This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3148208, IEEE Transactions on Mobile Computing

Joint Scheduling of Participants, Local Iterations, and Radio Resources for Fair Federated Learning over Mobile Edge Networks

Jiawei Zhang, Suhong Chen, Xiaochen Zhou, Xudong Wang *Fellow, IEEE*, Yi-Bing Lin *Fellow, IEEE/ACM*

Abstract—Federated learning (FL) provides a promising way to train a machine learning model among mobile devices without collecting their raw data to a central node. During training, proper devices are selected to participate in the training process to avoid model unfairness. In a mobile edge network, participant selection must be considered together with three factors: non-iid datasets possessed by devices, tunable local iterations on devices, and radio resource allocation to counter the impact of time-varying channel conditions on parameter transmissions. Since datasets of devices are given, to ensure model fairness and achieve fast convergence in the FL training process, participants, local iterations, and radio resources must be scheduled jointly in each iteration of FL training. In this paper, the joint scheduling problem is analyzed and formulated. Since it is NP-hard, a heuristic scheduling method called PALORA is designed to conduct joint scheduling of participants, local iterations, and radio resources. PALORA consists of three sequentially interactive function blocks: 1) a pointer network embedded deep reinforcement learning method to select participants, 2) an estimation algorithm to determine the numbers of local iterations, and 3) a breadth-first search method to allocate radio resources to the selected participants. PALORA is evaluated via extensive simulations based on real-world datasets. Results show that it significantly outperforms benchmark approaches.

Index Terms—Federated learning, mobile edge network, fairness, deep reinforcement learning, pointer network

1 Introduction

Recent years have witnessed an explosive growth of Internet of Things (IoT) devices. The data collected by IoT devices are usually analyzed using machine learning (ML) techniques, especially deep neural networks (DNNs) [1], to find the insights buried in data. Usually, to train an ML model with high test accuracy, massive training data are necessary. However, sending raw data from IoT devices to a central node results in expensive communication cost and loss of data privacy. To avoid these problems, a new ML paradigm called federated learning (FL) is proposed recently [2]. Its framework is shown in Fig. 1, where IoT devices and an edge server iterate to train an ML model collaboratively. During training, only parameters of local models at IoT devices are sent back to the edge server. Furthermore, differential privacy can be adopted to prevent the edge server from inferring private information from the model parameters [3]. With these benefits, FL has been widely applied in IoT scenarios, like smart home [4] and electric vehicle networks [5].

FL is naturally suitable for mobile edge networks, where mobile IoT devices, e.g., connected vehicles (CVs), connect to an edge server collocated with a base station (BS) to maintain collected data for FL model training. FL over cellular networks was proposed as a study item by the 3rd generation partnership project (3GPP) [6]. However, compared with FL conducted in a cross-silo manner (e.g., hospitals or banks as participants), FL conducted over mobile networks is confronted with the problem that the connections between IoT devices and an edge server are intermittent, due to the movement of IoT devices. Simply selecting all connected devices in the training process will lead to model unfairness after training, i.e., test accuracies vary a lot among different devices, since the contributions of devices to the global model are different. Thus, mobile devices that can participate in each iteration of FL training must be selected properly to ensure model fairness.

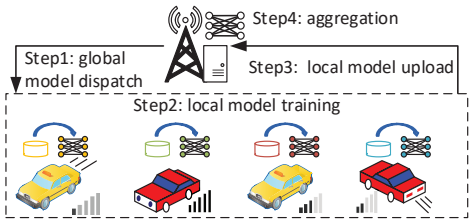


Fig. 1. The framework of federated learning.

Furthermore, participant selection is intertwined with three other common problems of FL, which is summarized in Fig. 2. First, datasets on devices are usually not independent and identically distributed (non-iid) and are in different sizes, which is verified to impact test accuracies of the FL model in different devices [7]. As the datasets are not adjustable, participant selection needs to handle such unequal-sized and non-iid datasets. Second, the number of local iterations in FL training is usually fine-tuned in each iteration so as to accelerate convergence of the FL training process [2], [8]. However, if a device conducts more local iterations than other devices during the entire training process, its local update contributes more to the global

J. Zhang, S. Chen, X. Zhou and X. Wang are with the UM-SJTU Joint Institute, Shanghai Jiao Tong University. Y. Lin is with National Yang Ming Chiao Tung University. Corresponding author: Xudong Wang, Email: wxudong@ieee.org

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3148208, IEEE Transactions on Mobile Computing

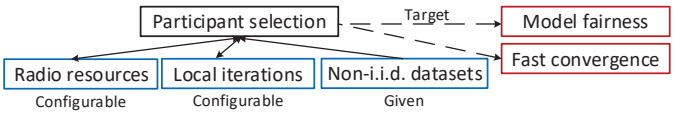


Fig. 2. The relations between impact factors in training FL models.

model than other devices, resulting in unfairness. Thus, participant selection and the number of local iterations must be considered simultaneously. Third, for connected devices, they experience time-varying fading conditions, and channels of different mobile devices are usually heterogeneous. Such communication channel characteristics can severely impact the FL training process, e.g., high time complexity of uploading local updates will slow down the training process. Hence, radio resource allocation needs to be conducted with participant selection to minimize the impact of parameter transmission on the convergence time of FL training.

Consequently, the FL should be coordinated with participant selection, determination of local iterations, and radio resource allocation, subject to real-time varying channel conditions and unequally-sized non-iid datasets. [1] To tackle this problem, the convergence criteria of the FL training with respect to key parameters involved in the joint scheduling problem is analyzed first. Moreover, fairness metrics are designed to measure the fairness of a converged FL model during training and testing, respectively. Afterwards, the joint scheduling problem is formulated to minimize the training time under the constraints of fairness, convergence, and radio resources. The problem is proved to be NP-hard, which illustrates the complexity of joint scheduling of participant selection, local iterations, and radio resources. To this end, a heuristic scheme is designed to conduct joint scheduling of **pa**rticipants, **lo**cal iterations, and **ra**dio resources (PALORA) by leveraging theoretical results derived from the joint scheduling problem. PALORA consists of three function blocks. The first function block selects participants from mobile devices based on deep reinforcement learning (DRL) by treating unknown local iterations and radio resources as part of the environment. DRL is a promising method for combinatorial optimization [9], which it is enhanced in PALORA with several features: 1) A pointer network [10] is embedded in the DRL framework to handle the varying dimensions of the state space due to a different number of devices selected in different iterations; 2) an action amender is designed to skip bad experiences so as to accelerate training. With participants being selected, the second function block determines the numbers of local iterations based on parameters estimated online, while the third function block allocates radio resources to the selected participants via a breadth-first search (BFS) based algorithm.

The performance of PALORA is evaluated via two types of real-world datasets: vehicle trajectory datasets representing mobile devices and public datasets for training ML models. The simulation results demonstrate that PALORA outperforms the state-of-the-art approaches to achieve the goal of joint scheduling among participants, local iterations, and radio resources.

1. This problem is denoted as *the joint scheduling problem* in this paper.

The following contributions are made in this paper:

- The problem that ensures fairness of FL models in mobile edge networks is first studied in this paper. The insights leading to model unfairness are revealed.
- Convergence criteria are analyzed to establish quantitative relationship between the convergence of an FL training process and key parameters such as participant selection and the number of local iterations. The process of training a fair FL model is formulated as a joint scheduling problem, and its complexity is further analyzed.
- A scheduling scheme called PALORA is developed to determine participants, local iterations, and radio resources in an iterative way. PALORA is validated against extensive simulations on real-world datasets.

The paper is organized as follows. In Section 2, the related work is reviewed. The system model is built in Section 3. Theories on training a fair FL model are derived in Section 4. The PALORA scheme is developed in Section 5. Performance evaluation is carried out in Section 6, followed by discussions of potential issues in Section 7. The paper is concluded in Section 8.

2 Related Work

Many studies have focused on optimizing FL over wireless networks recently. The method proposed in [11], [12] leverages optimization methods to determine invariable participants, radio resources, and transmission power during the training process, such that the loss value of the global model can be minimized. The target of the solutions formulated in [13]–[15] is to minimize the training time for an FL model by determining the participants and their radio resource allocations. More specifically, in [13], [14], a neural network is designed to predict the parameters of local models of the participants failed in uploading, so as to alleviate the impacts of transmission failure. To further take the energy consumption into consideration, the solutions in [16]–[18] determine the transmission time, radio resource allocation, transmission power, and CPU resource allocation, for achieving the tradeoff between the training time and the energy consumption. Moreover, in [19], [20], the convergence of FL was analyzed and compared with different scheduling algorithms adopted for selecting participants. The scenario that edge servers cooperate to train an FL model is considered in [8], and the numbers of local iterations are determined so as to minimize the loss value within given a time constraint. The above work indeed addresses some problems in the scenario where FL models are trained in the network edge. However, they only targeted at the convergence of FL models. The fairness issue of mobile devices is not addressed. Moreover, the participants, the number of local iterations, and the radio resources for participants are never jointly selected to optimize training of FL models.

The enhancement of model fairness has attracted researchers' interest recently. The basic idea of the scheme in [21] sets the weight of each local model when the

aggregation is conducted. The local model with a higher loss value is assigned with a higher weight to increase its impact on the global model. The same idea was also applied in a recent FL scheme, i.e., the agnostic FL [22]. Meanwhile, an equal probability of selecting devices as participants is guaranteed in [23] to eliminate the unfairness caused by different participants. These existing solutions cannot be directly applied in mobile edge networks. The methods of setting weights in [21], [22] are only effective when the participants are fixed during the whole training process. This is because the weights cannot be set to the devices that are not selected, and thus the performance on these devices can be poor. The solutions in [23] indeed take participant selection into account. However, due to non-i.i.d datasets and the changing numbers of local iterations during the training process, a simple equal selection probability cannot really ensure model fairness. Moreover, since devices have heterogeneous channel conditions, even participation for devices is not communication efficient.

As mentioned in [24], the data owned by the devices participating in distributed machine learning are usually non-i.i.d, which has been considered in several papers on federated learning. The work in [2] shows the test accuracy can still be high when the participants are randomly selected in each interval. The method proposed in [25] leverages Top-k sparsification to design an efficient communication protocol so that the amount of data transmission can be reduced in the case of non-i.i.d data. The scenario of semisupervised learning is targeted in [26], and swapping of local models among edge devices is enabled to tackle the weight divergence problem caused by non-i.i.d data. Different from the above existing work, we consider the scenario where devices are not always available to participate in training due to changing channel conditions. Meanwhile, we try to achieve the target of training a fair federated learning model (low variance of test accuracy among participants) by jointly selecting participants, determining the number of local iterations, and allocating radio resources.

3 System Model

We consider a mobile edge network consisting of multiple mobile devices, a BS (can be 5G or WiFi), and an edge server connected to the BS. The mobile devices are equipped with various sensors to collect data. The training dataset and the test dataset are randomly sampled from the collected data. The size of the training data set is enough to prevent overfitting. Considering the different amounts of data collected by devices, corresponding amounts of computational resources are reserved to ensure a same time cost for training local models (e.g., 1 GHz CPU frequency for 100 MB data and 2 GHz CPU frequency for 200 MB data). The mobile devices are motived to participant in FL training by incentive mechanisms [27], [28]. For the BS and the edge server, a collection of wireless resources and computational resources are reserved (e.g., via network slicing [29]) for collecting and aggregating the parameters of local models, respectively. To reserve computational resources of devices and the edge server, containers (a lightweight operating-system level virtualization technology widely applied in edge computing [30]) can be adopted, such that FL related processes are not suspended by the preemption of other applications.

TABLE 1
Notations and Descriptions

Notation	Definition
$\mathcal{M}$	Set of devices for FL training
$\mathcal{M}_i$	Set of selectable devices at the i-th interval
$\mathcal{D}_m$	Dataset of device m
$F_m(\cdot)$	Local loss function of device m
$F(\cdot)$	Global loss function
$f(\cdot)$	Loss function
τ_0	The time duration of a slot
L	The number of slots for a local iteration
N_i	The number of local iterations at the i-th interval
$\mathcal{N}$	The set of numbers of local iterations
$c_{m,i}$	Data transmission amount of device m with an RB
$s_{m,i}$	The binary indicator of participant selection
$\mathcal{S}$	The set of selection indicators
σ_0	The bandwidth of an RB
$e_{m,i}(\cdot)$	The binary indicator of allocating an RB
$\mathcal{E}$	The set of RB-allocation indicators
p_i	textcolorredThe packet error rate used in the i-th interval
$\gamma_{m,i}$	The SNR of device m in the i-th interval
B	The total number of RBs at a slot
U	The data amount for a participant to transmit
Q_i	The slots for communication in the i-th interval
t_i	The time of aggregation in the i-th interval
T	The total training time cost
η	Step size of gradient descent (learning rate)
$\omega_m(\cdot)$	The parameters of local model at device m
$\omega(t_i)$	The parameters of global model of the i-th interval
$\omega(T)$	The parameters of the final model after training

In the following subsections, an overview of the FL procedure in a mobile edge network is first presented, followed by the details about local training and parameter transmission. The notations are summarized in Table 1.

3.1 FL Procedure over Mobile Edge Networks

The set of mobile devices willing to cooperate for training an ML model is denoted as $\mathcal{M}$. Each device $m \in \mathcal{M}$ possesses a local dataset $\mathcal{D}_m$. The data distributions can be non-iid. The target of FL is to train a global model with the parameter $\boldsymbol{\omega}$ to minimize the loss value of the global model $F(\boldsymbol{\omega})$. For example, in linear regression, $\boldsymbol{\omega}^T\mathbf{x}_j$ indicates the predicted output, where $\boldsymbol{\omega}$ are the parameters. $F(\boldsymbol{\omega})$ is the average loss value of all devices weighted by their dataset sizes [2], i.e.,

$$F(\boldsymbol{\omega}) = \frac{\sum_{m\in\mathcal{M}} |\mathcal{D}_m| F_m(\boldsymbol{\omega})}{\sum_{m\in\mathcal{M}} |\mathcal{D}_m|}, \tag{1}$$

where $|\mathcal{D}_m|$ is the size of $\mathcal{D}_m$ and $F_m(\boldsymbol{\omega})$ is the local loss function of device m. $F_m(\boldsymbol{\omega})$ is further defined as

$$F_m(\boldsymbol{\omega}) = \frac{1}{|\mathcal{D}_m|} \sum_{(\mathbf{x}_j, y_j)\in\mathcal{D}_m} f(\boldsymbol{\omega}, \mathbf{x}_j, y_j), \tag{2}$$

where $(\mathbf{x}_j, y_j)$ is a data sample ($\mathbf{x}_j$ is the input variable and y_j is the label) in $\mathcal{D}_m$ and $f(\boldsymbol{\omega}, \mathbf{x}_j, y_j)$ is the loss value of the ML model with parameter $\boldsymbol{\omega}$ at data sample j. Examples of commonly-used loss functions [8], [20] are $\frac{1}{2}||y_j - \boldsymbol{\omega}^T\mathbf{x}_j||^2$ (linear regression), $\log[1 + exp(-y_j\boldsymbol{\omega}^T\mathbf{x}_j)]$ (logistic regression), and $\frac{1}{2}\max\{0, 1-y_j\boldsymbol{\omega}^T\mathbf{x}_j\}$ (linear SVM).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3148208, IEEE Transactions on Mobile Computing

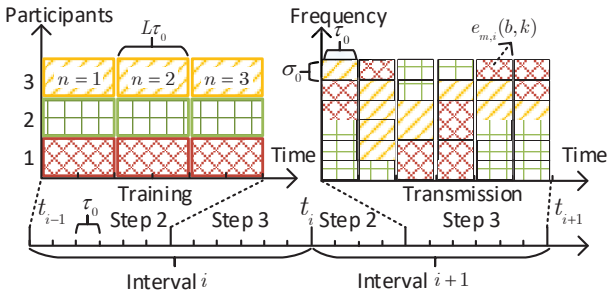


Fig. 3. The timeline of FL over mobile edge networks.

To obtain $\boldsymbol{\omega}$ without gathering all local datasets together, a four-step iterative FL procedure, as shown in Fig. 1, is conducted. Initially, a process of the aggregator is started at the edge server and the processes of local training are started at devices. The aggregator then sends two parameters to these devices: 1) initial parameters of the global model; 2) the learning rate η. Afterwards, in Step 1 a subset of mobile devices are first selected as the participants. In Step 2, the participants conduct multiple local iterations to train their local models. Hereinafter, synchronized FL is considered, i.e., the participants are assumed to start and complete Step 2 simultaneously (ensured by the reservation of computational resources corresponding to data amounts). In Step 3, the parameters of local models are sent back to the edge server. The elapsed time of Step 3 depends on the last participant that finishes transmitting model parameters. In Step 4, the aggregator at the edge server updates the global model based on the results of the local models. In this paper FederatedAverage [2] is adopted for model aggregation, since it is proved to be more communication efficient than other algorithms like FederatedSGD [2]. The above steps iterate until the loss-value and model-fairness requirements are satisfied.

The time line of the FL procedure is shown in Fig. 3. Let τ_0 to be the time unit of the FL procedure, e.g., 0.5 ms in 5G networks. An execution of Step 1-4 in the FL procedure is called an interval. The timestamp at the end of the i-th interval is denoted as t_i ($1 \leq i \leq I$, where I is the total number of intervals of the FL algorithm) and the global model obtained is denoted as $\boldsymbol{\omega}(t_i)$.

In an interval, the time delay of Step 1 is neglected, because the global model can be sent to the participants via a pre-allocated channel in a broadcast way, so its time cost is much smaller than the uplink case (in Step 3). In Step 2, each round of local training consists of multiple iterations, and each consists of L slots. In Step 3, the model parameters of the participants are sent through uplink radio resources within several slots. Finally, the time cost in Step 4 is neglected, because the edge server conducts simple computation (i.e., weighted sum in FederatedAverage) with much more computational resources than that of mobile devices, so the time cost at the mobile devices is negligibly small as compared to Step 2 (i.e., local training time).

3.2 Local Training and Aggregation (Steps 2 and 4)

In Step 2, the participants perform gradient descent [8], [14] to update their local models. For a participant, a local iteration consists of conducting gradient descent once on all of its data samples and then updating its local model. N_i local iterations are conducted in an interval. The updated local model of participant m (denoted as $\boldsymbol{\omega}_m(t_{i-1}+nL\tau_0)$) in the n-th local iteration can then be expressed as the model parameters minus corresponding gradients, i.e.,

$$\begin{aligned}\boldsymbol{\omega}_m(t_{i-1}+nL\tau_0) =& \boldsymbol{\omega}_m(t_{i-1}+(n-1)L\tau_0) \\ &- \eta\nabla F_m(\boldsymbol{\omega}_m(t_{i-1}+(n-1)L\tau_0)),\end{aligned} \tag{3}$$

where ∇F_m denotes the gradient computed based on $\mathcal{D}_m$. For each participant, the training process cannot be preempted thanks to the reserved computational resources by the containers. Hence, local training in an interval can complete within the duration of $N_iL\tau_0$ and the time of completion is $t_{i-1}+N_iL\tau_0$. Afterwards, the updated parameters are transmitted to the aggregator at the edge server (detailed in the next subsection).

With the parameters obtained from the participants, the aggregator updates the global model by performing the weighted average of participants' parameters in Step 4, i.e.,

$$\boldsymbol{\omega}(t_i) = \frac{\sum_{m\in\mathcal{M}}|\mathcal{D}_m|s_{m,i}\boldsymbol{\omega}_m(t_i)}{\sum_{m\in\mathcal{M}}|\mathcal{D}_m|s_{m,i}}, \tag{4}$$

where $s_{m,i}$ is a Boolean value to indicate whether device m is selected in the i-th interval or not, and $\boldsymbol{\omega}_m(t_i)$ represents the local model parameters of device m at the end of the i-th interval.

3.3 Transmission of Model Parameters (Step 3)

In Step 3, model parameters are transmitted via a mobile edge network, e.g., 5G or 802.11 ax, where the multiple access scheme is built based on orthogonal frequency division multiple access (OFDMA). The basic resource unit that can be allocated to a device is a resource block (RB) that consists of a sub-channel (with bandwidth σ_0) in the frequency domain and a slot τ_0 in the time domain. The number of sub-channels assigned to FL training is B, which is equal to the number of RBs in a slot.

For the uplink transmission of model parameters, starting from the next slot after local training ($t_{i-1}+N_iL\tau_0$), the RBs within B sub-channels and Q_i slots are reserved. Due to synchronized aggregation, the number of slots for all participants to complete transmissions in an interval, i.e., Q_i, depends on the last slot of all participants, i.e.,

$$Q_i = \max_{m\in\mathcal{M}}\{q_{m,i}\}, 1 \leq i \leq I, \tag{5}$$

where $q_{m,i}$ is the last slot of the RBs assigned to participant m. Within Q_i slots and B sub-channels, the allocation of an RB to a participant is denoted by a binary indicator $e_{m,i}(b,k)$ where b is the index of sub-channel and k is the index of slots. By allocating an RB to a device, the data transmission amount $c_{m,i}$ of a device is given as

$$c_{m,i} = \tau_0\sigma_0\log(1+\gamma_{m,i}), \forall m \in \mathcal{M}, 1 \leq i \leq I, \tag{6}$$

where $\gamma_{m,i}$ is the uplink signal-to-noise ratio (SNR) of device m in the i-th interval. $\gamma_{m,i}$ varies across different intervals, but remains the same within an interval. Such a consideration is valid for two reasons. First, the coherence time of a vehicle moving with a low speed in the city is relatively large, e.g., more than 20 ms (two 5G frames)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3148208, IEEE Transactions on Mobile Computing

according to some filed tests [31]. Second, the time duration of an interval in federated learning is usually within the coherence time (an example is shown in the simulation setup). To conduct radio resource allocation in such a time-varying channel, $\gamma_{m,i}$ at the beginning of each interval can be measured, and then radio resources are allocated to participants accordingly.

The RBs allocated to each participant need to be large enough to ensure successful transmission of its model parameters and hyper-parameters. In the transmission, each packet can be lost with a packet loss ratio and the lost packet can be fast retransmitted via hybrid automatic repeat request (HARQ) [32] within a frame in 5G or next generation WiFi. Since the time duration between two consecutive intervals is usually within the coherence time, the packet error rate can be measured from the previous interval and utilized in the current interval. For the i-th interval, the number of RBs needed is then determined by taking into account packet error rate p_i and the data amount U of both model parameters and hyper-parameters, i.e., $\frac{U}{1-p_i}$. To this end, the required RBs that are allocated to participant m within $q_{m,i}$ slots must satisfy that the amount of data transmitted in the RBs is larger than the total amount for a successful transmission, i.e.,

$$\sum_{k=1}^{q_{m,i}} \sum_{b=1}^{B} e_{m,i}(b,k) c_{m,i} \geq \frac{U}{1-p_i}, \forall m \in \mathcal{M}, 1 \leq i \leq I, s_{m,i} = 1. \tag{7}$$

Meanwhile, each RB can only be allocated to a device, i.e.,

$$\sum_{m \in \mathcal{M}} e_{m,i}(b,k) s_{m,i} \leq 1, \forall b \in \{1, ..., B\}, k \in \{1, ..., Q_i\}. \tag{8}$$

The time cost of Steps 2-4 in the i-th interval is composed of local training time and uplink transmission time, i.e., $t_i = t_{i-1} + N_i L \tau_0 + Q_i \tau_0$. The total training time T for an FL model is the sum of time for local training and data transmission, i.e.,

$$T = \sum_{i=1}^{I} \tau_0 (L N_i + Q_i). \tag{9}$$

4 Theoretical Study of the Joint Scheduling Problem

In this section, the criteria for the convergence of an FL model trained in a mobile edge network are analyzed. The indexes of model fairness are designed for both testing and training. Based on the analysis and design, the problem of joint scheduling for minimizing the training time of a fair FL model is formulated to further guide the design of the joint scheduling algorithm.

4.1 Convergence Criteria

Although the convergence of FL has been analyzed in existing work [8], [20], such analysis does not take into account the interaction of the above factors. Hence, it cannot be applied to our problem.

The convergence is analyzed by calculating the gap between the loss value of an FL model trained in mobile edge networks and the loss value of the model if trained in a centralized manner with all participants' datasets collected (denoted as the optimal model), i.e., $F[\boldsymbol{\omega}(T)] - F(\boldsymbol{\omega}^*)$, where $\boldsymbol{\omega}^*$ denotes the parameters of the optimal model. Such a gap is denoted as the *loss-value gap* hereinafter.

To facilitate analysis, we give the following properties of the loss functions: 1) $F_m(\boldsymbol{\omega})$ is convex; 2) There exists $\rho > 0$ such that $F_m(\boldsymbol{\omega})$ satisfies

$$||F_m(\boldsymbol{\omega}) - F_m(\boldsymbol{\omega}_0)|| \leq \rho ||\boldsymbol{\omega} - \boldsymbol{\omega}_0||, \tag{10}$$

for any $\boldsymbol{\omega}$ and $\boldsymbol{\omega}_0$ obtained during training (ρ-$Lipschitz$ [33]); 3) There exists $\beta > 0$ such that $F_m(\boldsymbol{\omega})$ satisfies

$$||\nabla F_m(\boldsymbol{\omega}) - \nabla F_m(\boldsymbol{\omega}_0)|| \leq \beta ||\boldsymbol{\omega} - \boldsymbol{\omega}_0||, \tag{11}$$

for any $\boldsymbol{\omega}$ and $\boldsymbol{\omega}_0$ (β-$smooth$ [33]). The above properties are also suitable for $F(\boldsymbol{\omega})$ (can be proved by triangle inequality [8]). These properties are naturally suitable for loss functions of linear models. For non-linear models, the model-parameter space of their loss functions can also be divided into convex regions [34], where the properties are also valid. The effectiveness of the analysis on non-linear models will be verified via follow-up simulation results.

For any interval i and model parameters $\boldsymbol{\omega}$, we define the gradient divergence $\delta_{m,i}$ by the following constraint

$$||\nabla F_m(\boldsymbol{\omega}) - \nabla F_i(\boldsymbol{\omega})|| \leq \delta_{m,i}, \tag{12}$$

where $F_i(\boldsymbol{\omega})$ is the weighted sum of the loss function of selected participants, i.e.,

$$F_i(\boldsymbol{\omega}) = \frac{\sum_{m \in \mathcal{M}} |\mathcal{D}_m| s_{m,i} F_m(\omega)}{\sum_{m \in \mathcal{M}} |\mathcal{D}_m| s_{m,i}}. \tag{13}$$

The average divergence is defined as the weighted sum of the gradient divergence $\delta_{m,i}$ of selected participants, i.e.,

$$\delta_i = \frac{\sum_{m \in \mathcal{M}} s_{m,i} |\mathcal{D}_m| \delta_{m,i}}{\sum_{m \in \mathcal{M}} s_{m,i} |\mathcal{D}_m|}. \tag{14}$$

An auxiliary model with parameters $\boldsymbol{v}_i$ is defined for each interval. At the beginning of an interval, $\boldsymbol{v}_i(t_{i-1})$ is set to the global model $\boldsymbol{\omega}(t_{i-1})$ obtained in the previous aggregation. Afterwards, the model is trained in a centralized manner, i.e., conducting gradient descent on the dataset collected from selected participants and updating the model parameters for each iteration. Formally, $\boldsymbol{v}_i(t_{i-1} + nL\tau_0) = \boldsymbol{v}_i(t_{i-1} + (n-1)L\tau_0) - \eta \nabla F_i(\boldsymbol{v}_i(t_{i-1} + (n-1)L\tau_0))$, where $F_i(\boldsymbol{v}_i(t_{i-1} + (n-1)L\tau_0)) = \frac{\sum_{m \in \mathcal{M}} s_{m,i} F_m(\boldsymbol{v}_i(t_{i-1} + (n-1)L\tau_0))}{\sum_{m \in \mathcal{M}} s_{m,i}}$.

For an FL model that can converge, the direction and modulus of $\nabla F(\boldsymbol{\omega})$ are assumed to be close to those of $\nabla F(\boldsymbol{\omega})$. That is, there exists $\xi \in (0,1)$ such that $||\nabla F(\boldsymbol{\omega}) - \eta\beta \nabla F_i(\boldsymbol{\omega})|| \leq \xi ||\nabla F(\boldsymbol{\omega})||$, where the learning rate η is small (typically $10^{-3} - 10^{-6}$) while the smoothness constant β is limited [8] ($10^{-1} - 10^{1}$ according to our experiments). We also define a training error ζ to capture the desired loss-value gap between the FL model and the optimal model, i.e.,$|F[\boldsymbol{\omega}(T)] - F(\boldsymbol{\omega}^*)| = \zeta$.

Based on the above properties and definitions, the upper bound of the loss-value gap can be given in the following theorem.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3148208, IEEE Transactions on Mobile Computing

Theorem 1. *The upper bound of* $F[\boldsymbol{\omega}(T)] - F(\boldsymbol{\omega}^*)$ *is*

$$F[\boldsymbol{\omega}(T)] - F(\boldsymbol{\omega}^*) \leq \frac{1}{\sum_{i=1}^{I}[\frac{N_i\kappa(1-\xi)}{2\beta} - \frac{\rho\psi_i(N_i)}{\zeta^2}]}, \quad (15)$$

where $\kappa = \min_{i=1}^{I} \frac{1}{||\boldsymbol{v}_i(t_{i-1}) - \boldsymbol{\omega}^*||^2}$, *and*

$$\psi_i(N_i) = \frac{\delta_i}{\beta}((\eta\beta+1)^{N_i} - 1) - \eta\delta_i N_i, 1 \leq i \leq I, N_i \in \mathbb{Z}^+. \quad (16)$$

Proof: The basic idea to derive such a bound of loss-value gap is first to analyze the loss-value gap between the auxiliary model and the optimal model. The gap between the auxiliary model and the FL model trained in the mobile edge network is then analyzed. Based on these two gaps, the bound of loss-value gap is derived. Details of proof can be found in the appendix [35]. □

From Eq. (15), it can be observed that: 1) The selected participants denoted by $s_{m,i}$ determine the average gradient divergence δ_i and further impact $\psi_i(N_i)$ according to Eq. (16); 2) The number of local iterations N_i impacts both $\psi_i(N_i)$ and $\frac{N_i\kappa(1-\xi)}{2\beta}$. Moreover, although radio resource allocations $e_{m,i}(b,k)$ do not change the convergence rate, they impact the convergence time T through data transmission time Q_i.

4.2 Metric of Model Fairness

4.2.1 Fairness index for testing (FITE)

FITE aims to measure the difference of test accuracies among all devices. Considering the varying sizes of datasets possessed by devices, the fairness of a model, denoted as $\Gamma(\boldsymbol{\omega})$, is measured by the coefficient of variation of the test accuracies on all devices, i.e.,

$$\Gamma(\boldsymbol{\omega}) = \frac{\sqrt{w_1[a_1(\boldsymbol{\omega}) - \bar{a}(\boldsymbol{\omega})]^2 + ... + w_m[a_m(\boldsymbol{\omega}) - \bar{a}(\boldsymbol{\omega})]^2}}{\bar{a}(\boldsymbol{\omega})} \quad (17)$$

where $a_m(\boldsymbol{\omega})$ is the test accuracy on device m evaluated in the test dataset of the device; w_m is the weight of the test accuracy $a_m(\boldsymbol{\omega})$, i.e., $w_m = \frac{|\mathcal{D}_m|}{\sum_{m\in\mathcal{M}}|\mathcal{D}_m|}$; $\bar{a}(\boldsymbol{\omega})$ is the weighted test accuracy, i.e. ,$\bar{a}(\boldsymbol{\omega}) = \sum_{m\in\mathcal{M}} w_m a_m(\boldsymbol{\omega})$. In Eq. (17), the weighted variance of test accuracy on each device is normalized by $\bar{a}(\boldsymbol{\omega})$ to eliminate the impact from the scale of test accuracies $\{a_1(\boldsymbol{\omega}), ..., a_m(\boldsymbol{\omega})\}$. $\Gamma(\boldsymbol{\omega})$ then represents the dispersion of the test accuracies around $\bar{a}(\boldsymbol{\omega})$. The lower $\Gamma(\boldsymbol{\omega})$ is, the smaller is the dispersion, i.e., the fairer the model is with parameters $\boldsymbol{\omega}$.

4.2.2 Fairness index for training (FITR)

During training, Eq. (17) is impractical to be leveraged, since devices cannot conduct testing frequently before the end of training. Hence, based on the parameters of local models in each aggregation, FITR (denoted as $\Phi_m(i)$) is designed to ensure fairness during training. The basic idea is to measure the gap between the parameters of the latest local model and the parameters of the up-to-date global model (the gap of parameters of two models is defined as the model gap hereinafter), and then divide such a gap by the number of historical local iterations on each device. $\Phi_m(i)$ is thus expressed as the weighted model gap normalized by the accumulated local iterations, i.e.,

$$\Phi_m(i) = \frac{||\text{diag}(\boldsymbol{z}_{i-1})[\boldsymbol{\omega}_m(t_{i'}) - \boldsymbol{\omega}(t_{i-1})]||}{G_m(i-1)}, \quad 1 \leq i, i' \leq I, m \in \mathcal{M}, \quad (18)$$

where i' is the index of the latest interval when device m is selected, and $\boldsymbol{z}_i$ is a vector representing the importance of corresponding model parameters to the model performance, i.e., test accuracy, during training [36]. diag($\boldsymbol{z}_i$) is a diagonal matrix, in which an element in the diagonal is $\boldsymbol{z}_i$. $\boldsymbol{z}_i$ is updated following a weighted low-pass filter, i.e.,

$$\boldsymbol{z}_i = (1-g_1)\boldsymbol{z}_{i-1} + g_1|||\boldsymbol{\omega}(t_i) - \boldsymbol{\omega}(t_{i-1})|||, 1 \leq i \leq I, \quad (19)$$

where g_1 is a control parameter and $|||\cdot|||$ is the operation to take the absolute value of each element in a vector to find the variation of model parameters [36]. Hence, the numerator of Eq. (18) represents the weighted gap between the local model of device m and the global model before the i-th aggregation is conducted. The denominator in Eq. (18) is the historical number of local iterations performed by device m, i.e.,

$$G_m(i) = (1-g_2)G_m(i-1) + g_2 N_i s_{m,i}, \quad 1 \leq i \leq I, \forall m \in \mathcal{M}, \quad (20)$$

which is also updated following a weighted low-pass filter with a control parameter g_2. The average FITR is further defined as the average $\Phi_m(i)$ of all devices, i.e., $\Phi(i) = \frac{\sum_{m\in\mathcal{M}}\Phi_m(i)}{|\mathcal{M}|}$.

FITR guides the training process to ensure fairness in an FL model, as long as devices with a larger FITR are selected with a higher probability. If a device with a major data distribution is selected, its model gap decreases, but its accumulative number of iterations increases. Hence, its index decreases, which reduces the chance of being selected again in the next interval of training and thus improves the opportunity of being selected for other devices. Meanwhile, for the devices whose data distributions are minorities, FITR can be large due to large model gaps. They are then more likely to be selected than the devices with major data distributions, and will contribute more to the aggregation of the global model. As a result, when the global model gradually converges and the average FITR $\Phi(i)$ becomes small, the gap between the global model and each local model is small.

The relation between FITE and FITR is then revealed, which is based on the basic principles of machine learning rather than deriving an equation to establish a theoretical relation due to two reasons. First, the average FITR involves participants' model parameters that are related to their data distribution and trajectories (determining whether a participant can be selected). However, no statistical features should be specified for the collected data and trajectories of mobile devices. Second, there exists no mathematical formula to characterize the relation between model parameters and test accuracy. Hence, relation between FITE and FITR is presented as follows.

For a global model with a small average FITR $\Phi(i)$, it fits well with the local datasets and achieves similar test accuracies on different devices. The reason is as follows.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3148208, IEEE Transactions on Mobile Computing

For a device, both of its training dataset and its test dataset are randomly sampled from the collected data, i.e., they are i.i.d, and the size of the training dataset is large enough to prevent overfitting. In this case, for a linear model, a small model gap (represented by FITR) indicates the inference results from the global model and the latest local model have only a slight difference, i.e., the test accuracies of both models are close to each other. For a non-linear model like the convolutional neural network (CNN), it can be viewed as a linear one when it approaches convergence [34]. Hence, a small gap also implies close test accuracies. Therefore, a small gap between the global model and local models (i.e., small FITR $\Phi_m(i)$) means that the test accuracy difference between devices is small (i.e., small FITE $\Gamma(\boldsymbol{\omega})$).

Meanwhile, a small FITE $\Gamma(\boldsymbol{\omega})$ of a converged FL model indicates that the global model has high test accuracies for all devices, which further implies the global model can accurately recognize the patterns of the test datasets of devices. Since the training dataset and the test dataset are i.i.d for each device, the patterns of the training datasets are the same as that of the test datasets. In this case, local training of the global model at each device will lead to small gradients, i.e., the update to the global model is tiny. Therefore, FITE $\Gamma(\boldsymbol{\omega})$ and FITR $\Phi_m(i)$ are consistent with each other.

4.3 Complexity Analysis of the Joint Scheduling Problem

Based on the convergence analysis and the fairness metrics, the problem of joint scheduling for training a fair FL model in mobile edge networks can be formulated. In a mobile edge network, the cost of reserving computational resources of each mobile device and the communication cost for transmitting model parameters during training are high. Hence, the goal of the scheduling problem is to minimize the total time T for achieving a converged fair FL model, by jointly determining participants $\mathcal{S}$ (the set of $s_{m,i}$), the numbers of local iterations $\mathcal{N}$, and the radio resource allocation $\mathcal{E}$ in all intervals. This goal must take into account three constraints. The first one is the convergence of the FL model, which can be represented as a low loss-value gap between the optimal model and the FL model trained in a mobile edge network. Achieving a low loss-value gap requires the gap to be less than a threshold, e.g., $\frac{1}{S}$. By taking the reciprocal of Eq. (15), we have

$$\sum_{i=1}^{I}\left(\frac{N_i\kappa(1-\xi)}{2\beta}-\frac{\rho\psi_i(N_i)}{\zeta^2}\right)\geq S, \tag{21}$$

where $\psi_i(\cdot)$ is defined in Eq. (16). The second constraint is that the global model needs to be fair. Since fairness can be ensured by selecting the devices with high fairness indexes, we need to consider the following rule to select participants in each interval: The average fairness indexes of the selected devices are higher than that of all selectable devices whose SNRs are above a certain threshold γ_0. Such a requirement can then be expressed as

$$\frac{\sum_{m\in\mathcal{M}} s_{m,i}|\mathcal{D}_m|\Phi_m(i)}{\sum_{m\in\mathcal{M}} s_{m,i}|\mathcal{D}_m|}\geq\frac{\sum_{m\in\mathcal{M}_i}|\mathcal{D}_m|\Phi_m(i)}{\sum_{m\in\mathcal{M}_i}|\mathcal{D}_m|}, 1\leq i\leq I, \tag{22}$$

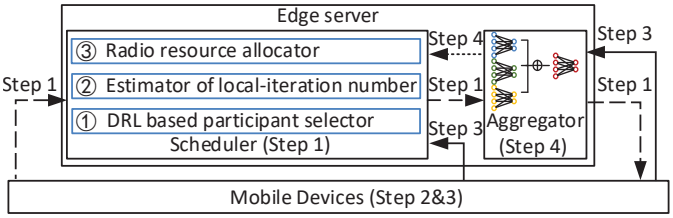


Fig. 4. The FL system with PALORA.

where $\mathcal{M}_i=\{m\in\mathcal{M},\gamma_{m,i}\geq\gamma_0\}$ is the set of selectable devices in the i-th interval, and FITR $\Phi_m(i)$ is calculated based on Eq. (18)-(20). The third constraint is that the radio resources for transmitting model parameters satisfy the constraints given in Eq. (6)-(8).

As a result, the joint scheduling problem can be formulated as

Problem Ω:

$$\begin{aligned}\min_{\mathcal{N},\mathcal{S},\mathcal{E}}\quad & \sum_{i=1}^{I}\tau_0(LN_i+t_cQ_i)\\ \text{s.t.}\quad & \text{Eq.}(6)-(8),(16),(18)-(22),\end{aligned}$$

where Eq. (16) and Eq. (21) are the constraints of model convergence; Eq. (18)-(20), (22) are the constraints of model fairness; Eq. (6)-(8) are the constraints of radio resources. Ω is NP-hard as stated in the following theorem..

Theorem 2. *Problem Ω is an NP-hard combinatorial optimization problem.*

Proof: The NP-hardness is proved by reducing the maximum density knapsack problem to Ω. See the appendix [35] for details. □

5 PALORA: The Joint Scheduling Scheme

Since Problem Ω is NP-hard, a heuristic scheme called PALORA is designed to conduct joint scheduling. To reduce performance deviation from the optimal result in Problem Ω, PALORA must be designed by leveraging theoretical results derived in Section 4.

5.1 Basic Idea of Designing PALORA

Problem Ω spans the entire training process. This feature does not match the iterative training process of federated learning. Thus, PALORA must meet two design requirements: 1) It must take an iterative approach to match the interval-by-interval procedure of federated learning; 2) In each iteration, the complexity of scheduling needs to be reduced significantly. To satisfy the first requirement, PALORA conducts participant selection, determination of local iterations, and radio resource allocation jointly in each interval instead of the entire training process. To satisfy the second requirement, the joint scheduling problem within one interval is simplified into a sequential scheduling mechanism. More specifically, considering the i-th interval, the participants that satisfy the constraints of model fairness Eq. (18)-(20),(22) are selected, and then the number of local iterations and radio resources are determined. The latter problem can be further decoupled, because, given the selected participants, determining the number of local iterations N_i is independent from radio resource allocation.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3148208, IEEE Transactions on Mobile Computing

To determine N_i, the convergence rate in each interval needs to be maximized, so the constraints of model convergence Eq. (16) and Eq. (21) need to be considered. With the participants determined, radio resources are determined to minimize transmission time Q_i. Hence, the radio resource $\{e_{m,i}(b,k)|m \in \mathcal{M}_i\}$ are determined under the constraints Eq. (6)-(8) to minimize Q_i.

Participant selection is a complicated combinatorial optimization problem related with the number of local iterations and radio resource allocation, but they are determined in separate steps. Therefore, deep reinforcement learning (DRL) is leveraged to provide an effective solution to complicated optimization problems [9] for PALORA. Furthermore, DRL can treat unknown parameters (such as the number of local iterations and radio resources) as part of the environment and then ensures actions (i.e., participant selection) to match the environment.

Therefore, PALORA is designed with three function blocks. A DRL based function block is designed to select participants. The information of the selected participants is then sent to the second and third function blocks to determine the number of local iterations and allocate radio resources, respectively. An FL system with PALORA is illustrated in Fig. 4, where the edge server is equipped with two modules, i.e., the FL scheduler with PALORA and the FL aggregator.

5.2 DRL Based Participant Selector (Function Block 1)

DRL selects the participants such that the constraints of fairness and convergence, i.e., Eq. (18)-(22), are satisfied.

5.2.1 DRL Model Design

In the DRL model, the state space, action space, and reward in the problem of selecting participants are designed as follows.

State. The state of the i-th interval, denoted as $\mathbf{s}_i$, is comprised of the channel conditions at the beginning of the i-th interval, FITR in the previous interval, and the weighted sum of participants' local loss values $F_{i-1}[\boldsymbol{\omega}(t_{i-2})]$ (calculated via Eq. (13)) received from the aggregator at Step 4 in the previous interval, i.e., $\mathbf{s}_i = [\{\Phi_m(i-1)|m \in \mathcal{M}_i\}, \{\gamma_{m,i}|m \in \mathcal{M}_i\}, \{F_{i-1}(\boldsymbol{\omega}(t_{i-2}))\}]$.

Action. The action taken by the agent is the selection of participants, i.e., $\mathbf{a}_i = [s_{1,i}, ..., s_{|\mathcal{M}|,i}]$.

Reward. The reward gained from applying $\mathbf{a}_i$ to the FL aggregator can be evaluated by the rate of loss-value descent (i.e., the difference of average loss values between two intervals divided by the time cost of an interval), and also by the satisfaction of the fairness constraint Eq. (22), i.e.,

$$r_i = \frac{\alpha[F_i(\boldsymbol{\omega}(t_{i-1})) - F_{i-1}(\boldsymbol{\omega}(t_{i-2}))]}{t_i - t_{i-1}} + \mu[\frac{\sum_{m\in\mathcal{M}} s_{m,i}|\mathcal{D}_m|\Phi_m(i)}{\sum_{m\in\mathcal{M}} s_{m,i}|\mathcal{D}_m|} - \frac{\sum_{m\in\mathcal{M}_i} |\mathcal{D}_m|\Phi_m(i)}{\sum_{m\in\mathcal{M}_i} |\mathcal{D}_m|}], \tag{23}$$

where α and μ are weights. Such a reward is calculated starting from the second interval and $\boldsymbol{\omega}(t_0)$ represent the parameters of the initial global model.

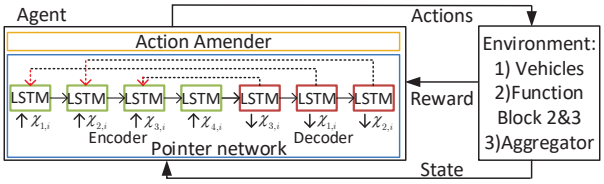


Fig. 5. The framework of pointer network embedded DRL.

5.2.2 Pointer Network Embedded DRL

In the DRL model, the dimension of the state space varies, due to the varying number of selectable devices ($\mathcal{M}_i$) in different intervals. Moreover, the total number of devices $\mathcal{M}$ in the system does not have a known upper limit for training different FL models. Hence, the classical fully-connected neural network that requires a fixed-size input is no longer suitable. To address this challenge, the pointer network [10] that can deal with input data with variable sizes is adopted. It includes two recurrent neural networks (RNNs) constructed with long short term memory (LSTM) [37] units, i.e., the encoder and the decoder. Each decoder unit generates an output element selected from the input sequence to form the output sequence. More details about the pointer network are provided in the appendix [35].

We apply the pointer network in the DRL agent, as shown in Fig. 5 (marked by a blue box). The input sequence is 3-tuple state $\{< \Phi_m(i-1), \gamma_{m,i}, F_{i-1}(\boldsymbol{\omega}(t_{i-1})) > |m \in \mathcal{M}_i\}$. Such a 3-tuple is denoted as $\chi_{m,i}$ hereinafter. The decoder returns a sequence of selected devices, which are considered as the action chosen by the pointer network.

5.2.3 Training procedure

To train the above DRL model, a training algorithm called PALORA-Train is designed. Policy gradient is adopted as the basic training method, since it can easily choose an action from a high-dimension action space. A typical policy gradient algorithm is REINFORCE [38]. To stabilize training, experience replay proposed in [39] is also utilized, where mini-batches are sampled from a pool storing previous experiences for training. To integrate experience replay with the REINFORCE algorithm, importance sampling [40] is applied to cancel the bias between the policy corresponding to the recorded experiences and the latest policy parameterized by the deep neural network.

To accelerate learning, we further design an action amender as shown in Fig. 5 (marked by a yellow box). The action from the pointer network, denoted as $\mathbf{a}'_i$, is usually improvable when the agent is inexperienced, e.g., selecting only a few participants when there are a large number of available devices. Such experiences are useless and can slow down the training of the pointer network. To reduce the effect of bad experiences, amending the actions $\mathbf{a}'_i$ at the beginning of DRL training can significantly speed up convergence [41], [42]. Thus, for the first time when the agent trains an FL model, the action sampled from the pointer network needs to be amended. The amending rule is designed as follows: If the selected participants are less than ϕ percents of the total selectable devices, the remaining devices are selected according to the descending order of SNR until enough participants are selected; On the contrary, if the selected participants are more than ϕ, the ones with the worst SNR are unselected one by one.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3148208, IEEE Transactions on Mobile Computing

The DRL model is trained in parallel with the training of the FL model. At the beginning of each interval of FL, an action is selected and applied. The reward and update state are obtained at the end of the interval. The training details are shown in Algorithm 1. The pointer network is first initialized with randomly generated parameters $\boldsymbol{\theta}$ in Line 1. Meanwhile, if the same type of model, e.g., CNN with the same architecture, is trained by PALORA-Train previously, the parameters of the previous DRL model $\boldsymbol{\theta}$ can be inherited for the initialization. The index of interval and the time are initialized in Line 2. Starting from Line 3, PALORA-Train iterates until either of the stop criteria is satisfied: 1) the change of loss value is smaller than a predefined threshold and the variance of fairness indexes $\Phi_m(i)$ is smaller than a constraint; 2) The total time for training runs out. When an interval begins, the system state $\mathbf{s}_i$ is observed and reorganized as tuples of devices' information, i.e., $\{\chi_{m,i}|m \in \mathcal{M}_i\}$. They are fed into the pointer network sequentially. $\mathbf{a}_i'$ is then sampled from the policy $\pi(\mathbf{a}_i|\mathbf{s}_i, \boldsymbol{\theta})$, as shown in Line 4. If the execution time of the previous interval t_{i-1} is smaller than the time that needs action amending (denoted as T'), then $\mathbf{a}_i'$ is modified to $\mathbf{a}_i$ as shown in Line 6-14. Otherwise, $\mathbf{a}_i'$ is directly adopted as $\mathbf{a}_i$ in Line 15-16. In Line 17, the action $\mathbf{a}_i$ is applied to the environment including the function blocks 2 and 3, the aggregator, and the mobile devices, and then determines the reward r_i and the next system state $\mathbf{s}_{i+1}$ at the end of the i-th interval. In Line 18-19, the behavioral likelihood $\mathcal{H}_i$ is then calculated, and then is stored together with the state, the action, and the reward, as an experience in the experience pool. In Line 20-21, the pointer network is updated as follows. First, the REINFORCE stochastic gradient $\mathbf{g}$ is calculated as

$$\mathbf{g} = \frac{1}{|\mathcal{R}|} \sum_{(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathcal{N}_i) \in \mathcal{R}} r_i \frac{\pi(\mathbf{a}_i|\mathbf{s}_i, \boldsymbol{\theta})}{\mathcal{H}_i} \nabla_{\boldsymbol{\theta}} \log \pi(\mathbf{a}_i|\mathbf{s}_i, \boldsymbol{\theta}), \quad (24)$$

where $\mathcal{R}$ is the experience pool and $|\mathcal{R}|$ is the pool size. Second, the parameters $\boldsymbol{\theta}$ are then updated with $\mathbf{g}$. In Line Line 22, the index of the i-th interval and the time t_i are updated. The complexity of this function block is $o(EH)$, where E and H are the dimensions of the embedding layer and the hidden layer of the pointer network, respectively.

Algorithm 1 PALORA-Train

1: Initialize pointer network with parameters $\boldsymbol{\theta}$ and indicate the aggregator to initialize the parameters $\boldsymbol{\omega}$ of the global model;
2: $i \leftarrow 1$, $t_0 \leftarrow 0$;
3: **while** the stop criteria of FL are not satisfied **do**
4: Sample action $\mathbf{a}_i'$ from the policy $\pi(\mathbf{a}_i|\mathbf{s}_i, \boldsymbol{\theta})$ represented by the pointer network and obtain corresponding $\{s_{m,i}\}$;
5: **if** t_{i-1} is less than the time for action amending T' **then**
6: **if** $\sum_{m \in \mathcal{M}_i} s_{m,i} < \phi|\mathcal{M}_i|$ **then**
7: **while** $\sum_{m \in \mathcal{M}_i} s_{m,i} < \phi|\mathcal{M}_i|$ **do**
8: Find the device $m \in \{m|m \in \mathcal{M}_i, s_{m,i} = 0\}$ with the maximum $\gamma_{m,i}$;
9: $s_{m,i} \leftarrow 1$;
10: **else**
11: **while** $\sum_{m \in \mathcal{M}_i} s_{m,i} > \phi|\mathcal{M}_i|$ **do**
12: Find the device $m \in \{m|m \in \mathcal{M}_i, s_{m,i} = 1\}$ with the minimum $\gamma_{m,i}$;
13: $s_{m,i} \leftarrow 0$;
14: $\mathbf{a}_i \leftarrow \{s_{m,i}\}$;
15: **else**
16: $\mathbf{a}_i \leftarrow \mathbf{a}_i'$
17: Apply $\mathbf{a}_i$ in the environment, obtain N_i and Q_i from function blocks 2 & 3 respectively, and calculate r_i via Eq. (23) and $\mathbf{s}_{i+1}$;
18: Calculate behavioral likelihood $\mathcal{H}_i \leftarrow \log \pi(\mathbf{a}_i|\mathbf{s}_i, \boldsymbol{\theta})$;
19: Store experience $(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathcal{H}_i)$ in $\mathcal{R}$;
20: Calculate gradient g from Eq. (24);
21: Update $\boldsymbol{\theta}$ with gradient $\mathbf{g}$ using Adam optimizer;
22: $i \leftarrow i + 1$, $t_i \leftarrow t_{i-1} + LN_i\tau_0 + Q_i\tau_0$;

5.3 Estimator of Local-Iteration Number (Function Block 2)

In function block 2, we aim to find the proper N_i that minimizes the loss-value gap. According to Eq. (1), given the selected participants, $\frac{\kappa(1-\xi)}{2\beta} + \eta\delta_i$, $\frac{\rho\delta_i}{\beta\zeta^2}$, and $\eta\beta + 1$ are then determined, which are denoted as A_1, B_1, and C_1 in the following statement, respectively. The convergence rate then becomes

$$F(\boldsymbol{\omega}(T)) - F(\boldsymbol{\omega}^*) \leq \frac{1}{\sum_{i=1}^{I}[A_1N_i - B_1(C_1^{N_i} - 1)]}. \quad (25)$$

The high convergence rate can be achieved by maximizing $A_1N_i - B_1(C_1^{N_i} - 1)$. Since C_1 is larger than 1, the corresponding N_i can be directly determined, if the values of A_1, B_1, and C_1 are known. A_1, B_1, and C_1 are related to β (β-$smooth$), η (learning rate), δ_i (the average gradient divergence), ρ (ρ-$Lipschitz$), ζ (the desired loss-value gap between the FL model and the optimal model), and two constants in the upper bound of the loss-value gap, i.e., κ and ξ. Among them, η is given. κ and ζ depend on the final model parameters upon completion of training, and ξ involves $\nabla F(\boldsymbol{\omega}(t_i))$ that cannot be obtained online (e.g., the gradients of all devices including the ones that are not selected). Hence, they cannot be determined before or during training. Instead, they are treated as hyper-parameters and tuned via experiments.

ρ, β, and δ_i are related to loss functions, but they can be estimated during training. The estimation method is illustrated below. The basic idea of estimating ρ and β is to keep sampling model parameters $\boldsymbol{\omega}$ and updating the maximal estimates of ρ and β in each interval for approaching true ρ and β. The estimation is conducted following Eq. (10) and Eq. (11). More specifically, in the i-th interval, the participants conduct local training to obtain $\boldsymbol{\omega}_m(t_i)$ (Step 2), and then calculate $\rho_{m,i} = \frac{||F_m(\boldsymbol{\omega}(t_{i-1})) - F_m(\boldsymbol{\omega}_m(t_i))||}{||\boldsymbol{\omega}(t_{i-1}) - \boldsymbol{\omega}_m(t_i)||}$ and $\beta_{m,i} = \frac{||\nabla F_m(\boldsymbol{\omega}(t_{i-1})) - \nabla F_m(\boldsymbol{\omega}_m(t_i))||}{||\boldsymbol{\omega}(t_{i-1}) - \boldsymbol{\omega}_m(t_i)||}$. Both $\rho_{m,i}$ and $\beta_{m,i}$ are transmitted back to the scheduler at the edge server in Step 3. The scheduler then takes the largest $\rho_{m,i}$ and $\beta_{m,i}$ as ρ and β, respectively. At the beginning of training, the gap between the estimated values of ρ and β and the true values can be large. However, the difference quickly diminishes and its impact on the result of N_i is limited, as confirmed by our experiments.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3148208, IEEE Transactions on Mobile Computing

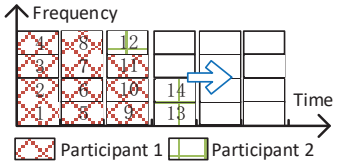


Fig. 6. The process of Function Block 3.

Similarly, δ_i is estimated in each interval. The scheduler first calculates $\nabla F_m(\boldsymbol{\omega}_m(t_i)) = \frac{\boldsymbol{\omega}_m(t_i)-\boldsymbol{\omega}(t_{i-1})}{\eta}$ based on the local models received in the i-th interval and the global model in the $(i-1)$-th interval. $\nabla F_i(\boldsymbol{\omega}(t_i))$ is then calculated based on the global model in two intervals, i.e., $\nabla F_i(\boldsymbol{\omega}(t_i)) = \frac{\boldsymbol{\omega}_{(t_i)}-\boldsymbol{\omega}(t_{i-1})}{\eta}$. Afterwards, $\delta_{m,i}$ is approximated by taking the equality of Eq. (12), i.e., $\delta_{m,i} = ||\nabla F_m(\boldsymbol{\omega}) - \nabla F_i(\boldsymbol{\omega})||$. δ_i can then be calculated according to Eq. (14).

With the calculation above, ρ, β, δ_i can be obtained in the i-th interval. They are applied in the $(i+1)$-th interval for determining the number of local iterations. The complexity of the second function block is $o(1)$.

5.4 Radio Resource Allocator (Function Block 3)

The function block 3 aims to allocate the RBs to the selected participants within the reserved subchannels for federated learning, so as to minimize the transmission time in an interval. More specifically, given the SNRs of participants $\gamma_{m,i}$, the target is to minimize the finish time of the last participant Q_i by determining $\{e_{m,i}(b,k)\}$, under the constraints given in Eq. (6)-(8).

This combinatorial problem can be solved with polynomial time, as explained below. Considering that: 1) the SNRs of participants within an interval remain invariable; 2) the frequency selectivity can be neglected thanks to a small number of subchannels, the data transmission amount via any RB is the same. Thus, the resource allocation problem can be solved simply by a breadth-first search (BFS) based algorithm. More specifically, given a participant, the required number of RBs is calculated following the same rule as Eq. (7). Next, starting from the first slot, available RBs with the smallest slot index are sought and allocated until the first participant gets sufficient RBs. For example, RB indexes are indexed, as shown in Fig. 1. Starting from the first RB, 11 continuous RBs are allocated to Participant 1. The RB allocation result $\{e_{m,i}(b,k)\}$ is then updated. Afterwards, starting from the allocable RB with the smallest index, i.e., 12, RBs with continuous indexes are allocated to Participant 2. The same procedure is also conducted for the remaining participants. With such a BFS based algorithm (with breadth denoting the RBs of all subchannels in a given time slot), the RBs of all the sub-channels in a time slot are allocated first to participants and then the next time slot is considered. Therefore, when the number of RBs for all participants are allocated, the total transmission time required by all these participants is minimized. The complexity is the same as BFS, i.e., $o(n)$, where n is the number of selected participants.

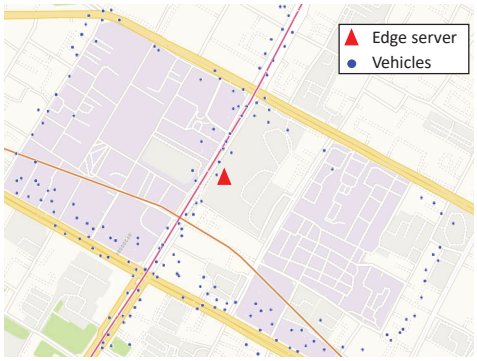


Fig. 7. The geographical distributions of vehicles and the edge server.

6 Performance Evaluation

6.1 Simulation Settings

Platform Setup. Simulations are conducted on a workstation (CPU: Intel i9-10900K @5.1GHz; RAM: 64 GB DDR4 3200 MHz; GPU: NVIDIA GeForce RTX 3090 with CUDA version 11.1; OS: Manjaro 20.2 with Linux kernel version 5.9). The simulator is composed of three parts: 1) the FL part that instantiates FL clients for training and aggregate local models, 2) the scheduler of PALORA, and 3) the environment that represents channel conditions. For the FL part, we utilize the FedML framework [43], which supports to instantiate FL clients with given datasets and opens the API for a customized scheduler. For the scheduler, we leverage the PyTorch library (version 1.1.0) to build the DRL model and the embedded pointer network. The environment is built using Python 3.8.

Dataset Description. The datasets utilized in the simulator can be classified into two types: 1) the datasets for representing locations of mobile devices and an edge server in the environment, and 2) the datasets for training FL models in the FL part. For the first type, we further adopt two datasets: 1) the locations of 46050 base stations in Chengdu City, and 2) the trajectories of Didi expresses in Chengdu City. Among all base stations, we select the one located at 30.658° N and 104.082° E as the edge server to provide service. We also choose the vehicles within the area around the edge server (30.655° N - 30.660° N and 104.078° E - 104.086° E, which covers a 500 m $\times$ 800 m area of a crossroad). The area of moving vehicles (blue dots show the vehicle in a slot) and the location of the edge server are shown in Fig. 7. From the map, it can be observed that most of the vehicles move around the edge server. For the second type of datasets, we adopt public datasets, including Federated EMNIST (671,585 data samples) and MNIST (61,664 data samples) that are adopted in FedML. These datasets are segmented into non-i.i.d batches [43] for FL clients. The average data sizes for each client are 198 samples and 62 samples, respectively.

System Setup. Typical ML models are considered for FL training, i.e., a support vector machine (SVM) and a logistic regression (LR). Moreover, a two-layer convolutional neural network (CNN) is also adopted to evaluate the performance of PALORA for the model with non-convex loss functions. The file sizes of recording their model parameters are 31 KB, 31 KB, and 4.7 MB, respectively. Their computational resources required for a local iteration on local datasets are 0.2 GFLOP, 0.4 GFLOP, and 16 GFLOP respectively.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3148208, IEEE Transactions on Mobile Computing

The hyper-parameters in PALORA are listed as follows. In PALORA-Train of the first function block, the learning rate of the DRL model is 0.01; The weights of the reward function are $\alpha = 10000$ and $\mu = 10$; The slots for action amender T' is 36,000; The size of the memory pool $|\mathcal{R}|$ is 16. The control parameters of fairness indexes are $g_1 = 0.1$ and $g_2 = 0.1$. The threshold of loss value is 0.6, 1.9, and 0.6 for SVM, LR, and CNN, respectively. The variance constraint of $\Phi_m(i)$ is 0.5, 0.7, and 0.6 for SVM, LR, and CNN, respectively. The parameters in the second function blocks are set as follows: $\eta = 0.0006, \kappa = 12, \zeta = 0.5, \xi = 0.7$ for SVM, $\eta = 0.003, \kappa = 10, \zeta = 1.7, \xi = 0.7$ for LR, and $\eta = 0.03, \kappa = 30, \zeta = 0.2, \xi = 0.7$ for CNN.

The transmission of model parameters from devices to the BS is measured from 5G networks with the carrier frequency of 2.6 GHz, the subchannel bandwidth of 60 KHz, the slot of 0.5 ms, and the frame of 10 ms. The urban macro (UMa) channel model [44] is utilized for deriving the SNR ratios according to the distance between BS and devices. The transmission power is set to 23 dBm [45], the noise density is -174 dBm/Hz, and the pack loss rate p is set to 10^{-4}. To facilitate performance comparison, we set the number of subchannels and computational resources according to the ML models: 1) 2 subchannels, 2 subchannels, and 20 subchannels for SVM, LR, and CNN, respectively; 2) 0.2 TFLOPS, 0.4 TFLOPS, and 16 TFLOPS for SVM, LR, and CNN, respectively. In such a setup, the number of slots for a local iteration L is 2.

Evaluation Metrics. The target of this paper is to minimize the training time while achieving the same level of test accuracy as benchmark schemes, since the process of federated learning is costly. To evaluate the training time of each scheme, the time to reach the thresholds of both loss value and the variance of FITR is first measured. Six more metrics are also adopted to represent the whole training process: 1) the loss value of the DRL model; 2) the time cost for training an FL model; 3) the global loss value of the FL model; 4) the average test accuracy for all devices; 5)the variance of FITR $\Phi_m(i)$; 6) the FITE $\Gamma(\boldsymbol{\omega})$; and 7) the communication cost, i.e., the accumulated number of transmissions of model parameters to the aggregator for training an FL model. The first two metrics measure the convergence of the trained FL model. The third and the fourth metrics indicate the fairness of the FL model. The fifth metric represents the cost for data transmissions.

Benchmark Approaches. We compare PALORA with the following approaches: 1) PALORA with the first term in the reward function set to zero, i.e., $\alpha = 0$ (denoted as WoL); 2) PALORA with the second term in the reward function set to zero, i.e., $\mu = 0$ (denoted as WoF); 3) PALORA with fixed local iteration number (denoted as FLI); 4) PALORA with fixed bandwidth for each participant (denoted as FR); 5) PALORA with the action amenders removed (denoted as AAR); 6) the method in [15], [19], i.e., participants are selected and allocated with sufficient radio resources for transmissions according to the descending order of SNRs, and the number of local iterations is fixed for all intervals (denoted as DSNR); 7) the method in [20], i.e., participants are selected in a round-robin manner, allocated with sufficient radio resources for transmissions, and set for a fixed number of local iteration for all intervals (denoted as RR); 8) the method in [13], [14], i.e., participants are selected according to the descending order of local loss values, and they are allocated with sufficient radio resources for transmissions and set with a fixed local-iteration number (denoted as LV); 9) the method in [16], [17], i.e., all devices are selected as participants and set with a fixed number of local iteration obtained from analysis (denoted as ALA); 10) randomly selecting participants (denoted as Rand). Approaches 1-5 are the variants of PALORA with the function blocks modified, and the others are existing methods. The numbers of local iterations in approach 6-9 are set to 10, which is corresponding to the setting in their simulations. The number of local iteration for approach 10 is also set to 10.

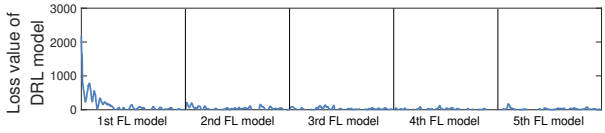


Fig. 8. Loss value of the DRL model in PALORA.

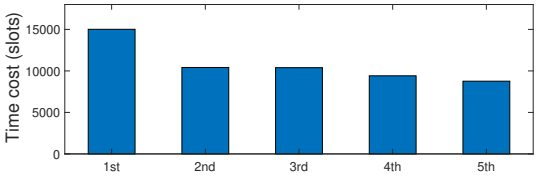


Fig. 9. The time cost for training SVM models.

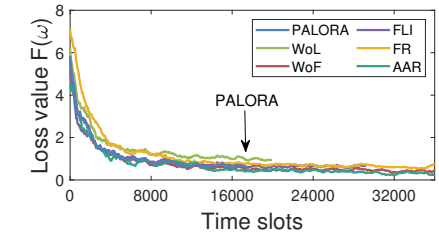
6.2 Simulation Results

The simulation results are organized into four parts: 1) the effectiveness of the DRL model and PALORA-Train, 2) the comparison between PALORA and its variants on SVM to validate each function block, 3) the comparison between PALORA and existing approaches on SVM, and 4) the evaluation on two other models, i.e., LR and CNN.

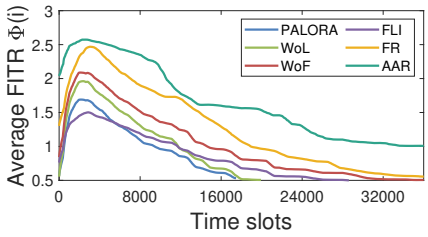
6.2.1 The performance of the DRL model and PALORA-Train

The DRL model and PALORA-Train in the first function block learn to adaptively select participants according to states of devices and wireless channels. To justify their effectiveness, 5 SVM models are trained in the environment that contains device trajectories in 5 continuous days, respectively. The parameters $\boldsymbol{\theta}$ in PALORA-Train are inherited from the training of the previous SVM model when a new SVM model is trained. The loss value of the DRL model and time cost for training 5 SVM models are recorded. According to the loss value in Fig. 8, the DRL model quickly converges during the training of the first SVM model, which indicates that PALORA can quickly find the policy to select participants without wasting too much time and resources. Based on the knowledge learned from the first SVM model, the converge time is much shortened when comes to the second SVM model, and is even negligible for training the following three SVM models. As a result, the training time cost of a latter SVM model decreases compared to the previous one, as shown in Fig. 9, especially the second SVM model. Such a feature further enables PALORA to improve its participant selection policy from training models to reduce training time.

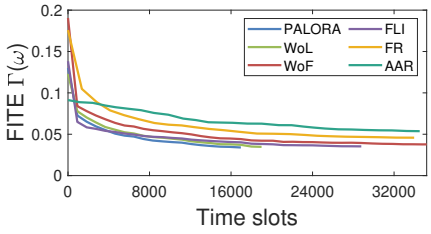
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3148208, IEEE Transactions on Mobile Computing



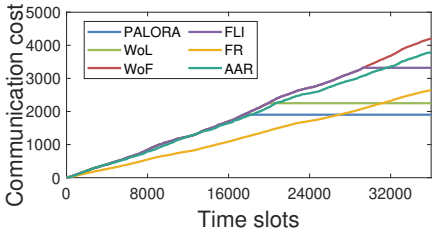
(a) Loss value



(b) FITR

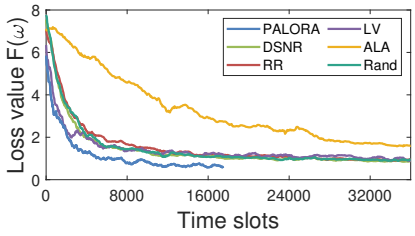


(c) FITE

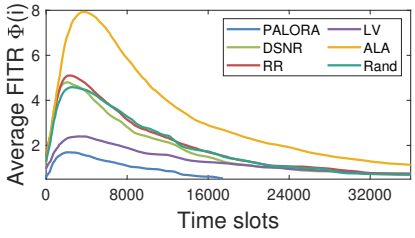


(d) Data transmission amounts

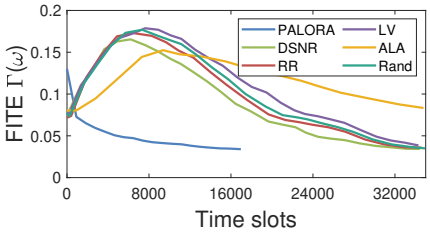
Fig. 10. The performance of PALORA on SVM compared with its variants.



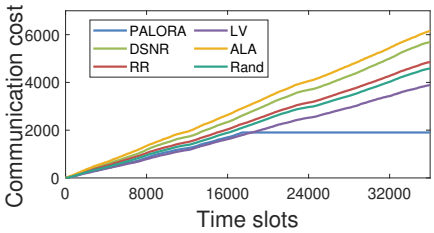
(a) Loss value



(b) FITR



(c) FITE



(d) Data transmission amounts

Fig. 11. The performance of PALORA on SVM compared with existing approaches.

6.2.2 *The comparisons with variants of PALORA*

The performance of PALORA and its variants on the 5th SVM model is shown in Fig. 10. From the loss value recorded in Fig. 10(a) and the average FITR in Fig. 10(b), it can be observed that PALORA is the fastest to achieve both convergence and fairness among all approaches at around 18,000 slots. With a much smaller training time cost, the test accuracy of PALORA is even higher than benchmark approaches according to Table 2. In contrast, by setting the first term in the reward function to zero, WoL has a higher loss value and thus a 5% lower test accuracy than PALORA, as shown in Table 2. This result justifies the effectiveness of the first term, which represents the descent speed of loss value in an interval. Meanwhile, by setting the second term that focuses on fairness to zero, i.e., the fairness metric FITR is not considered for participant selection, WoF cannot complete training within the given training time, since the threshold of model fairness is not reached.

From Fig. 10(a) and Fig. 10(b), it is also obvious that, compared with PALORA, additional 12,000 slots are required by FLI to complete training. This result verifies

TABLE 2
Test Accuracy of PALORA Compared with Its Variants

PALORA	WoL	WoF	FLI	FR	AAR
0.88	0.83	0.88	0.85	0.84	0.86

that improper numbers of local iterations slow down convergence. As for FR, it cannot complete training, which is resulted from the long transmission time in each interval, so that fewer aggregations can be conducted than other approaches within the same duration. Moreover, AAR also cannot complete training because it cannot satisfy the fairness requirement. This result shows the action amender improves the convergence of the FL model, by enhancing the training of the DRL model.

By comparing Fig. 10(b) and Fig. 10(c), it can be found that the results of average FITR ($\Phi(i)$) agree with the results about FITE, i.e., the methods with low average FITR also have low FITE. Such a correspondence validates the equivalence between FITR and FITE, and the effectiveness of selecting devices with high FITR to enhance model fairness. Hence, by maintaining the lowest average FITR, the FL model trained by PALORA is also the fairest. Furthermore,

TABLE 3
Test Accuracy of PALORA Compared with Existing Approaches

PALORA	DSNR	RR	LV	ALA	Rand
0.88	0.84	0.84	0.85	0.84	0.85

from the accumulative number of transmissions shown in Fig. 10(d), it can be observed that lower time cost for training also results in smaller communication cost. For instance, PALORA only requires 2,000 transmissions in the first training period, which is around half of those required by AAR and WoF.

6.2.3 The comparisons with existing approaches

The performance of PALORA is also compared with the existing approaches, which is shown in Fig. 11. From Fig. 11(a) and Fig. 11(b), it can be observed that, given the convergence thresholds where PALORA can quickly complete training, the existing approaches cannot reach the thresholds. Meanwhile, PALORA results in the fairest FL model, as shown in Fig. 11(c), causes the least communication cost, as shown in Fig. 11(d), and the highest test accuracy, as shown in Table 3. These results demonstrate PALORA's superiority, without which the performance deteriorates. For example, simply relying on SNR (DSNR) may result in a repetitive selection of devices with high SNR. With the resulting unfairness, the FL algorithm cannot converge within the given time period. Moreover, the devices with high loss values selected by LV can have low SNR, so the transmission time in each interval can be large, which results in a small number of aggregations within a given time period. Thus, LV fails to complete training in both periods. Similarly, for the method that selects all devices, i.e., ALA, the devices with low SNR can not be avoided. Hence, ALA cannot complete training, even with a fine-tuned number of local iterations.

6.2.4 The performance on LR and CNN

The performance of PALORA and all benchmark approaches are tested on LR and CNN. The results are summarized in Table 4 and 5, respectively. The results similar to those on SVM are observed: For both LR and CNN, PALORA takes the least time and transmits the least amount of data for completing training compared with all the other approaches. Meanwhile, the model trained by PALORA is also the fairest according to the results of FITE. Such a result justifies the effectiveness of PALORA on typical ML models. The effectiveness on CNN further demonstrates that PALORA is can be applied for non-linear models.

7 Discussion

This paper considered the scenario with only one edge server. However, mobile devices outside the service range of an edge server can also need to train an FL model. In this case, multiple edge servers need to be involved. Hierarchical FL frameworks [46], [47] can then be adopted, with the devices at the bottom layer, edge servers at the middle layer, and the cloud (or a selected edge sever) at the top layer. Each edge server utilizes PALORA to train an FL model with the devices within its service range. The edge servers are further considered as the clients for the cloud server where the models aggregated by the edge servers are further aggregated into a global model.

To implement the hierarchical FL framework, two issues need to be considered. First, the timestamps for the edge servers to transmit their models can be different. Therefore, synchronized FL algorithms cannot be adopted at the cloud server. In this case, the asynchronous FL [48], [49] scheme can be adopted, where the transmissions are not required to be simultaneous. Second, devices can move from the service range of one edge server to another during the training process. As a result, these devices cannot upload their model parameters to the previous edge server. To address this issue, device mobility in the near future can be predicted by methods like trajectory prediction [50]. Based on the prediction, these devices with high probabilities of leaving or entering the service range of one edge server within an interval are not selected as participants.

In PALORA, FederatedAverage is adopted as the algorithm for training FL models, because FederatedAverage is widely-adopted in existing platforms like FedML [43]. There also exist other FL algorithms like FederatedSGD [2], FedProx [51], and SCAFFOLD [52]. They all follow the 4-step FL procedure, but do not involve the adjustment of the number of local iterations in each interval. PALORA is also valid for these FL algorithms by simplifying the second function block to return a fixed number of local iterations.

8 Conclusion

In this paper, the problem of training a fair FL model in mobile edge networks was studied. The system model of an FL model trained in mobile edge networks was first built. Based on this model, the criteria of model convergence were analyzed. It was found that the participant selection, the number of local iterations, and the radio resource allocation jointly impact the FL training process. Two metrics for measuring model fairness during training and testing, i.e., FITE and FITR, were designed. Then the joint scheduling problem for training a fair FL model was formulated. By leveraging the theoretical results from the joint scheduling problem, a practical iterative scheduling scheme called PALORA was designed with three function blocks: 1) a pointer-network embedded DRL method to select participants, 2) an estimation method to determine the numbers of local iterations, and 3) a BFS based radio-resource allocation algorithm. Extensive trace-driven simulations validated the effectiveness of PALORA with respect to loss value, test accuracy, model fairness, and cost of parameter transmission.

References

[1] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the Internet of Things with edge computing," *IEEE Netw.*, vol. 32, no. 1, pp. 96–101, 2018.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Int. Conf. Artif. Intell. and Statist. (AISTATS)*. PMLR, 2017, pp. 1273–1282.

[3] K. Wei, J. Li, M. Ding, C. Ma, H. Su, B. Zhang, and H. V. Poor, "User-level privacy-preserving federated learning: Analysis and performance optimization," *IEEE Trans. Mobile Comp. (TMC)*, 2021.

TABLE 4
Simulation Results on LR

	PALORA	WoL	WoF	FLI	FR	AAR	DSNR	RR	LV	ALA	Rand
Training Time (slots)	**5400**	6084	12780	13200	17936	10464	9016	11032	13612	8428	12156
Loss Value	**1.68**	1.90	1.68	1.82	1.79	1.81	1.82	1.82	1.83	1.88	1.87
Test Accuracy	**0.82**	0.78	0.83	0.80	0.79	0.80	0.80	0.79	0.80	0.81	0.82
Average FITR $\Phi_m(i))$	**0.7**	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7
FITE $\Gamma(\boldsymbol{\omega})$	**0.128**	0.146	0.161	0.147	0.146	0.147	0.138	0.143	0.145	0.134	0.148
Communication Cost	**535**	617	1313	1376	1252	1274	1351	1414	1370	1417	1388

TABLE 5
Simulation Results on CNN

	PALORA	WoL	WoF	FLI	FR	AAR	DSNR	RR	LV	ALA	Rand
Training Time (slots)	**6964**	12548	11444	16548	17312	12660	10960	12980	11636	15408	12572
Loss Value	**0.51**	0.54	0.50	0.51	0.52	0.51	0.52	0.52	0.56	0.50	0.52
Test Accuracy	**0.86**	0.81	0.86	0.85	0.83	0.85	0.84	0.84	0.82	0.83	0.84
Average FITR $\Phi_m(i))$	**0.5**	0.5	0.5	0.5	0.5	0.50	0.50	0.5	0.5	0.49	0.50
FITE $\Gamma(\boldsymbol{\omega})$	**0.051**	0.169	0.174	0.144	0.161	0.171	0.148	0.138	0.163	0.101	0.164
Communication Cost	**701**	1290	1208	1760	1204	1294	1619	1615	1226	2542	1440

[4] Q. Wu, X. Chen, Z. Zhou, and J. Zhang, "Fedhome: Cloud-edge based personalized federated learning for in-home health monitoring," *IEEE Trans. Mobile Comp. (TMC)*, 2021.

[5] Y. M. Saputra, D. Nguyen, H. T. Dinh, T. X. Vu, E. Dutkiewicz, and S. Chatzinotas, "Federated learning meets contract theory: Economic-efficiency framework for electric vehicle networks," *IEEE Trans. Mobile Comp. (TMC)*, 2021.

[6] 3GPP TR22.874, "Study on traffic characteristics and performance requirements for AI/ML model transfer in 5GS," Rel. 18, V0.1.0, 2020.

[7] S. Itahara, T. Nishio, Y. Koda, M. Morikura, and K. Yamamoto, "Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data," *IEEE Trans. Mobile Comp. (TMC)*, 2020.

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

[9] Y. Bengio, A. Lodi, and A. Prouvost, "Machine learning for combinatorial optimization: a methodological tour d'horizon," *Eur. J. Oper. Res.*, vol. 290, no. 2, pp. 405–421, 2021.

[10] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," in *Advances in Neural Inf. Process. Syst. (NeurIPS)*, 2015, pp. 2692–2700.

[11] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "Performance optimization of federated learning over wireless networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[12] ——, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Trans. Wireless Commun. (TWC)*, vol. 20, no. 1, pp. 269–283, 2020.

[13] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence time minimization of federated learning over wireless networks," in *Proc. Int. Conf. Commun. (ICC)*. IEEE, 2020, pp. 1–6.

[14] ——, "Convergence time optimization for federated learning over wireless networks," *IEEE Trans. Wireless Commun. (TWC)*, vol. 20, no. 4, pp. 2457–2471, 2020.

[15] W. Shi, S. Zhou, and Z. Niu, "Device scheduling with fast convergence for wireless federated learning," in *Proc. Int. Conf. Commun. (ICC)*. IEEE, 2020, pp. 1–6.

[16] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," *IEEE Trans. Wireless Commun. (TWC)*, vol. 20, no. 3, pp. 1935–1949, 2020.

[17] Z. Yang, M. Chen, W. Saad, C. S. Hong, M. Shikh-Bahaei, H. V. Poor, and S. Cui, "Delay minimization for federated learning over wireless communication networks," in *Int. Conf. Mach. Learn. (ICML) Workshop on Federated Learn.*, 2020.

[18] N. H. Tran, W. Bao, A. Zomaya, N. M. NH, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. Int. Conf. Comput. Commun.(INFOCOM)*. IEEE, 2019, pp. 1387–1395.

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

[20] H. H. Yang, Z. Liu, T. Q. Quek, and H. V. Poor, "Scheduling policies for federated learning in wireless networks," *IEEE Trans. Commun. (TCOMM)*, vol. 68, no. 1, pp. 317–333, 2019.

[21] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," in *Int. Conf. Learn. Representations (ICLR)*, 2019.

[22] W. Du, D. Xu, X. Wu, and H. Tong, "Fairness-aware agnostic federated learning," in *Proc. SIAM Int. Conf. Data Mining (SDM)*. SIAM, 2021, pp. 181–189.

[23] T. Huang, W. Lin, W. Wu, L. He, K. Li, and A. Y. Zomaya, "An efficiency-boosting client selection scheme for federated learning with fairness guarantee," *IEEE Trans. Parallel Distrib. Syst. (TPDS)*, vol. 32, no. 7, pp. 1552–1564, 2021.

[24] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," *Proc. the IEEE*, vol. 107, no. 8, pp. 1738–1762, 2019.

[25] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-iid data," *IEEE Trans. Neural Netw. and Learn. Syst. (TNNLS)*, vol. 31, no. 9, pp. 3400–3413, 2019.

[26] T.-C. Chiu, Y.-Y. Shih, A.-C. Pang, C.-S. Wang, W. Weng, and C.-T. Chou, "Semisupervised distributed learning with non-iid data for aiot service platform," *IEEE Internet of Things J.*, vol. 7, no. 10, pp. 9266–9277, 2020.

[27] X. Lin, J. Wu, J. Li, X. Zheng, and G. Li, "Friend-as-learner: Socially-driven trustworthy and efficient wireless federated edge learning," *IEEE Trans. Mobile Comp. (TMC)*, 2021.

[28] Y. Jiao, P. Wang, D. Niyato, B. Lin, and D. I. Kim, "Toward an automated auction framework for wireless federated learning services market," *IEEE Trans. Mobile Comp. (TMC)*, 2021.

[29] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Commu. Surveys & Tut.*, vol. 20, no. 3, pp. 2429–2453, 2018.

[30] J. Zhang, X. Zhou, T. Ge, and X. Wang, "Joint task scheduling and containerizing for efficient edge computing," *IEEE Trans. Parallel Distrib. Syst. (TPDS)*, vol. 32, no. 8, pp. 2086–2100, 2021.

[31] S. Wang, K. Guan, D. He, G. Li, X. Lin, B. Ai, and Z. Zhong, "Doppler shift and coherence time of 5g vehicular channels at 3.5 ghz," in *IEEE Int. Symp. Antennas and Propagation & USNC/URSI National Radio Science Meeting*. IEEE, 2018, pp. 2005–2006.

[32] 3GPP TS38.321, "NR; medium access control (MAC) protocol specification," Rel. 16, V16.2.1, 2020.

[33] L. V. Kantorovich, *Functional analysis and applied mathematics*. Russian Academy of Sciences, Steklov Mathematical Institute of Russian, 1948.

[34] T. Milne, "Piecewise strong convexity of neural networks," in *Advances in Neural Inf. Process. Syst. (NeurIPS)*, 2019, pp. 12 973–12 983.

[35] J. Zhang, S. Chen, X. Zhou, and X. Wang. Appendix of the paper entitled "joint scheduling of participants, local iterations, and radio resources for fair federated learning over mobile edge

networks". [Online]. Available: https://github.com/Tnnidm/PALORA-Appendix/blob/main/appendix.pdf
[36] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," *Int. Conf. Learn. Representations (ICLR)*, 2018.
[37] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[38] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3-4, pp. 229–256, 1992.
[39] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[40] Z. Wang, V. Bapst, N. Heess, V. Mnih, R. Munos, K. Kavukcuoglu, and N. de Freitas, "Sample efficient actor-critic with experience replay," *Int. Conf. Learn. Representations (ICLR)*, 2016.
[41] J. Zhang, S. Chen, X. Wang, and Y. Zhu, "DeepReserve: Dynamic edge server reservation for connected vehicles with deep reinforcement learning," in *Proc. Int. Conf. Comput. Commun.(INFOCOM)*. IEEE, 2021.
[42] ——, "Dynamic reservation of edge servers via deep reinforcement learning for connected vehicles," *IEEE Transactions on Mobile Computing*, 2021.
[43] C. He, S. Li, J. So, M. Zhang, H. Wang, X. Wang, P. Vepakomma, A. Singh, H. Qiu, L. Shen, P. Zhao, Y. Kang, Y. Liu, R. Raskar, Q. Yang, M. Annavaram, and S. Avestimehr, "FedML: A research library and benchmark for federated machine learning," *arXiv:2007.13518*, 2020.
[44] 3GPP TR38.901, "Study on channel model for frequencies from 0.5 to 100 GHz," Rel. 16, V16.1.0, 2019.
[45] 3GPP TR38.101, "User equipment (UE) radio transmission and reception," Rel. 17, V17.0.0, 2020.
[46] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," in *Int. Conf. Commu. (ICC)*. IEEE, 2020, pp. 1–6.
[47] W. Wu, L. He, W. Lin, and R. Mao, "Accelerating federated learning over reliability-agnostic clients in mobile edge computing systems," *IEEE Trans. Parallel Distrib. Syst. (TPDS)*, vol. 32, no. 7, pp. 1539–1551, 2021.
[48] X. Lian, Y. Huang, Y. Li, and J. Liu, "Asynchronous parallel stochastic gradient for nonconvex optimization," in *Advances in Neural Inf. Process. Syst. (NeurIPS)*, 2015, pp. 2737–2745.
[49] J. Liu, H. Xu, L. Wang, Y. Xu, Q. Chen, J. Huang, and H. Huang, "Adaptive asynchronous federated learning in resource-constrained edge computing," *IEEE Trans. Mobile Comp. (TMC)*, 2021.
[50] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha, "Trafficpredict: Trajectory prediction for heterogeneous traffic-agents," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 6120–6127.
[51] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proc. Mach. Learn. and Syst.*, vol. 2, pp. 429–450, 2020.
[52] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *Int. Conf. Mach. Learn. (ICML)*. PMLR, 2020, pp. 5132–5143.

Jiawei Zhang received the B.S. degree in communication engineering from the Harbin Institute of Technology, Harbin, China, in 2016 and the Ph.D. degree in information and communication engineering from Shanghai Jiao Tong University, Shanghai, China, in 2021. From 2016 to 2021, he was with the Wireless Networking and Artificial Intelligence Lab. His research interests include edge computing and the Internet of Things.

Suhong Chen received the B.S. degree in electronic engineering from the Tsinghua University, Beijing, China, in 2020. He is currently working toward the master's degree with the Wireless Networking and Artificial Intelligence Lab, Shanghai Jiao Tong University, Shanghai, China. His research interests include edge computing and distributed systems.

Xiaochen Zhou received the B.S. degree in telecommunication engineering from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2017. He is currently pursuing the Ph.D. degree with the Wireless Networking and Artificial Intelligence Lab, Shanghai Jiao Tong University. His current research interests include federated learning and on-device machine learning.

Xudong Wang (F'18) received the Ph.D. degree in electrical and computer engineering from the Georgia Institute of Technology in 2003. He is the John Wu and Jane Sun professor of engineering with UM-SJTU Joint Institute, Shanghai Jiao Tong University. He is also an affiliate professor with the Department of Electrical and Computer Engineering, University of Washington. He was a senior research engineer, a senior network architect, and a R&D manager with several companies. His research interests include wireless communication networks, edge computing, and joint communications and sensing. He was the editor of IEEE Transactions on Mobile Computing, IEEE Transactions on Vehicular Technology, Elsevier Ad Hoc Networks, and China Communications. He was the guest editor of several international journals. He was the general chair of 2017 IEEE 5G Summit in Shanghai and the TPC co-chair of the 32nd International Conference on Information Networking. He was the demo co-chair of ACM International Symposium on Mobile Ad Hoc Networking and Computing (ACM MOBIHOC 2006), technical program co-chair of Wireless Internet Conference (WICON) 2007, and general co-chair of WICON 2008. He is an IEEE fellow and was a voting member of IEEE 802.11 and 802.15 Standard Committees.

Yi-Bing Lin (M'96-SM'96-F'03) is Winbond Chair Professor of National Yang Ming Chiao Tung University (NYCU), Chair Professor of China Medical University, National Cheng Kung University and Asia University. During 2014 - 2016, Lin was Deputy Minister, Ministry of Sci ence and Technology, Taiwan. Lin is AAAS Fel low, ACM Fellow, IEEE Fellow, and IET Fellow.