Edge-Cloud Enabled Smart Sensing Applications with Personalized Federated Learning in IoT

Yingchi Mao[†], Yi Rong[†], Jiajun Wang[†], Zibo Wang[†], Xiaoming He[‡], and Jie Wu[§]
[†]College of Computer Science and Software Engineering, Hohai University, Nanjing, China.
[‡]College of Internet of Things, Nanjing University of Posts and Telecommunication, Nanjing, China.
[§]Center for Networked Computing, Temple University, Philadelphia, USA
Email: [†]yingchimao@hhu.edu.cn, [†]rongyi1220@163.com, [†]211307040017@hhu.edu.cn, [†]221307040022@hhu.edu.cn,
[‡]hexiaoming@njupt.edu.cn, [§]jiewu@temple.edu

***Abstract*—The rapid development of deep learning technologies and the widespread deployment of sensing devices have brought considerable attention to Internet of Things (IoT). The smart sensing application is one of the popular applications in IoT. Personalized Federated Learning (pFL) is a replacement to traditional Federated Learning (FL) to tackle the statistical heterogeneity of clients' private datasets (*e.g.*, non-Independent and Identically (non-IID) data). However, existing pFL methods encounter two challenges in smart sensing applications: a) the global model preference, causing poor global model performance for minority classes on sensing device data. b) the dynamic role differences in each hierarchy of the layer-stacked deep learning model that needs to be considered. Jointly considering these challenges, we present a novel edge-cloud enabled pFL framework named *pFL-Sensing* for smart sensing applications. Specifically, the sensing device serves as an edge server. Each edge server produces a customized model through two phases: a *model training* phase and a *model aggregation* phase. In the model training phase, we design a novel loss function to alleviate the issue of the global model preference. In the model aggregation phase, hierarchical aggregation and an Adaptive Weight Calculation (AWC) mechanism are proposed to capture the dynamic role differences of model hierarchies. We perform simulation experiments on real image and text classification tasks. Experimental results show that pFL-Sensing demonstrates higher classification accuracy than advanced pFL baselines.**

***Index Terms*—pFL, edge-cloud, smart sensing applications, IoT**

I. Introduction

With the rapid proliferation of the Internet of Things (IoT), various sensing devices (*e.g.*, monitoring stations, cameras, and smartphones) have been extensively deployed [1], [2]. These devices produce large amount of sensor data, which can be analyzed and mined to serve IoT applications. The emergence of deep learning technology provides endless possibilities for extracting valuable information from massive data [3]. Some popular deep learning models, such as, Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN), show excellent performance on various smart sensing applications like prediction and classification [4], [5]. These models typically employ centralized learning to execute training tasks, *i.e.*, data from sensing devices is transmitted to a distant cloud server for training. Unfortunately, as the number of sensing devices increases, centralized training becomes challenging due to the limitations of computational load and network bandwidth.

To address the issues of centralized learning, McMahan *et al*. proposed Federated Learning (FL), a distributed learning architecture [6]. The key point of FL is to combine all local models deployed on users that we call clients to implement a more generalized global model. FL is a standard edge-cloud collaborative architecture. Since FL does not differentiate between clients, its effectiveness heavily relies on the data distribution of clients. FL performs perfectly on Independent and Identical Distribution (IID) data, *i.e.*, all clients have similar data distributions. However, in smart sensing applications, the data from each sensing device (client) obeys a non-IID with statistical heterogeneity, *i.e.*, the data distributions of sensing devices vary significantly due to differences in geographical distribution. FL that does not consider individual private data fails to individual sensing devices.

In order to cope with non-IID data, several studies have developed personalized Federated Learning (pFL) methods, which learn customized models corresponding to individual clients [7]–[13]. The customized models adapt to their corresponding client's data distribution. One common type of pFL is weighted model aggregation-based pFL, which consists of different types of approaches [14]: 1) ones that train a global model and fine-tune it, including the method Per-FedAvg [7], 2) ones that train an additional customized model for each client, including Ditto [8] and pFedMe [9], and 3) ones that train customized models with personalized aggregation, including FedPHP [10], FedAMP [11], FedFomo [12], and PartialFed [13]. Weighted model aggregation-based pFLs in Categories (1) and (2) employ knowledges of the global model to initialize local models before each round of iteration training. Unfortunately, since the global model contains both useful and useless information for each individual client, clients only benefit from partial information, and the model has poor generalization performance. In light of these shortcomings, Category (3) approaches aim to learn the useful information in the global model using personalized aggregation. Despite further improving the performance of the model on non-IID data, these approaches are difficult to effectively implement smart sensing applications due to two challenges:

The effect of the global model preference: In real application scenarios, sensing device data may contain classes that occur relatively infrequently. In this case, the global model

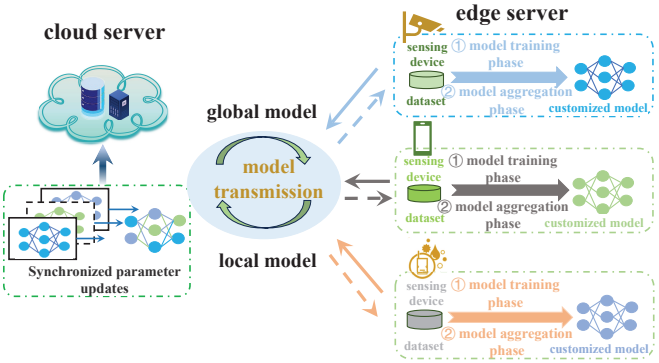

Fig. 1. Edge-Cloud Enabled pFL framework for smart sensing applications. The edge server leverages sensing device data to learn a customized model and communicate with the cloud server at consistent intervals. In each communication round, the edge server adopts a two-phase strategy to acquire the customised model after downloading the global model. In the cloud server, the parameters of the global model are updated by synchronously aggregating the gradients of the local models on each edge server.

training primarily targets the majority classes and exhibits poor performance on the minority classes on the sensing device. We define this phenomenon as the *global model preference*. The global model preference weakens the performance of customized models in pFL. Existing pFLs based on weighted model aggregation have not succeeded in reducing the adverse effect of the global model preference [15].

Dynamic role differences of model hierarchy: In a deep learning model, different hierarchies play different roles. Given a Convolutional Neural Network (CNN) model involving two Convolutional (Conv) hierarchies and two Fully Connected (FC) hierarchies, the initial two Conv hierarchies focus primarily on feature extraction, while the latter FC hierarchies are responsible for integrating features and making decisions. Additionally, the role of each hierarchy in the model is dynamic and continuous changes in training round iterations. Most weighted model aggregation-based pFLs treat the model as a whole when aggregating, neglecting the different roles of each hierarchy in the model [15], [16]. Few studies have investigated hierarchical aggregation, but there is a lack of adaptive aggregation weights to capture the dynamic changes of hierarchical roles, leading to inaccurate personalization [17].

To address the above challenges, We present a novel edge-cloud enabled weighted model aggregation-based pFL framework, named *pFL-Sensing*, to effectively implement smart sensing applications. The main idea of pFL-Sensing is displayed in Fig. 1. The sensing device serves as an edge server. Each edge server generates a customized model through two phases-a *model training* phase and a *model aggregation* phase. In the model training phase, we introduce Inverse Category Frequency (ICF) [18] into the loss function when training the local model. ICF increases the proportion of loss from minority class samples to the total loss, thus alleviating the adverse effect of the global model preference. In the model aggregation phase, hierarchical aggregation is first used to implement hierarchy-level aggregation with aggregation weights. The Adaptive Weight Calculation (AWC) mechanism is then used to adaptively measure and update aggregation weights based on the dynamic role differences of model hierarchy. The primary contributions of this work are summarized as follows:

- We present a novel edge-cloud enabled weighted model aggregation-based pFL framework named pFL-Sensing for smart sensing applications. To mitigate the global model preference, in each sensing device, a novel model training strategy is designed to train a local model. Specifically, an ICF is introduced into the loss function for training the local model, raising the proportion of loss from minority class samples in the total loss.
- To capture dynamic role differences in model hierarchy, in each sensing device, we design a novel model aggregation strategy with hierarchical aggregation and an AWC mechanism. The former enables a layer-wised aggregation of the local model and the customized model for each client based on aggregation weights. The latter aims to adaptively update the aggregation weights in accordance with the dynamic role differences in model hierarchy.
- We conduct simulation experiments on three public and representative datasets (*e.g.*, CIFAR-100, Tiny-ImageNet, and AG News), which cover both image and text classification tasks. Experimental results prove that our proposed model achieves superior performance compared to advanced pFL baselines in classification accuracy.

II. pFL-Sensing Design

In this section, we illustrate the design of pFL-Sensing. We first provide the problem formulation of smart sensing applications. Second, a system overview of pFL-Sensing is introduced. Third, we introduce the detailed design of pFL-Sensing.

A. Problem Formulation

In the work, our objective is to train customized models on edge servers with heterogeneous sensing device data for a smart sensing application. It is worth noting that an edge server is a sensing device. We select a deep learning model as the model to be trained. The smart sensing application is identified as a classification task.

Specifically, Given N sensing devices with private classification task datasets $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_N$, respectively. These datasets are non-IID with statistical heterogeneity. Let $\mathcal{D}_i = \{(x_{i,b}, y_{i,b})\}_{b=1}^{m_i}, i \in [1, N]$ denote the dataset of the i^{th} sensing device, where $x_{i,b}$ and $y_{i,b}$ stand for the b^{th} sample and the corresponding label in the i^{th} sensing device's dataset. m_i is the total number of samples in the i^{th} sensing device's dataset. The total size of all sensing device datasets is $M = \sum_{i=1}^{N} m_i$. We collaboratively train customized models $\bar{v}_1, \bar{v}_2, ..., \bar{v}_N$ through $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_N$, where $\bar{v}_i$ is a customized model for the i^{th} sensing device. The optimal

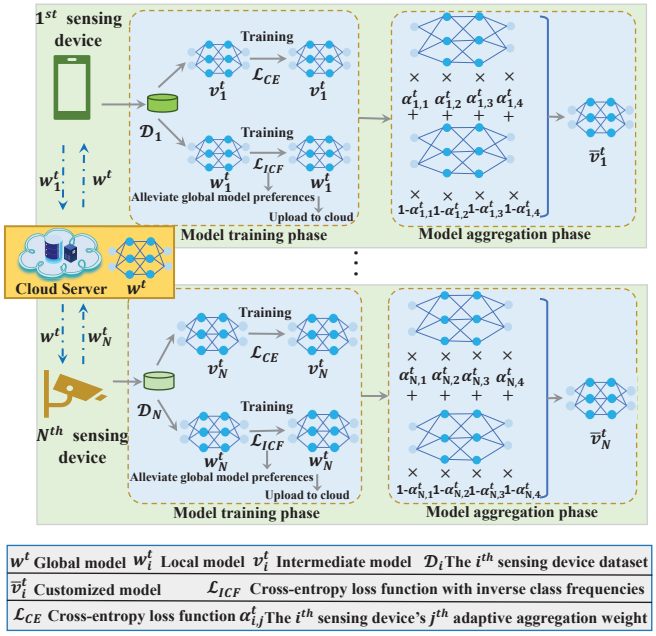

Fig. 2. The workflow of pFL-Sensing.

customized models are acquired through minimizing the global loss:

$$(\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_N) = \arg\min_{\bar{v}} \sum_{i=1}^{N} \frac{m_i}{M} \mathcal{L}_i, \tag{1}$$

where

$$\mathcal{L}_i = \mathcal{L}\left(D_i; \bar{v}_i; w\right), \forall i \in [N], \tag{2}$$

where $\mathcal{L}(.)$ stands for the global loss function. $\mathcal{L}_i$ represents as the loss function of the i^{th} sensing device related to dataset $\mathcal{D}_i$, measuring the discrepancies between the predicted value and the real label of m_i data samples. w stands for the global model, which provides useful information to train the i^{th} sensing device's customized model.

B. System Overview

In order to address the above problem, we present a weighted model aggregation-based pFL framework over edge-cloud collaborative architecture named pFL-Sensing. In each communication round, pFL-Sensing can be decomposed into two phases: a model training phase and a model aggregation phase. Fig. 2 shows the workflow of pFL-Sensing in the t^{th} round of communications. A total number of N sensing devices are involved in the communication round. Taking the i^{th} sensing device as an example, the sensing device stores three models, containing the local model w_i^t, the intermediate version of the customized model called intermediate model v_i^t, and the customized model $\bar{v}_i^t$. The initial intermediate model v_i^t is derived from the customized model $\bar{v}_i^{t-1}$ output in the $t-1^{th}$ round for the i^{th} sensing device. The initial local model w_i^t is distributed by the global model w^t generated through local model aggregations before the t^{th} round of communication on the cloud server. Note that all N local models are aggregated using synchronization. Each sensing device stores these three models.

In the model training phase, dual loss functions are adopted to learn the local model w_i^t and the intermediate model v_i^t. For the training of w_i^t, the cross-entropy loss function [19] is adopted, the most commonly employed loss function for classification tasks. Moreover, to alleviate the low performance of the local model on the minority class in sensing device data due to the global model preference, we introduce ICF into the cross-entropy loss function for updating the local model. ICF serves as a weight factor in the cross-entropy loss function to balance class proportions. We refer to the new loss function as $\mathcal{L}_{ICF}$. After training, we upload the local model to a cloud server for synchronous aggregation, which produces a new global model for the next communication round. Additionally, for the training of v_i^t, we directly use the cross-entropy loss function, which can be denoted as $\mathcal{L}_{CE}$.

In the model aggregation phase, to capture the dynamic role differences of model hierarchy, the hierarchical aggregation and an AWC mechanism are introduced to implement the aggregation of the local model w_i^t and the intermediate model v_i^t for the i^{th} sensing device. Specifically, we first stratify these two models. We then use the AWC to calculate an aggregation weight to each hierarchy, in which the chain rule of differentiation adaptively updates the aggregation weight based on the dynamic changes of each hierarchy's role. Finally, a customized model $\bar{v}_i^t$ for the i^{th} sensing device is acquired through the weighted hierarchical aggregation of w_i^t and v_i^t.

All sensing devices execute the model training phase and the model aggregation phase. Next, we describe these two phases in greater detail.

C. Model Training Phase

As shown in Fig. 2, we design the loss function $\mathcal{L}_{ICF}$ and the loss function $\mathcal{L}_{CE}$ to train w_i^t and v_i^t, respectively. Below, we detail how each loss function is designed to train corresponding models for the i^{th} sensing device in the t^{th} communication round.

1) The loss function $\mathcal{L}_{ICF}$: Due to the adverse effects of the global model preference, ICF is considered in constructing the loss function for training w_i^t. ICF is used to weight loss and improve the proportion of the total loss contributed by the minority class.

The frequency of each class sample is calculated as follows:

$$F_j = m_{i,j}/m_i, j \in [1, C_i], \tag{3}$$

where $m_{i,j}$ denotes as the sample size in the j^{th} class for the i^{th} sensing device. C_i is the total number of classes in the i^{th} sensing device dataset, and F_j denotes the frequency of the j^{th} class. Based on these frequencies, the weight for the loss of each class sample is derived as follows:

$$\mu_j = 1/F_j, j \in [1, C_i], \tag{4}$$

where μ_j represents as the weight for the loss of the j^{th} class sample. The loss function for training w_i^t is calculated as:

$$\mathcal{L}_{ICF} = \sum_{b=1}^{m_i} u_j \xi \left(x_{i,b}, y_{i,b}, w_i^t\right), j \in [1, C_i], \tag{5}$$

where ξ is the cross-entropy loss function, and the sample $(x_{i,b}, y_{i,b})$ belongs to the j^{th} class label, *i.e.*, $y_{i,b} = j$. Finally, gradient descent is employed to update w_i^t denoted as:

$$w_i^t = w^t - \eta \nabla_{w_i^t} \mathcal{L}_{ICF}\left(w^t\right), \tag{6}$$

where $\nabla_{w_i^t}\mathcal{L}_{ICF}(.)$ denotes the stochastic gradient of w_i^t in the loss function $\mathcal{L}_{ICF}$ of the i^{th} sensing device, and η is the learning rate.

2) The loss function $\mathcal{L}_{CE}$: Since the intermediate model v_i^t does not engage in global model aggregation and has no impact on other sensing devices, we directly adopt the cross-entropy loss function ξ. Hence, the loss function $\mathcal{L}_{CE}$ of the intermediate model v_i^t is expressed as follows:

$$\mathcal{L}_{CE} = \sum_{b=1}^{m_i} \xi \left(x_{i,b}, y_{i,b}, v_i^t\right). \tag{7}$$

We then update the intermediate model v_i^t through the following gradient descent:

$$v_i^t = \bar{v}_i^{t-1} - \eta \nabla_{v_i^t} \mathcal{L}_{CE}\left(\bar{v}_i^{t-1}\right), \tag{8}$$

where $\nabla_{v_i^t}\mathcal{L}_{CE}$ stands for the stochastic gradient of v_i^t in the loss function $\mathcal{L}_{CE}$, and $\bar{v}_i^{t-1}$ denotes the customized model from the output of the $t-1^{th}$ round communication.

D. Model Aggregation Phase

To extract dynamic role differences in model hierarchy, we aggregate the local model w_i^t and the intermediate model v_i^t, which are produced by the model training phase of the t^{th} communication round, by hierarchical aggregation and an AWC mechanism. Unlike model aggregation as a whole, hierarchical aggregation considers the roles of different hierarchies in the model. Furthermore, AWC uses the chain rule of differentiation to adaptively update aggregation weights, which reflect the dynamic changes of each hierarchy's roles. Next, we describe the details of this phase.

In the i^{th} sensing device, let $w_i^t = \left(w_{i,1}^t, \ldots w_{i,P}^t\right)$, $v_i^t = \left(v_{i,1}^t, \ldots v_{i,P}^t\right)$, and $\bar{v}_i^t = \left(\bar{v}_{i,1}^t, \ldots \bar{v}_{i,P}^t\right)$ denote each hierarchy of the local model w_i^t, the intermediate model v_i^t, and the customized model $\bar{v}_i^t$, where P is the total number of hierarchies in these models. $w_{i,h}^t$, $v_{i,h}^t$, and $\bar{v}_{i,h}^t$ denote the h^{th} hierarchy of w_i^t, v_i^t, and $\bar{v}_i^t$, respectively. The customized model $\bar{v}_i^t$ can be formulated as follows:

$$\bar{v}_i^t = v_i^t \odot \alpha_i^t + w_i^t \odot \left(\mathbf{1} - \alpha_i^t\right), \tag{9}$$

where $\odot$ is the hadamard product. $\mathbf{1}$ represents the unit vector with the same dimension as α_i^t. $\alpha_i^t = \left(\alpha_{i,1}^t, \ldots, \alpha_{i,P}^t\right) \in \mathbb{R}^{1\times P}$ is the aggregation weight vector, in which $\alpha_{i,h}^t \in [0,1]$ is the h^{th} hierarchy aggregation weight of v_i^t.

According to Eq.(9), it is necessary to balance the aggregation weight vector α_i^t of the local model and the intermediate model when training the customized model. Since we cannot identify the differences between sensing device data distribution and global data distribution, we cannot leverage prior knowledge to determine each sensing device's aggregation weight. In light of this, we propose the AWC mechanism. Specifically, Eq.(9) shows that the customized model $\bar{v}_i^t$ is the function of α_i^t. Therefore, we employ the chain rule of differentiation to update and find the optimal value of α_i^t. The updating process of α_i^t is as follows:

$$\alpha_i^t = \alpha_i^{t-1} - \eta \left(v_i^t - w_i^t\right) \nabla_{\alpha_i^t} f_i\left(\bar{v}_i^{t-1}\right), \tag{10}$$

where $\nabla_{\alpha_i^t} f_i(.)$ denotes the stochastic gradient of α_i^t. Eq.(10) shows that the aggregation weight vector α_i^t is updated based on the differences between the local model and the intermediate model. So if the local model deviates from the intermediate model, α_i^t changes, thus adjusting the balance between the former and the latter.

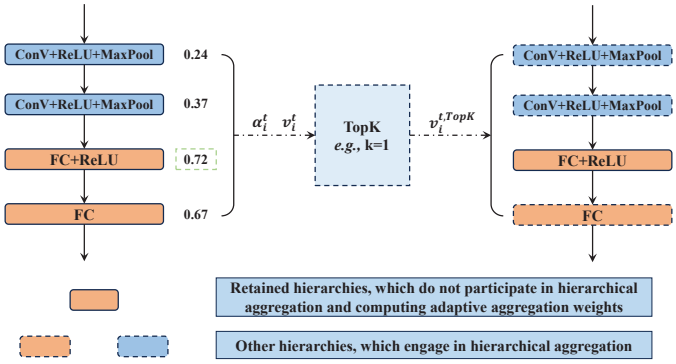

Fig. 3. Description of the TopK mechanism in the AWC. The retained top k hierarchies of the intermediate model do not participate in hierarchical aggregation, while the remaining hierarchies are involved in the aggregation weight calculation and hierarchical aggregation.

Since we need to calculate the aggregation weights of all hierarchies in the hierarchical aggregation, pFL-Sensing incurs significant computational overhead compared to traditional FL such as the FedAvg method [6]. In light of this, a TopK strategy is introduced into the AWC. The key idea of the TopK is to heuristically retain the partial hierarchies of the intermediate model based on the top k aggregation weights. These preserved hierarchies are directly included as a part of the customized model without participating in hierarchical aggregation, thus reducing the computational cost of aggregation weights for k hierarchies. Specifically, by adopting the aggregation weight vector $\alpha_i^t = \left(\alpha_{i,1}^t, \ldots, \alpha_{i,P}^t\right) \in \mathbb{R}^{1\times P}$ for all P hierarchies, we arrange the elements of α_i^t in descending order and select the corresponding top k weights.

$$v_i^{t,TopK} = TopK\left(v_{i,1}^t, \ldots, v_{i,k}^t \mid \alpha_{i,1}^t, \ldots, \alpha_{i,k}^t\right), \tag{11}$$

where $TopK$ denotes the TopK mechanism mentioned above. k is the hyperparameter, preserving the top k hierarchies in the intermediate model.

A detailed description of the TopK mechanism is presented in Fig. 3. When $k = 0$, all hierarchies of the intermediate

model are aggregated. When $k = P$, all hierarchies of the intermediate model are not involved in the aggregation, *i.e.*, $\bar{v}_i^t = v_i^t$. In this case, the customized model is trained only on private sensing device data, without learning any generalized knowledge. pFL-Sensing degenerates into local training.

The hierarchical aggregation weights continue to change throughout the total E communication rounds. At the beginning stage of communications, some hierarchies have large weights. These hierarchies are retained and not updated for a long time. To deal with this issue, we introduce a periodic computing strategy. Specifically, pFL-Sensing calculates aggregation weights of all hierarchies after every Q communication rounds. This effectively prevents aggregation weights of some hierarchies from remaining unchanged for a long time.

III. Experiments

To evaluate the effectiveness of pFL-sensing, we simulate smart sensing environments using real datasets. Specifically, the classification task is treated as a smart sensing application. Sensing devices are represented as clients in this section. The experimental setup and analysis are as follows.

A. Experimental Setup

1) Datasets: We assess pFL-Sensing on two image classification datasets CIFAR-100 and Tiny-ImageNet, and a text classification dataset AG News. The CIFAR-100 and Tiny-ImageNet datasets contain 100 and 200 classes, respectively. The AG News dataset covers 4 classes. The three datasets all have the same amount of samples in each class. Two non-IID scenarios are taken into account: 1) a pathological heterogeneous setting, in which each client randomly selects 10 and 20 classes from the CIFAR-100 and Tiny-ImageNet datasets as their private datasets, respectively, and 2) a practical heterogeneous setting. We simulate a real non-IID scenario through the Dirichlet function [20], which can be denoted as $\mathrm{Dir}(\beta)$, where β is the degree of data heterogeneity. When $\beta \to 0$, client data's heterogeneity is strong. When $\beta \to \infty$, client data tends to IID. CIFAR-100, Tiny-ImageNet and AG News datasets are adopted to the practical heterogeneous setting.

2) Baselines: The performance of pFL-Sensing is compared with advanced baseline algorithms. In addition to FedAvg [6], we also assess Per-FedAvg [7], Ditto [8], pFedMe [9], FedAMP [11], FedPHP [10], FedFomo [12], and PartialFed [13].

3) Model Selection: For the CIFAR-100 dataset, we apply the CNN model with two Conv hierarchies and two FC hierarchies to verify the performance of pFL-Sensing and baselines. For the Tiny-ImageNet dataset, a CNN model with two Conv hierarchies and two FC hierarchies is employed to confirm the performance of pFL-Sensing and the baselines. To verify the performance of pFL-Sensing and the baselines on large-scale models, we also introduce the ResNet-18 model in the Tiny-ImageNet dataset. For the AG News dataset, we use the FastText model with a feature embedding layer, a hidden layer formed by one FC, and an output layer formed by one FC.

TABLE I
THE CLASSIFICATION ACCURACY (%) IN THE PATHOLOGICAL HETER SETTING AND PRACTICAL HETER SETTING

Setting	pathological heter setting		practical heter setting			
Methods	CIFAR-100	TINY	CIFAR-100	TINY	TINY*	AG News
FedAvg	25.98	14.20	31.98	19.46	19.45	79.57
Per-FedAvg	56.80	28.06	44.28	25.07	21.81	93.27
Ditto	67.23	40.23	52.87	32.15	35.92	95.45
pFedMe	58.20	27.71	47.34	26.93	33.44	91.41
FedAMP	64.34	37.15	47.69	27.99	29.11	94.18
FedPHP	63.09	37.88	50.52	35.69	29.90	94.38
FedFomo	62.49	35.87	45.39	30.33	32.84	95.84
PartialFed	65.35	37.76	51.37	32.78	36.91	94.87
pFL-Sensing	68.76	42.16	54.19	39.38	40.42	95.93

4) Parameter Settings: In the experiments conducted on the CIFAR-100, Tiny-ImageNet, and AG News datasets, the communication round E is 500. CNN, ResNet-18, and FastText are set to retain default hierarchies with $k = 1$, $k = 3$, and $k = 1$, respectively, and the batch size is 10. We set the learning rate η to 0.005, while η in ResNet-18 is 0.1. The number of clients N defaults to 20. The aggregation weights of all hierarchies in these three models is initialized to 0.5 for all clients. For each communication round, the default parameter of the Dirichlet function β is 0.1 in the practical heterogeneous setting.

B. Experimental Results Analysis

1) Classification Performance Comparison: On the CIFAR-100, Tiny-ImageNet, and AG News datasets, the average classification accuracy of pFL-Sensing and the advanced baselines in the two non-IID scenarios, the pathological heterogeneous setting and the practical heterogeneous setting, are contrasted in TABLE I. "Tiny" represents using the 4-hierarchy CNN on the Tiny-ImageNet dataset. "Tiny*" denotes utilizing ResNet-18 on the Tiny-ImageNet dataset. The experimental results are as follows:

Pathological Heterogeneous Setting: In the pathological heterogeneous setting, we draw these conclusions: Compared with these baselines, pFL-Sensing achieves the best performance, demonstrating how considering the global model preference with the ICA and dynamic role differences of model hierarchy with the AWC can improve classification precision.

Practical Heterogeneous Setting: We compare the performance of pFL-Sensing and the baselines on the CIFAR-100, Tiny-ImageNet, and AG News datasets in the second non-IID scenario. The experimental results for the default setting $\mathrm{Dir}(0.1)$ are shown in TABLE I. Compared with these baselines, pFL-Sensing achieves the highest classification accuracy because of the inclusion of the global model preference and dynamic role differences of model hierarchy. In contrast to the 4-hierarchy CNN, ResNet-18 is regarded as a large backbone with more hierarchies. Although most methods exhibit higher performance with ResNet-18, pFL-Sensing still performs best, proving that our proposed method is also adaptable to deep neural networks.

2) Aggregation Weight Evolution Analysis: We also conduct an experiment to exhibit the dynamic role differences of model hierarchy in a client through visualizing the aggregation weights of each hierarchy. We use a 4-hierarchy CNN model on the CIFAR-100 dataset with the practical heterogeneous

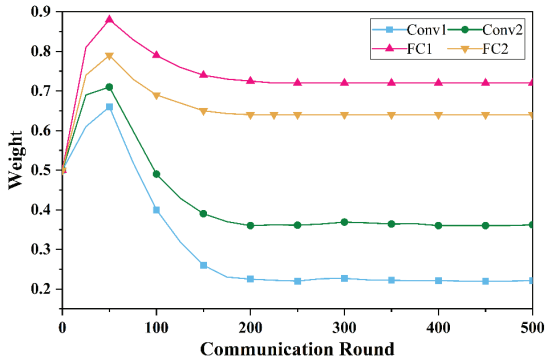

Fig. 4. Aggregation weight evolution on one client of the CIFAR-100 dataset using pFL-Sensing.

setting. Specifically, the first two hierarchies are convolutional hierarchies, which are used for feature extraction and called Conv1 and Conv2, respectively. The third hierarchy is a FC hierarchy named FC1, which is adopted to integrate features. The fourth hierarchy is also a FC hierarchy, which is employed to predict classes. The visualization results are presented in Fig. 4, where the horizontal coordinate denotes the communication rounds. The vertical coordinate is the value of the aggregation weights, ranging from 0 to 1. We draw the following conclusions. Throughout the communication rounds, the dynamic changes of aggregation weights reflect the model's adjustment to personalized and generalized information. In the early stages, since the customized model needs to learn personalized knowledges from individual clients, the intermediate model is given larger weights. As the communication rounds iterate, generalized information requires to be considered for the customized model. Meanwhile, the customized model gradually converges. As a result, the aggregation weights of the intermediate model progressively decrease and tend to stabilize. For pFL, this dynamic weight adjustment process provides an effective mechanism, which reflects an adaptive trade-off between personalization and generalization.

IV. Conclusion

In this paper, we have proposed an edge-cloud enabled weighted model aggregation-based pFL framework named pFL-Sensing for smart sensing applications. In particular, the sensing device has been treated as an edge server. Each edge server has generated a customized model through two phases-a model training phase and a model aggregation phase. In the former phase, to alleviate the global model preference, we have introduced the ICF into the loss function for local model training. In the latter phase, we have integrated hierarchical aggregation and the AWC. We have proposed hierarchical aggregation to aggregate each hierarchy of the local model and the intermediate model with aggregation weights to produce an individual customized model for each client. We also have designed the AWC to adaptively update aggregation weights based on dynamic role differences of model hierarchy. The results of simulation experiments on three typical classification datasets have confirmed the effectiveness of pFL-Sensing on classification accuracy.

References

[1] H. Xu, J. Wu, Q. Pan, X. Guan, and M. Guizani, "A survey on digital twin for industrial internet of things: Applications, technologies and tools," *IEEE Communications Surveys & Tutorials*, 2023.

[2] M. A. Ferrag, O. Friha, B. Kantarci, N. Tihanyi, L. Cordeiro, M. Debbah, D. Hamouda, M. Al-Hawawreh, and K.-K. R. Choo, "Edge learning for 6g-enabled internet of things: A comprehensive survey of vulnerabilities, datasets, and defenses," *IEEE Communications Surveys & Tutorials*, 2023.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[4] M. A. Morid, O. R. L. Sheng, and J. Dunbar, "Time series prediction using deep learning methods in healthcare," *ACM Transactions on Management Information Systems*, vol. 14, no. 1, pp. 1–29, 2023.

[5] Y. Liu, Y. Zhou, K. Yang, and X. Wang, "Unsupervised deep learning for iot time series," *IEEE Internet of Things Journal*, 2023.

[6] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[7] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," *Advances in Neural Information Processing Systems*, vol. 33, pp. 3557–3568, 2020.

[8] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *International Conference on Machine Learning*. PMLR, 2021, pp. 6357–6368.

[9] C. T Dinh, N. Tran, and J. Nguyen, "Personalized federated learning with moreau envelopes," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21 394–21 405, 2020.

[10] X.-C. Li, D.-C. Zhan, Y. Shao, B. Li, and S. Song, "Fedphp: Federated personalization with inherited private models," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2021, pp. 587–602.

[11] 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 *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 9, 2021, pp. 7865–7873.

[12] M. Zhang, K. Sapra, S. Fidler, S. Yeung, and J. M. Alvarez, "Personalized federated learning with first order model optimization," *arXiv preprint arXiv:2012.08565*, 2020.

[13] B. Sun, H. Huo, Y. Yang, and B. Bai, "Partialfed: Cross-domain personalized federated learning via partial initialization," *Advances in Neural Information Processing Systems*, vol. 34, pp. 23 309–23 320, 2021.

[14] J. Zhang, Y. Hua, H. Wang, T. Song, Z. Xue, R. Ma, and H. Guan, "Fedala: Adaptive local aggregation for personalized federated learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 9, 2023, pp. 11 237–11 244.

[15] J. Liu, J. Wu, J. Chen, M. Hu, Y. Zhou, and D. Wu, "Feddwa: personalized federated learning with dynamic weight adjustment," in *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, 2023, pp. 3993–4001.

[16] X. Zhou, Q. Yang, X. Zheng, W. Liang, I. Kevin, K. Wang, J. Ma, Y. Pan, and Q. Jin, "Personalized federation learning with model-contrastive learning for multi-modal user modeling in human-centric metaverse," *IEEE Journal on Selected Areas in Communications*, 2024.

[17] C. You, K. Guo, H. H. Yang, and T. Q. Quek, "Hierarchical personalized federated learning over massive mobile edge computing networks," *IEEE Transactions on Wireless Communications*, 2023.

[18] D. Wang and H. Zhang, "Inverse-category-frequency based supervised term weighting scheme for text categorization," *arXiv preprint arXiv:1012.2609*, 2010.

[19] Y. Wang, J. Chu, Y. Chen, D. Liang, K. Wen, and J. Cai, "Dual entropy-controlled convolutional neural network for mini/micro led defect recognition," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–14, 2023.

[20] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 2351–2363, 2020.