_{i=1}^{N} \mathcal{L}_i(\Theta_i)\,, \tag{1}$$

where $\mathcal{L}_i$ is loss of the i-th clients associated with dataset D_i. $\mathcal{L}_i$ is calculated as:

$$\mathcal{L}_i(\Theta_i) = \frac{1}{|O_i|} \sum_{(i,j)\in O_i} |r_{i,j} - r'_{i,j}|\,, \tag{2}$$

where $r_{i,j}$ denotes the real QoS value of i-th clients and j-th service, $r'_{i,j}$ denotes the predict value of i-th clients and j-th service, O_i denotes the gathered invocations matrix and $|O_i|$ is the number of valid QoS values in O_i.

Equation (1) is our objective function, optimizing the minimum of the sum of the overall client model losses. The loss of each client is defined as the average of the difference between all predicted and true values, and the difference is calculated by the loss function (2). To optimize the objective loss function, we adopt the Adam [19] optimizer.

B. LAYER-WISED AND NEIGHBOR-BASED AGGREGATION OF pFedLN

As shown in Figure 2, we propose a novel aggregation strategy applied in the federated learning framework. We aim

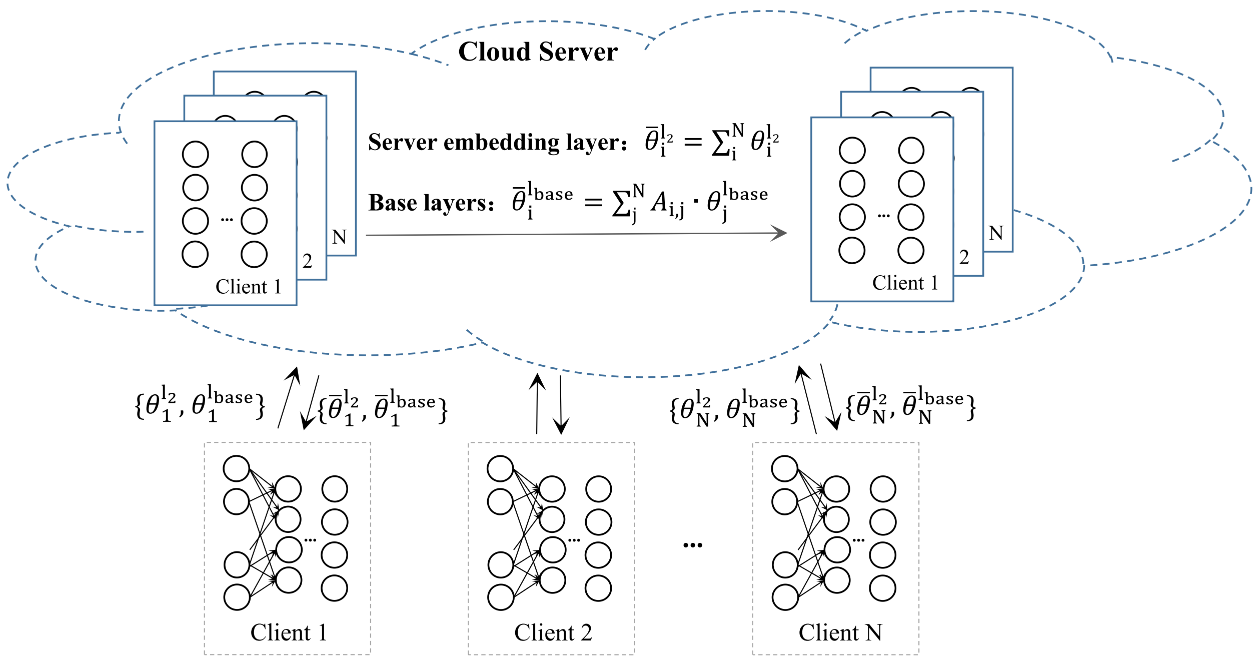

Cloud Server
Server embedding layer: $\bar{\theta}_i^{l_2} = \sum_i^N \theta_i^{l_2}$
Base layers: $\bar{\theta}_i^{l_{base}} = \sum_j^N A_{i,j} \cdot \theta_j^{l_{base}}$
Client 1
2
N
$\{\theta_1^{l_2}, \theta_1^{l_{base}}\}$
$\{\bar{\theta}_1^{l_2}, \bar{\theta}_1^{l_{base}}\}$
$\{\theta_N^{l_2}, \theta_N^{l_{base}}\}$
$\{\bar{\theta}_N^{l_2}, \bar{\theta}_N^{l_{base}}\}$
Client 1
Client 2
Client N

FIGURE 2. The framework of pFedLN approach for personalization federated learning. Clients only upload the parameters of service embedding layer θ^{l_2} and the parameter of base layer $\theta^{l_{base}}$ to the cloud server for collaborative training. On the cloud server, θ^{l_2} and $\theta^{l_{base}}$ will be aggregated by different strategies. Then the aggregated results will be sent to the corresponding clients.

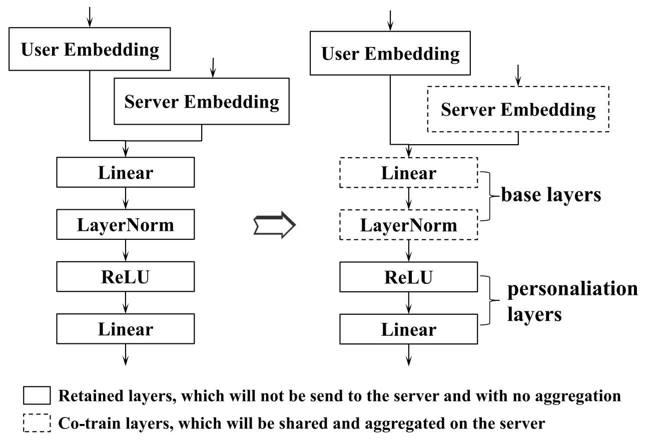

User Embedding
Server Embedding
Linear
LayerNorm
ReLU
Linear
User Embedding
Server Embedding
Linear
LayerNorm
ReLU
Linear
base layers
personaliation layers
Retained layers, which will not be send to the server and with no aggregation
Co-train layers, which will be shared and aggregated on the server

FIGURE 3. Illustration of the introduction of neural network layers in pFedLN and the functionally division of these layers.

at the QoS prediction neural network model, considering the privacy and functional differences of each layer in it.

We categorize the layers of the neural network model into four segments as depicted in the Figure 3: the user embedding layer, the service embedding layer, the base layers, and the personalization layers. This categorization is based on the depth and the distinct functionality of model layers. More specifically, we designate the two input layers as user embedding layer and service embedding layer, layers 3 through k-1 of the network as the base layer, with layers k through the final output layer categorized as the personalization layer. This novel division offers the potential for more efficient training and improved personalization performance within the context of federated learning.

For these four types of layers, we use different aggregation strategies:

(1) The user embedding layer is only allowed to be trained and stored locally. To ensure the security of users' local data, we exclusively conduct local training and storage for this layer, which may potentially contain user privacy. And the user embedding layer remains uninvolved in collaborative training.
(2) After the service embedding layer is trained locally, it will be uploaded to the server for average aggregation, and then the aggregation result will be sent to each client for local update. The service embedding layer encompasses global service information, while each client possesses only a limited dataset. Therefore, the server-side collaborative training for the service embedding layer is necessary, effectively enhancing the model's predictive capabilities.
(3) The base layers will be uploaded by the client to the server for neighbor-based aggregation, and then sent to each client for local update.
(4) The personalization layers are only locally trained and not uploaded for aggregation because its functional characteristics are conducive to personalization and have less impact on global convergence.

We define the model produced by client i at the t-th local training iteration as Θ_i^t:

$$\Theta_i^t = \{\theta_i^{l_1,t}, \theta_i^{l_2,t}, \theta_i^{l_{base},t}, \theta_i^{l_{per},t}\}, \tag{3}$$

where θ denotes the parameter of a layer of the model, $\theta^{l_1,t}$ denotes the parameter of the 1-st layer of client i at the t-th iteration, and similarly $\theta^{l_2,t}$, $\theta^{l_{base},t}$, $\theta^{l_{per},t}$ denote the parameter of the 2-nd layer, the base layers, and the personalization layers, respectively. Here, $\theta^{l_1,t}$ and $\theta^{l_{per},t}$ represent the user embedding layer and the personalization layer, respectively, both of which are only updated locally, not shared, and not aggregated. $\theta^{l_2,t}$ and $\theta^{l_{base},t}$ represent the service embedding layer and the base layers, respectively, which can be shared and co-trained by multiple clients, but they are aggregated in different ways.

The service embedding layer uses gradient averaging aggregation [7]:

$$\bar{\theta}^{l_2,t} = \bar{\theta}^{l_2,t-1} - \frac{\sum_{i=1}^{N}\left(\bar{\theta}^{l_2,t-1} - \theta_i^{l_2,t}\right)}{N}, \tag{4}$$

where $\theta^{l_2,t}$ represents the local service embedding layer parameters at the t-th iteration, and $\bar{\theta}^{l_2,t}$ denotes the update layer parameters at the $(t-1)$-th iteration. Here $\bar{\theta}^{l_2,0}$ is initialized as $\theta^{l_2,0}$. N represents the number of clients participating in collaborative training, $\theta_i^{l_2,t}$ denotes the parameters of the i-th client at the t-th iteration. The gradient at t-th iteration is the difference between $\theta_i^{l_2,t}$ and $\bar{\theta}^{l_2,t}$. The aggregation results are distributed to all clients, and each client updates these results locally before proceeding with the next training iteration.

The base layers utilize similarity neighbor models for average aggregation. We adopt a matrix A to represent the similarity between clients. A is a $N * N$ matrix and N represents the number of clients in the system. The element $A_{i,j}$ indicates the similarity between client i and client j.

In collaborative training, contextual information is commonly employed to determine the similarity between clients. For services with similar functionalities, the quality of service data may be influenced by various factors, including geographical location and network conditions. Among the contextual information available, clients' geographical details, such as their region, can be particularly informative. Clients sharing similar geographic locations are likely to exhibit similarity in QoS values. However, due to privacy concerns, we only collect a subset of contextual information, such as the country. This partial context information serves as the foundation for calculating similarity. Using this context information, we compute a similarity matrix, denoted as matrix A where $A_{i,j} \in \{0, 1\}$, to quantify the similarity between clients. When the context information (e.g., the country of residence) is the same, the value of $A_{i,j}$ is 1, and vice versa, $A_{i,j}$ is 0. After establishing the 0-1 similarity matrix, we perform a normalization transformation on it to obtain the final similarity matrix.

After completing the construction of the matrix A, we use A to perform neighbor-based aggregation on the base layers:

$$\bar{\theta}_i^{l_{base},t} = \sum_{j=1}^{N} \theta_j^{l_{base},t} \cdot A_{i,j}, \tag{5}$$

where $\theta^{l_{base}}$ denotes the base layer parameters, $\theta_i^{l_{base}}$ denotes the parameters of the i-th client aggregated by the server at the t-th iteration, N denotes the number of clients participating in the aggregation, $\theta_j^{l_{base}}$ denotes the parameters of neighbor j at the t-th iteration result, and $A_{i,j}$ denotes the similarity between client i and client j. The server performs neighbor aggregation for each client separately and generates the corresponding parameters for each client. Distinguishing from the traditional approach where all users share the same global model, our approach generates N different sets of parameters corresponding to N clients after the base layer aggregation, which better achieves personalized federated learning training.

The local model parameter Θ_i^t is updated as:

$$\Theta_i^t = \{\theta_i^{l_1,t}, \bar{\theta}_i^{l_2,t}, \bar{\theta}_i^{l_{base},t}, \theta_i^{l_{per},t}\}, \tag{6}$$

where $\theta_i^{l_1,t}$ and $\theta_i^{l_{per},t}$ are still the results of the local training, while $\bar{\theta}_i^{l_2,t}$ and $\bar{\theta}_i^{l_{base},t}$ are the aggregated results. When the model reaches convergence, the predicted value is computed as follow:

$$r'_{i,j} = F\left(u_i, s_j \mid \Theta_i^t\right), \tag{7}$$

where u_i,s_j denote client i and service j, respectively, and $F(\cdot)$ denotes the neural network model function, u_i and s_j are two inputs of $F(\cdot)$, Θ_i^t denotes the model parameters, and $r'_{i,j}$ is the model output.

Algorithm 1 demonstrates the pFedLN procedure. The whole process consists of three parts: initialization, client execution and server execution. In the initialization phase, each client adopts the initial model parameters provided by the server; the server computes the similarity by the context information from users, and constructs an adjacency matrix using the *GetNeighbor()* function. The main idea of *GetNeighbor()* is to compute the similarity based on the context information, so as to generate the adjacency matrix A. During the client execution phase, the client carries out local training to update its model parameters. Equation (1) is the objective function for model optimization. Upon completion of the local training, each client then transfers the parameters of the service embedding layer and the base layer to the server. In the server execution phase, the server collects parameters uploaded by clients and subsequently aggregates the parameters of both the service embedding layer and the base layers. Equation (4) represents the aggregation method for the service embedding layer parameters, while (5) is the aggregation method for the base layer parameters. The aggregated results comprise two parts: one is a set of global service embedding layer parameters, and the other is a collection of N sets of base layer parameters, with each of these N sets corresponding respectively to N different clients.

After receiving the aggregated results, each client subsequently updates its local model. The next round of training will also be initiated after the update, during which both the client and server-side programs will continue to iterate until the model converges.

Algorithm 1 pFedLN Algorithm

```
Input:
    dataset $\{D_1, D_2, \dots, D_N\}$, learning rate $\eta$. Total communication rounds $T$.
Output: Trained personalized models $\{\theta_1, \theta_2, \dots, \theta_N\}$.
 1: Initialize local model parameters: $\Theta_1^0, \Theta_2^0, \dots, \Theta_N^0$.
 2: Initialize adjacency matrix $A$: GetNeighbor().
 3: for each communication round $t \in \{1, \dots, T\}$ do
      Procedure Client Executes:
 4:   model training: $F(u_i, s_j|\Theta_i^{t-1})$
 5:   local parameters: $\Theta_i^t : \{\theta_i^{l_1,t} \theta_i^{l_2,t}, \theta_i^{l_{base},t}, \theta_i^{l_{per},t}\}$
 6:   parameters send to server: $\theta_i^{l_2,t}, \theta_i^{l_{base},t}$
      Procedure Server Executes:
 7:   initialize the sum of the parameters of the service embedding layer: $sum^{l_2}$
 8:   for each client $i$ do
 9:     $sum^{l_2} += \theta_i^{l_2,t}$
10:   end for
11:   $\bar{\theta}_i^{l_2,t} = \bar{\theta}_i^{l_2,t-1} - \left(\frac{sum^{l_2}}{N} - \theta_i^{l_2,t}\right)$
12:   for each client $i$ do
13:     initialize $\bar{\theta}_i^{l_{base},t}$
14:     for each client $j \in N$ do
15:       $\bar{\theta}_i^{l_{base},t} += \theta_j^{l_{base},t} \cdot A_{i,j}$
16:     end for
17:   end for
      Procedure Client Executes:
18:   receive paramters from server and update model:
      $\Theta_i^t \leftarrow \{\theta_i^{l_1,t} \bar{\theta}_i^{l_2,t}, \bar{\theta}_i^{l_{base},t}, \theta_i^{l_{per},t}\}$
19: end for
    GetNeighbor():
20: initialize $A = [0 * N] * N$
21: for each client $i$ do
22:   for each client $j$ do
23:     if the context of client $i$ is equal to that of client $j$ then
24:       $A_{i,j} = 1$
25:     else
26:       $A_{i,j} = 0$
27:     end if
28:     $A \leftarrow Normalization\_transformation(A)$
29:   end for
30: end for
31: return A
```

IV. EXPERIMENT

In this section, sufficient experiments are conducted in a real-world dataset to answer the following research questions:

RQ1: Dose our proposed approach perform more effectively in QoS prediction based on federated learning than existing methods?
RQ2: What is the effect of layer-wised and neighbor-based aggregation on prediction accuracy?
RQ3: How does our method perform on convergence speed?

In the following subsections, we will conduct experiments based on the aforementioned three research questions and delve into a comprehensive discussion of these three issues.

A. EXPERIMENT SETTINGS

1) DATASET AND PARAMETER SETTINGS

To evaluate our approach, we conducted our experiment on a real-world QoS dataset, WS-DREAM [20]. This dataset includes 1,974,675 real-world Web service invocations conducted by 339 service users from 30 countries on 5,825 actual web services in 73 countries. Here is a set of default settings for our experiments: 1) *batch_size*: the number of samples at each training iteration, was set to 256. 2) *iteration*: the number of iterations of the model, default is 500. 3) *client_epochs*: the number of iterations of local training, default is 1. 4) *learning rate*: a rate controls the model weights updated during training, is set to 0.01. For each experiment, we tested our proposed model five times at each density and took the average of the results as the final outcome.

2) EVALUATION METRIC

In order to measure the performance of our model, we use Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) as the predictive accuracy metrics.

- **MAE**: It measures the average absolute difference between the predicted and the actual values of a dataset. The formula for calculating MAE is:

$$MAE = \frac{\sum_{(i,j,r_{i,j})\in O} |\hat{r}_{i,j} - r_{i,j}|}{|O|}, \tag{8}$$

- **RMSE**: It measures the square root of the average of the squared differences between the predicted and the actual values. The formula for calculating RMSE is:

$$RMSE = \sqrt{\frac{\sum_{(i,j,r_{i,j})\in O} |\hat{r}_{i,j} - r_{i,j}|^2}{|O|}}, \tag{9}$$

where O denotes observed records in the training dataset, |O| denotes the number of records in O, $r_{i,j}$ denotes the actual QoS value of server j invoked by user i, while $\hat{r}_{i,j}$ denotes the predicted value. Both MAE and RMSE are measures of the error between predicted and actual values, where a lower value of either metric indicates higher prediction accuracy.

B. PREDICTION ACCURACY COMPARISON (RQ1)

To verify the prediction accuracy of our approach, we compare it via the following existing personalization federated learning methods:

- FedAvg [7]. FedAvg is an aggregation method for federated learning, which operates by averaging the parameters across all participating clients. In this methodology, each client uploads its individual parameters, and all users collaboratively train a unified global model.
- FedAtt [21]. A method for introducing attention mechanisms in federated learning involves aggregating client

TABLE 2. Comparison of response-time prediction MAE and RMSE among baseline approaches and pFedLN.

Approach	Density=0.50%		Density=1.00%		Density=1.50%		Density=2.00%	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
FedAvg	0.6695	1.5411	0.584	**1.3176**	0.6118	1.4993	0.5812	**1.3418**
FedAtt	0.6648	1.5426	0.6087	1.5644	0.6035	1.5669	0.5952	1.5444
FedPer	**0.6128**	**1.482**	**0.558**	1.5127	**0.5245**	**1.3705**	**0.5288**	1.3549
pFedLN	**0**.4751	**1.171**	**0.3964**	**1.114**	**0.3761**	**1.071**	**0.3682**	**1.0371**
%Improv. (vs. FedPer)	22.47%	20.99%	28.96%	26.36%	28.29%	21.85%	30.37%	23.46%

parameters according to attention weights. In this method, clients upload all their parameter information and collaboratively train a unified global model.

- FedPer [12]. A personalized federated learning aggregation method that directly divides the neural network layers into the base layers and the personalized layers, where the base layers are globally averaged and updated, while the personalized layers are only updated locally. In this method, users only upload the parameters of the base layer and all users train a unified global model together.

In the IoS environment, obtaining paired QoS values is a challenging task, leading to a highly sparse collection of QoS data [3]. To accurately simulate a QoS scenario with high sparsity, we set the density of the training set to 0.5%, 1%, 1.5%, and 2%. Table 2 provides comparative prediction MAE and RMSE results on RT in WS-DREAM. We have the following observations:

1) The experimental results show that FedPer and our proposed pFedLN method have better effectiveness than FedAvg and FedAtt. This improved performance can be attributed to the consideration of functional differences between neural network layers. In both the FedPer and pFedLN, attention is given to the functional disparities among neural network layers, and the layers are divided based on the disparities. As a result, in the case of uploading only some of the parameters for co-training, it achieves improving the effect of personalized training.
2) Among the baseline methods, we found that FedPer has the best results overall. Comparing with FedPer, our method has superior performance. The MAE criterion of our method is decreased by 22.47% to 30.37%, and the RMSE criterion is decreased by 20.99% to 26.36%. This demonstrates the enhanced efficiency of our approach in personalized prediction. Importantly, our method consistently outperforms these baseline techniques, even in scenarios with sparse training data, underscoring its robustness in such data-scarce conditions.

C. EFFECT OF LAYER-WISED AND NEIGHBOR-BASED AGGREGATION (RQ2)

In this section, we will discuss the effects of layer-based aggregation and neighbor-based aggregation in our model.

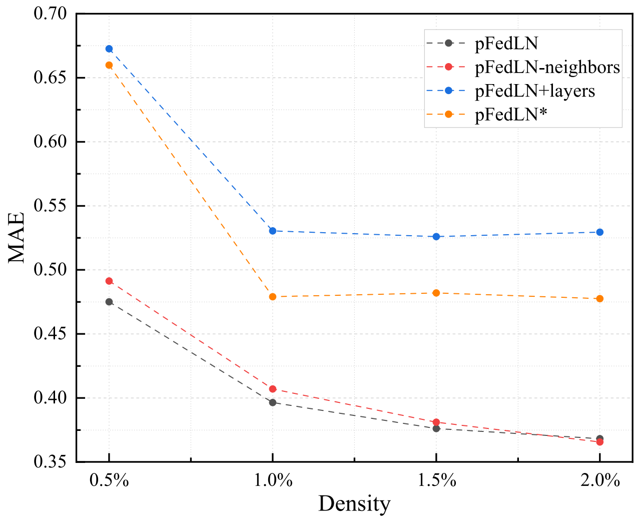


FIGURE 4. Comparison of the models' performance at different densities, evaluated by the criterion MAE.

We conducted controlled ablative experiments using pFedLN to explore the effects of specific functionalities:

- pFedLN-neighbors: User similarity is not considered in client-side aggregation in this method.
- pFedLN+layers: Rather than segmenting the neural network layers, all layers are aggregated in this approach. The user embedding layer and service embedding layer are aggregated using gradient average aggregation, while the remaining layers are aggregated through simple average aggregation.
- pFedLN*: In this method, the same aggregation method employed for the base layer is also applied to the service embedding layer.

The MAE and RMSE results are shown in Figure 4 and Figure 5 respectively. We have the following observations:

- Comparing pFedLN-neighbors to pFedLN, pFedLN outperforms, enhancing the effectiveness by 1.04% overall. This suggests that considering neighbor relationships and assigning aggregation weights to neighbors based on similarity can enhance the model's performance.
- When comparing pFedLN+layers to pFedLN, pFedLN significantly outperforms pFedLN with an overall

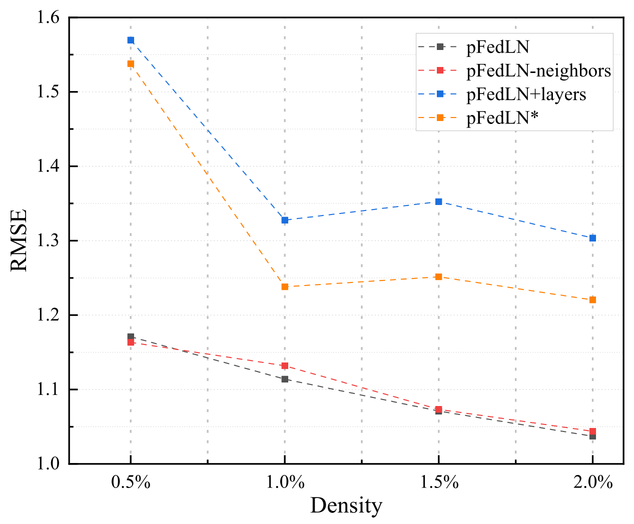


FIGURE 5. Comparison of the models' performance at different densities, evaluated by the criterion RMSE.

improvement of 24.54%. Experimental results indicate that the performance of all layers participating in collaborative training does not improve with an increase in parameter size. In contrast, pFedLN, which only utilizes a subset of layer parameters for collaborative training, performs better. This achievement can be attributed to the consideration of functional differences between various neural network layers. Specifically, we adopted a unique aggregation method for shallower network layers, enabling the capture of fundamental and general features. For deeper network layers focused on task-specific and personalized feature learning, we perform updates only locally. Notably, for user embedding layers containing user information and potential privacy concerns, we strictly conduct updates locally to enhance user privacy protection in the model.

- Comparing pFedLN* and pFedLN, pFedLN outperforms pFedLN* with an overall improvement of 19.18%. The design of pFedLN* is aimed at investigating whether using gradient updates or employing a similar neighbor aggregation approach for the service embedding layer leads to better model performance. Experiments have demonstrated that applying a specific aggregation approach to the service embedding layer is more conducive to improving the model's effectiveness. This idea is inspired by the practice of gradient updating for latent factors in federated matrix factorization [5], where collaborative training overcomes the limitations of local service feature quantities, enabling access to global service features.

In summary, the development of distinct aggregation methods that consider the layer-to-layer disparities in neural networks and rely on client similarity proves highly advantageous for personalized aggregation. This grants our approach a distinct advantage in enhancing the efficacy of personalization.

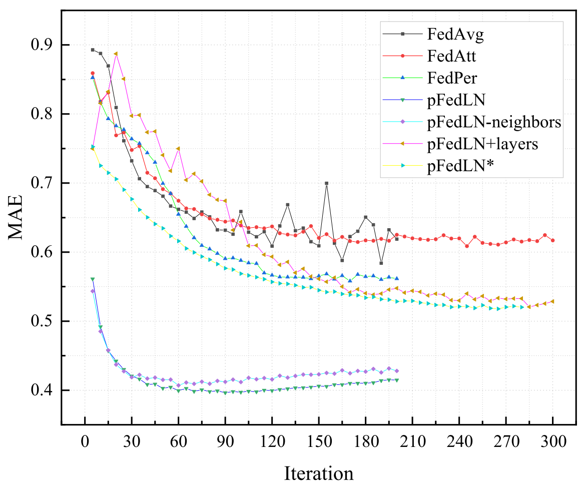


FIGURE 6. The MAE value of models as the iteration proceeds.

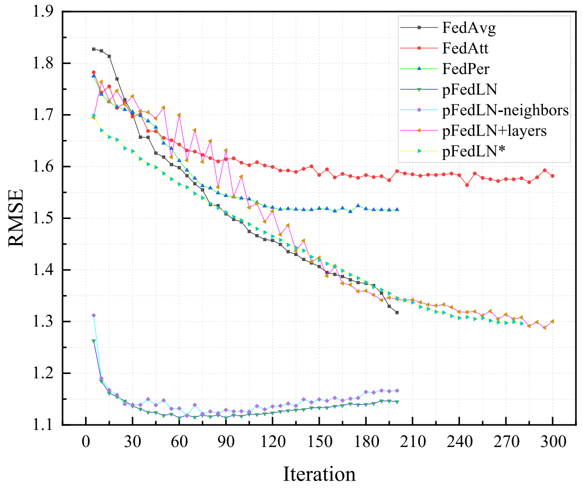


FIGURE 7. The RMSE value of models as the iteration proceeds.

D. CONVERGENCE SPEED COMPARISON(RQ3)

In this subsection, we conduct a comparison of the convergence times required for each method. We have configured the maximum number of iterations to 300 and established a training matrix density of 1%. The learning rates for the different methods have been set to their optimal values. We document the number of iterations necessary for each method to achieve convergence. The MAE and RMSE results are shown in Figure 6 and Figure 7 respectively. We have the following findings:

- The pFedLN method demonstrated the shortest convergence time, becoming close to convergence by the 60th iteration. In comparison, among the baseline methods, FedPer was approaching convergence at the 120th iteration, while FedAtt did not come close until the 245th iteration. Simultaneously, FedAvg reached optimal MAE value around the 90th iteration, subsequently experiencing some fluctuations. When comparing pFedLN's performance with the three baseline methods - FedAvg, FedAtt, and FedPer - it's clear that pFedLN has significant advantages, taking 33.33%, 75.51%, and 50.00% less time to converge, respectively. These findings illustrate another advantage of our method: pFedLN not only bolsters personalization effectiveness but also accelerates model convergence.

- Among the ablation methods, pFedLN-neighbors was approaching convergence at the 85th iteration, pFedLN+layers was at the 235th iteration, and pFedLN* was at the 240th iteration. Comparing to pFedLN-neighbors, pFedLN+layers, and pFedLN*, the convergence time of FedPer is reduced by 29.41%, 74.47%, and 75.00%, respectively.
- We observed that the MAE and RMSE for the pFedLN and pFedLN-neighbors schemes bounce back slightly after continuously decreasing for around 100 iterations. One possible reason is that when the training iterations exceed the optimal number, the learning rate or other model parameters go beyond their adaptive range. In practice, the training only needs to meet the convergence requirement to end, and does not require as many iterations.

  This demonstrates that the method of aggregating the service embedding layer and the base layer in different ways is very effective, significantly improving the convergence speed of the model. Furthermore, implementing the neighbor-based aggregation can also contribute to the acceleration of the model's convergence rate to a certain degree.

In a real-world scenario, the time required for model training serves as a critical criterion for assessing the practicality of a model. It has been demonstrated that our method effectively enhances the model's convergence speed while ensuring higher-quality personalized predictions.

V. CONCLUSION AND FUTURE WORK

In this paper, we focus on enhancing pFL QoS prediction in IoS scenarios. Our proposed scheme has several advantages over existing QoS prediction methods, which include: (1) Highly accurate personalization prediction: Our approach considers the privacy and functional differences between neural network layers, and makes a classification of these layers. We develop different aggregation strategies for the co-trained service embedding layer and the base layers. In addition, the user embedding layer which may contain user privacy, and the personalization layer which is more effective for personalized feature mining, are only trained and updated locally. Our scheme greatly improves the effectiveness of pFL model. The experimental results indicated that comparing the best performing baseline method, pFedLN reduced the MAE values by 22.47% to 30.37% and the RMSE values by 20.99% to 26.36%. (2) Considering neighbor similarities for aggregation: For clients participating in collaborative training, we do not perform aggregation in general, but select similar clients for weighted aggregation, which effectively improves the efficiency of collaborative training. (3) Enhancing model convergence speed. In pFedLN, the clients involved in co-training only need to upload part of the model parameters, and the server only aggregates this part of parameters, which greatly reduces the time consumption for training. Experimental results show that our method improves at least 67.54% in convergence speed over the baseline method.

In our future work, we will continue to explore more effective and adaptable personalization methods. In pFedLN, the classification of the layers is designed simply by the results of the experiment. But for more complex and flexible models, a more flexible classification strategy is needed. What's more, our method is now only experimented on offline data. We will further consider personalized federated learning in mobile edge environment, where the client's contextual information will change in real time. It is also necessary to consider how the client's similarity should be computed to adapt to the characteristics of the dynamic environment. Moreover, the model will further take into account the analysis of the online data, and enhance its scalability and robustness to better suit real-world applications. This will ensure that our model is more practical and effective in real-world environments.

REFERENCES

[1] H. I. Al-Salman and M. H. Salih, "A review cyber of industry 4.0 (cyber-physical systems (CPS), the Internet of Things (IoT) and the Internet of Services (IoS)): Components, and security challenges," *J. Phys., Conf. Ser.*, vol. 1424, no. 1, Dec. 2019, Art. no. 012029.

[2] X. Xu, Q. Z. Sheng, L.-J. Zhang, Y. Fan, and S. Dustdar, "From big data to big service," *Computer*, vol. 48, no. 7, pp. 80–83, Jul. 2015.

[3] J. Xu, Z. Xia, Y. Li, Y. Zeng, and Z. Liu, "Subgraph sampling for inductive sparse cloud services QoS prediction," in *Proc. IEEE 28th Int. Conf. Parallel Distrib. Syst. (ICPADS)*, Jan. 2023, pp. 745–753.

[4] Z. Zheng, X. Li, M. Tang, F. Xie, and M. R. Lyu, "Web service QoS prediction via collaborative filtering: A survey," *IEEE Trans. Services Comput.*, vol. 15, no. 4, pp. 2455–2472, Jul. 2022.

[5] D. Chai, L. Wang, K. Chen, and Q. Yang, "Secure federated matrix factorization," *IEEE Intell. Syst.*, vol. 36, no. 5, pp. 11–20, Sep. 2021.

[6] H. Wu, Z. Zhang, J. Luo, K. Yue, and C.-H. Hsu, "Multiple attributes QoS prediction via deep neural model with contexts," *IEEE Trans. Services Comput.*, vol. 14, no. 4, pp. 1084–1096, Jul. 2021.

[7] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.* Ft. Lauderdale, FL, USA: Proceedings of Machine Learning Research (PMLR) 2017, pp. 1273–1282.

[8] X. Ma, J. Zhang, S. Guo, and W. Xu, "Layer-wised model aggregation for personalized federated learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 10082–10091.

[9] J. Xu, J. Lin, Y. Li, and Z. Xu, "MultiFed: A fast converging federated learning framework for services QoS prediction via cloud–edge collaboration mechanism," *Knowl.-Based Syst.*, vol. 268, May 2023, Art. no. 110463. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705123002137

[10] S. Lee, T. Zhang, and A. S. Avestimehr, "Layer-wise adaptive model aggregation for scalable federated learning," in *Proc. AAAI Conf. Artif. Intell.*, 2023, vol. 37, no. 7, pp. 8491–8499.

[11] M. Duan, D. Liu, X. Chen, R. Liu, Y. Tan, and L. Liang, "Self-balancing federated learning with global imbalanced data in mobile systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 1, pp. 59–71, Jan. 2021.

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

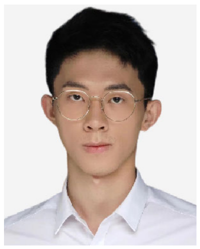
[13] J. Schneider and M. Vlachos, "Personalization of deep learning," in *Proc. 3rd Int. Data Sci. Conf.* Cham, Switzerland: Springer, 2021, pp. 89–96.

[14] C. T. Dinh, N. Tran, and J. Nguyen, "Personalized federated learning with Moreau envelopes," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 33, 2020, pp. 21394–21405.
[15] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," in *Proc. NeurIPS Conf.*, vol. 33, Dec. 2020, pp. 3557–3568.
[16] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai, "Exploiting shared representations for personalized federated learning," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 2089–2099.
[17] Y. Huang, L. Chu, Z. Zhou, L. Wang, J. Liu, J. Pei, and Y. Zhang, "Personalized cross-silo federated learning on non-IID data," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 9, pp. 7865–7873.
[18] F. Chen, G. Long, Z. Wu, T. Zhou, and J. Jiang, "Personalized federated learning with graph," 2022, *arXiv:2203.00829*.
[19] D. Kingma and L. Ba, "Adam: A method for stochastic optimization," in *Proc. 2015 Int. Conf. Learn. Representations (ICLR)*. Ithaca, NY, USA, 2015, p. 13.
[20] Z. Zheng, Y. Zhang, and M. R. Lyu, "Investigating QoS of real-world web services," *IEEE Trans. Services Comput.*, vol. 7, no. 1, pp. 32–39, Jan. 2014.
[21] S. Ji, S. Pan, G. Long, X. Li, J. Jiang, and Z. Huang, "Learning private neural language modeling with attentive aggregation," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–8.

WEIWEI SHE is currently a Graduate Student with the Department of Computer Science and Technology, Shantou University, China. Her research interests include service computing, privacy protection, and artificial intelligence.

JIANLONG XU (Member, IEEE) received the Ph.D. degree from the South China University of Technology, in 2013. He was a Postdoctoral Fellow with the Shenzhen Research Institute, The Chinese University of Hong Kong. He is currently a Lecturer with the Department of Computer Science and Technology, Shantou University, China. His research interests include service computing, machine learning, and information security.

MENGQING JIN is currently a Graduate Student with the Department of Computer Science and Technology, Shantou University, China. Her research interests include service computing, artificial intelligence, and information security.



YUELONG LIU is currently a Graduate Student with the Department of Computer Science and Technology, Shantou University, China. His research interests include service computing, the Internet of Things, and information security.