



Learning From Your Neighbours: Mobility-Driven Device-Edge-Cloud Federated Learning

Songli Zhang

Shanghai Jiao Tong University
Shanghai, China
zhang_sl@sjtu.edu.cn

Bingshuai Li

Huawei Noah's Ark Lab
Shanghai, China
libingshuai@huawei.com

Zhenzhe Zheng

Shanghai Jiao Tong University
Shanghai, China
zhengzhenzhe@sjtu.edu.cn

Yunfeng Shao

Huawei Noah's Ark Lab
Beijing, China
shaoyunfeng@huawei.com

Fan Wu

Shanghai Jiao Tong University
Shanghai, China
fwu@cs.sjtu.edu.cn

Guanghai Chen

Shanghai Jiao Tong University
Shanghai, China
gchen@cs.sjtu.edu.cn

ABSTRACT

Federated learning (FL) in large-scale wireless networks is implemented in a hierarchical way by introducing edge servers as relays between the cloud server and devices, where devices are dispersed within multiple clusters coordinated by edges. However, the devices are usually mobile users with unpredictable mobile trajectories, whose effects on the model training process are still less studied. In this work, we propose a new Mobility-Driven federated Learning framework, namely MIDDLE in wireless networks, which can relieve unbalanced and biased model updates by leveraging the new model aggregation opportunities on mobile devices due to their mobility across edges. Specifically, mobile devices can have different models while traversing across edges, and adequately aggregate these models on the device. By theoretical analysis, we can show that this on-device model aggregation can reduce the bias of model updating on edges and cloud, and then accelerate the convergence of model training in FL. Then, we define a model similarity utility to measure the difference in gradient updates among various models, which guides the adaptive on-device model aggregation and in-edge device selection to facilitate the comprehensive information sharing between edges. Extensive experiment results validate that MIDDLE can achieve $1.51 \times - 6.85 \times$ speedup on the model training, compared with the state-of-the-art model training approaches in hierarchical FL.

CCS CONCEPTS

• Human-centered computing → Mobile devices; • Computing methodologies → Machine learning.

KEYWORDS

Federated Learning, Device-Edge-Cloud Cooperation, Device Mobility.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ICPP 2023, August 07–10, 2023, Salt Lake City, UT, USA

© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM. ACM ISBN 979-8-4007-0843-5/23/08...\$15.00
<https://doi.org/10.1145/3605573.3605643>

ACM Reference Format:

Songli Zhang, Zhenzhe Zheng, Fan Wu, Bingshuai Li, Yunfeng Shao, and Guihai Chen. 2023. Learning From Your Neighbours: Mobility-Driven Device-Edge-Cloud Federated Learning. In *52nd International Conference on Parallel Processing (ICPP 2023), August 07–10, 2023, Salt Lake City, UT, USA*. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3605573.3605643>

1 INTRODUCTION

Federated learning (FL) is an emerging privacy-preserving distributed machine learning paradigm [11]. Classical FL algorithms, e.g. FedAvg [23], require devices to perform multiple local training rounds before uploading local models, but the non-independent identical (Non-IID) data across devices can cause gradient drift [21], resulting in the slow convergence of model training. To overcome this drawback, hierarchical federated learning (HFL) [1, 2] is introduced in large-scale wireless networks, which leverages edge servers (e.g., base stations, routers, and switches) as relays between the cloud and devices, and divides the devices into multiple clusters. Multiple edges can frequently aggregate the local models from devices within the associated cluster in a parallel way, and then to relieve gradient drift and also promote communication efficiency [19, 33].

However, HFL still cannot fully escape the curse of Non-IID data distribution within the classical FL. Specifically, the edge model¹ is subject to the data distributions within the edge, which could still be biased from the global distribution, and guide mobile devices to update their local models in a gradient descent direction deviating from the global one [22]. We emphasize that devices in FL are geographically distributed and can move across edges [32], which can be leveraged as a new opportunity to solve the notorious problem of Non-IID data distribution. The natural idea is that edge servers can filter the beneficial data samples from incoming mobile devices to assist in training their edge models. The more profound discovery is that each edge can also use mobile devices as relays to learn complementary information from the other edge models, which is different from the traditional model aggregation in the cloud server. However, it is challenging for the edge to screen valuable data samples and complementary information from the devices with unpredictable mobility patterns, which leads to the

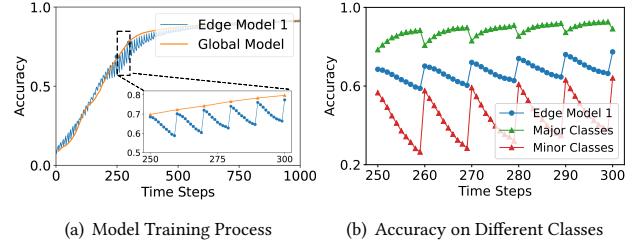
¹The term edge model, local model, and global model refer to the model on edge, device, and cloud, respectively. Edges distribute edge models to the coordinated devices for their local model training, and submit edge models to the cloud for model aggregation, forming the global model.

dynamic devices accessing the edges, instead of the static device sets as in classical FL[4, 30].

In this work, we attempt to exploit device mobility on FL model training to resolve Non-IID data distributions across devices and edges, and then enhance model convergence speed in device-edge-cloud FL. Achieving this goal needs to overcome the following two major challenges. The first and basic challenge comes from dynamic candidate devices within edges, directly brought by the unpredictable devices' mobility patterns. Appropriate device selection strategies in FL have been proven effective in dealing with the problems of Non-IID data distribution and stragglers [29], and the classical device selection approaches usually depend on the historical training performance of devices, such as training loss and testing accuracy[4, 30]. However, the arbitrary mobility patterns can lead to a dynamic and unpredictable candidate device set for each edge to select, and the edge lacks historical information about the training performance of the newly entered devices. Furthermore, mobile devices experience various model training processes within different edges, and thus the historical training performance could not be directly used for the current edge to select devices.

The other thorny challenge is that the edge should reduce the impact of the low-quality models introduced by the newly entered devices when leveraging the mobile devices as a new opportunity for model aggregation. Although the edge can learn the complementary information from local models brought by newly entered devices, which is inherited within the previous edge models, these local models are not all beneficial. On the one hand, each device participates in training intermittently due to the device selection in FL, resulting in the stale local models on some devices. On the other hand, the Non-IID data distribution across edges leads to some edge models being updated in a more biased way from the global [17]. Introducing these local models with low quality into the edge can result in large gradient drift, which makes the edge model and then the global model hard to converge [26].

In this work, jointly considering the above two challenges, we propose Mobility-Driven fer_{ated} Learning framework, namely MIDDLE, to improve the training efficiency of the global model in FL. The key idea behind MIDDLE is to exploit the devices' mobility to mitigate the bias updates of edge models with respect to the direction of the global model. First, we analyze the limitations of the current FL approaches through illustrated experiments, and motivate the potential opportunities for exploiting the mobility of devices. MIDDLE contains two components, including on-device model aggregation and in-edge device selection. Then, we define a similarity utility metric to quantify the divergence of the gradient descent directions between two models. For on-device model aggregation, the mobile device aggregates its carried local model from the previous edge and the current edge model with the weights, which are the similarly utility between these two models. In this way, the mobile device can provide complementary information and promote information exchange across edges by adjusting its local model aggregation. The aggregated local model is then considered as the new starting point of the device's local training. For in-edge device selection, each edge calculates the similarity utility of each local model within the edge to the global model, which is then used as a principle for device selection to fully exploit the dynamic data samples within edges for optimizing the global model training.



(a) Model Training Process (b) Accuracy on Different Classes

Figure 1: The Non-IID data across edges makes the edge models insufficiently learn the minor classes, hindering the convergence of the global model.

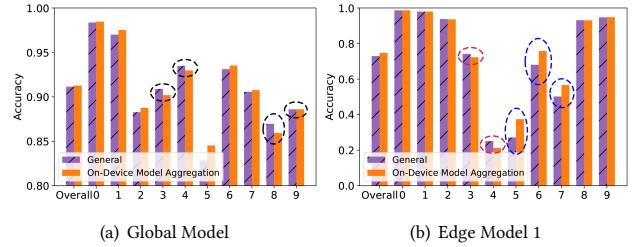


Figure 2: Model accuracy comparison between whether to perform on-device model aggregation.

We summarize our key contributions in this work as follows:

- We explore the impact of devices' mobility on the performance of model training in FL, and propose MIDDLE to leverage the devices' mobility to overcome the notorious problem of Non-IID data distribution, and then accelerate the model convergence.
- We define the similarity utility metric to describe the differences between various models, and promote information exchanging across edges by on-device model aggregation and in-edge device selection in MIDDLE.
- For on-device model aggregation, we provide the theoretical analysis of model convergence bound to show the mobility of devices can correct the bias of the local model training process, which can accelerate the model convergence.
- The extensive data-driven simulations with various learning tasks show that MIDDLE can effectively reduce the time steps to a target accuracy, which outperforms all competitive baselines by $1.51 \times - 6.85 \times$ in terms of model convergence speed, and improve the model accuracy.

2 MOTIVATION

In this section, we conduct some simple experiments to demonstrate our goal of leveraging devices' mobility to accelerate HFL convergence, aiming to answer the following two questions:

Question 1: In HFL, how does Non-IID data distribution across edges hinder the convergence of global model?

Question 2: When devices move across edges, what is the impact of on-device model aggregation on the convergence of global model?

To answer Question 1, we simulate a three-layer HFL with two edges: edge 1 and edge 2, and 50 devices, to train a CNN on MNIST data set with a learning rate of 0.001. The training data is divided

skew across devices and edges in an unbalanced manner. There are 70% of training data labeled as $\{0, 1, 2, 3, 4\}$ (major classes) and 30% of data labeled as $\{5, 6, 7, 8, 9\}$ (minor classes) in edge 1, while the data distribution is opposite in edge 2. The devices perform 10 local SGD in each time step, and update local models to aggregate on the corresponding edge to form the edge model. Then all edge models are aggregated on the cloud to obtain the global model every 10 time steps.

Response to Question 1: As shown in Figure 1(a), although the average accuracy of the global model is steadily improving during the training process, the accuracy of edge model 1 could decrease. In Figure 1(b), we further analyze the accuracy of edge model 1 on the major classes and minor classes. For the major classes, the edge 1 contains more training samples, and the accuracy on the major classes gradually improves. However, since there are fewer data on minor classes in edge 1, the accuracy on minor classes decreases. The Non-IID data across edges leads to edge models updated towards different directions, eventually hindering the convergence of the global model. Furthermore, when the device moves across edges, each edge model will be updated towards different directions according to the associated dynamic data distribution.

To answer Question 2, we conduct a set of similar experiments as Question 1, but each device is assigned the samples of only one class. The training data of two edges are associated with the devices with labels $\{0, 1, 2, 3, 4\}$ and $\{5, 6, 7, 8, 9\}$, respectively. Then, the mobile devices with labels $\{3, 4\}$ move from edge 1 to edge 2, and the devices with labels $\{8, 9\}$ move from edge 2 to edge 1, *i.e.*, the training data within two edges are changed to $\{0, 1, 2, 8, 9\}$ and $\{5, 6, 7, 3, 4\}$, respectively. Two methods are conducted to compare: 1) “General”: the device downloads the edge model from the associated edge, and directly uses the newly downloaded edge model as the starting point of local training; 2) “On-Device Model Aggregation (A Case)”: each moved device simply averages the newly downloaded edge model and its own local model for local training. These two methods then continue training for several steps, and aggregate all local models as the cloud model. Finally, the test accuracy of the cloud model and the edge model 1 is presented for the overall classes and also for each class, in Figure 2.

Response to Question 2: When evaluating the overall model accuracy across all classes, the “On-Device Model Aggregation” shows slight improvements in both the cloud model and edge model 1. However, when examining the accuracy of each individual class of the global model, “On-Device Model Aggregation” performs lower than “General” on several classes (marked with black circles), which is consistent with the exchanged classes $\{3, 4, 8, 9\}$. For the edge model 1 in Figure 2(b), “On-Device Model Aggregation” achieves higher accuracy on classes $\{5, 6, 7\}$ (marked with blue circles), while experiencing lower accuracy on classes $\{3, 4\}$ (marked with red circles), compared with the “General”. It is worth noting that in “On-Device Model Aggregation”, the moved device does not directly adopt the directly downloaded edge model when initializing its starting point of local update, but retains part of the local model inherited from edge 2. Edge 2 is initialized with data samples of class $\{5, 6, 7, 8, 9\}$, and the inherited local models on moved devices can bring the feature of classes $\{5, 6, 7\}$, which is the complementary information for edge 1. Thus, “On-Device Model Aggregation” has a significant improvement on classes $\{5, 6, 7\}$. However, there is a

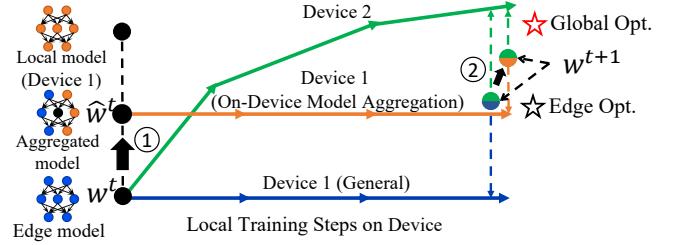


Figure 3: Illustration of the edge model parameter space, the device 1 performs on-device model aggregation, resulting in the change of the local training starting point and further the aggregated edge model.

slight drop on classes {3, 4} in Figure 2(b), since “On-Device Model Aggregation” do not directly use a fully trained edge model 1 on classes {3, 4} as the starting point of training.

We further analyze the impact of “General” and “On-Device Model Aggregation” on the FL training by showing the parameter space of edge models in Figure 3. Supposing devices 1 and 2 participate in model training at the current edge, where device 1 is newly entering. The solid line indicates the device’s local gradient update direction.

General: Under the classical HFL setting [1, 2], devices directly download the edge model w^t from the currently associated edge, and then perform multiple local SGD starting from the downloaded edge model, which is optimized towards their local optima. The new edge model is the average of the updated local models from device 1 and device 2, which is still approximately optimized towards the edge optimum but may also deviate from the global optimum.

On-Device Model Aggregation: Device 1 aggregates the local model and the edge model, and uses this aggregated model \hat{w}^t as the starting point for local model updating. The change of the starting point directly affects its local updating, which further leads to the change of the aggregated edge models and the cloud model. Due to the Non-IID data distributions across edges, the local model of the newly entering device 1, which is inherited from the previous edge, may contain the complementary information for the current edge. Although the aggregated edge model deviates from the edge optimum, it may be closer to the global optimum when introducing the complementary information from other edges, accelerating the convergence of global model.

3 PRELIMINARIES

In this section, we first discuss the background of general FL. Then, we introduce the device-edge-cloud HFL, emphasizing the devices' mobility in wireless networks.

3.1 Federated Learning

In general FL, one cloud server and multiple devices coordinately train a global cloud model for some learning task, such as image classification and speech recognition in IoT applications [17]. The model training process of FL is performed by a set of M mobile devices \mathbb{M} over a series of time steps, denoted by $\mathcal{T} = \{1, \dots, t, \dots, T\}$. Each device $m \in \mathbb{M}$ trains the local model based on its local data samples by running I local updates:

$$w_m^{t+1} = w_m^t - \eta_t \nabla F_m(w_m^t, \xi_m^t), \quad (1)$$

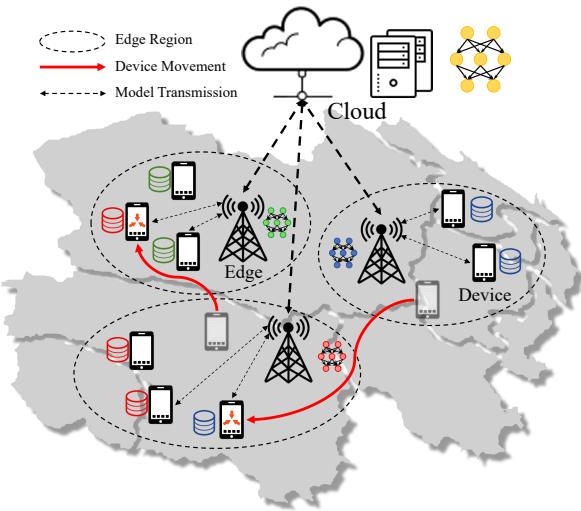


Figure 4: Device-edge-cloud Hierarchical Federated Learning

where w_m^t is the local model of the device m at time step t , $F_m(\cdot)$ is the loss function, ξ_m^t is the randomly selected data samples at time step t , and η_t is the learning rate. Let d_m be the number of data samples on device m . Considering that full device participation is impossible in practice, the cloud server often selects a subset of devices $S(\mathbb{M}) \in \mathbb{M}$ with the size K at each time step for local training. The goal of the cloud server is to solve the following optimization problem:

$$\min_{\mathbf{w}} F(\mathbf{w}) = \sum_{m \in \mathbb{M}} \frac{d_m}{\sum_{m \in \mathbb{M}} d_m} F_m(w_m). \quad (2)$$

3.2 Device Mobility in Federated Learning

In contrast with the vanilla cloud-based FL, device-edge-cloud HFL introduces multiple edges between the cloud server and devices, forming a three-layer FL framework, as illustrated in Figure 4. First, for the bottom device layer, mobile devices are dispersed into different edge regions, and can move across edges. For the middle edge layer, each edge is connected to a wide variety of devices via wireless networks, and to the remote cloud server via wide area networks. Each edge dynamically selects a subset of devices within the edge region to participate in each round of FL model training. The data samples of devices are Non-IID across edges, resulting in different edge models. For the top cloud layer, the cloud server periodically aggregates all edge models to get the global model. In the following discussion, we mainly illustrate the differences between HFL with mobile devices and the vanilla FL.

Mobile Devices: Let \mathbb{N} be the set of all edges, and the devices connected to the edge $n \in \mathbb{N}$ in the time step t form a set \mathbb{M}_n^t . Each device $m \in \mathbb{M}$ updates the local model w_m^t to w_m^{t+1} based on local data samples according to Eq. (1).

The device can move geographically between two sequential time steps $t-1$ and t , and the edges only focus on the newly coming devices from the other edges. Let P_m be the probability of the device m to move across edges, and global mobility probability \bar{P} is defined as the average of P_m .

All devices follow the two basic principles during the training process: 1) each device $m \in \mathbb{M}$ always connects to the nearest edge due to the consideration of communication quality:

$$\mathbb{M}_n^t \cap \mathbb{M}_{n'}^t = \emptyset, \cup_{n \in \mathbb{N}} \mathbb{M}_n^t = \mathbb{M}, \forall t \in \mathcal{T}, \forall n, n' \in \mathbb{N} (n \neq n'), \quad (3)$$

2) any device $m \in \mathbb{M}_n^t$ can complete the entire one-round model training process in time step t , which involves downloading the current edge model, performing local model training and uploading the local model.

We note that our solution is orthogonal to the classic mobility models [16] or mobile trajectory prediction algorithms [27], since we do not need a whole mobile trajectory.

The Edges: At each time step t , each edge $n \in \mathbb{N}$ selects a subset of devices $S(\mathbb{M}_n^t)$ of the size K based on the current available device set \mathbb{M}_n^t to participate in the model training. The edge n distributes the edge model w_n^t to each device $m \in S(\mathbb{M}_n^t)$, receives the updated local model w_m^{t+1} from all the selected devices, and aggregate the edge model w_n^{t+1} for the next time step. After every T_c time steps, the edge communicates with the cloud server.

The Cloud Server: After all edges uploading the edge models, the cloud server aggregates these edge models to obtain the cloud model w_c^{t+1} . Similar to the vanilla FL, the cloud server aims to obtain the optimal cloud model w_c^* by solving the following optimization problem:

$$w_c^* = \underset{w_c}{\operatorname{argmin}} F(w_c) = \frac{\sum_{n \in \mathbb{N}} \sum_{m \in \mathbb{M}_n^t} d_m F_m(w_m)}{\sum_{m \in \mathbb{M}} d_m}, \quad (4)$$

which is the extension of Eq. (2) based on the hierarchical architecture. In this work, we expect to accelerate the convergence of the FL training process with the help of the devices' mobility.

4 DESIGN OF MIDDLE

In this section, we introduce the design of MIDDLE in a top-down manner. We first provide a description of its top-level architecture, and then show the two underlying components: on-device model aggregation and in-edge device selection in details. In on-device model aggregation, we also define the similarity utility metric.

4.1 Overview of MIDDLE

The key idea of mobile-driven FL is to leverage the devices' mobility to accelerate the convergence of FL model training, where edges and devices alternately update edge models and local models on devices in each time step.

We describe the detailed procedure of the training process within a single time step in Algorithm 1. At the beginning of each time step t , each edge $n \in \mathbb{N}$ selects a subset of devices $S(\mathbb{M}_n^t)$ of size K to participate in its edge model training (Line 2). Each edge should select the devices which can promote the edge model updating closer to the global optimum rather than its edge optimum. We describe the detailed procedure of in-edge device selection in Sub-Section 4.3. Each device $m \in \mathbb{M}_n^t$ downloads the edge model w_n^t , and updates this model to a new initial local model \hat{w}_m^t by jointly considering the current edge model and the previous ones (Line 5). We leverage the on-device model aggregation by the incoming devices to learning the “knowledge” from the previous edges, which will be discussed in Sub-Section 4.2 in details. With this new initialized

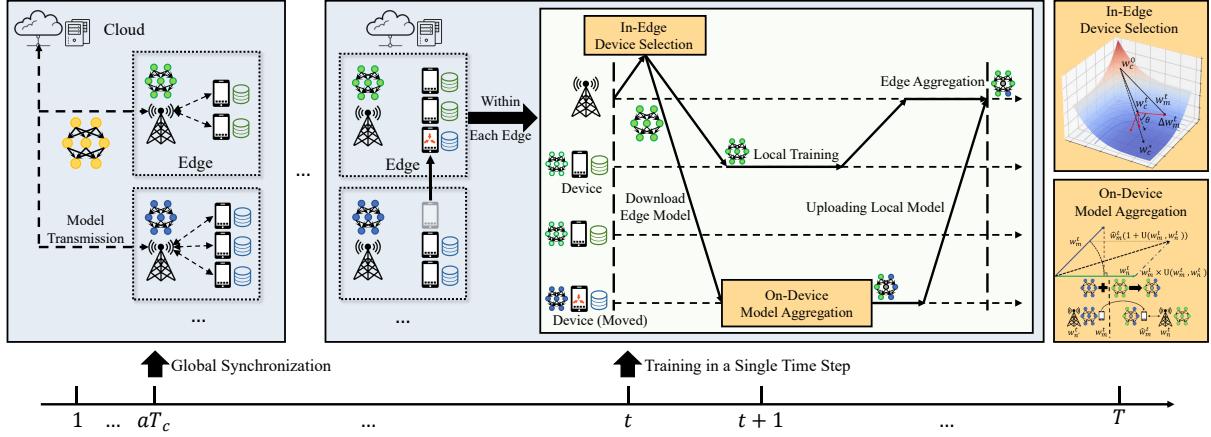


Figure 5: MIDDLE Framework

Algorithm 1: A Single Training Round in MIDDLE

Input: Candidate devices within each edge $\mathbb{M}_n^{t-1} \cup \mathbb{M}_n^t$, global model w_c^t , edge models $\{w_n^t | \forall n \in \mathbb{N}\}$, local models $\{w_m^t | \forall m \in \mathbb{M}\}$;

Output: The updated global, edge and local model w_c^{t+1} , $\{w_n^{t+1} | \forall n \in \mathbb{N}\}$, $\{w_m^{t+1} | \forall m \in \mathbb{M}\}$.

1 **for** each edge $n \in \mathbb{N}$ **do**

2 $S(\mathbb{M}_n^t) \leftarrow \text{InEdge_DeviceSelection}(w_c^t, \{w_m^t | \forall m \in \mathbb{M}_n^t\})$;

3 **for** each device $m \in S(\mathbb{M}_n^t)$ **do**

4 **if** device $m \notin \mathbb{M}_n^{t-1}$ **then**

5 $\hat{w}_m^t \leftarrow \text{OnDevice_ModelAggregation}(w_n^t, w_m^t)$;

6 **else**

7 $\hat{w}_m^t \leftarrow w_n^t$;

8 Update local model: $w_m^{t+1} \leftarrow \hat{w}_m^t - \eta_t \nabla F_m(\hat{w}_m^t, \xi_m^t)$;

9 Edge n aggregates the new edge model as

10 $w_n^{t+1} \leftarrow \sum_{m \in S(\mathbb{M}_n^t)} \frac{d_m}{\sum_{m \in S(\mathbb{M}_n^t)} d_m} w_m^{t+1}$;

11 **if** $t \bmod T_c = 0$ **then**

12 Cloud aggregates the new global model as

13 $w_c^{t+1} \leftarrow \sum_{n \in \mathbb{N}} \frac{\hat{d}_n^t}{\sum_{n \in \mathbb{N}} \hat{d}_n^t} w_n^{t+1}$;

14 **for** each edge $n \in \mathbb{N}$ **do**

15 $w_n^{t+1} \leftarrow w_c^{t+1}$;

16 **for** each device $m \in \mathbb{M}$ **do**

17 $w_m^{t+1} \leftarrow w_c^{t+1}$;

18 **Return** $w_c^{t+1}, w_n^{t+1}, w_m^{t+1}$.

local model \hat{w}_m^t , each participating device $m \in S(\mathbb{M}_n^t)$ performs I local updates based on its local data samples to obtain the new local model w_m^{t+1} , and uploads the new local model to edge n (Line 8):

$$w_m^{t+1} = \hat{w}_m^t - \eta_t \nabla F_m(\hat{w}_m^t, \xi_m^t). \quad (5)$$

After receiving local models from all the selected devices $m \in S(\mathbb{M}_n^t)$, the edge n aggregates these local models using the weight

of the size of data samples per device as in FedAvg:

$$w_n^{t+1} = \sum_{m \in S(\mathbb{M}_n^t)} \frac{d_m}{\sum_{m \in S(\mathbb{M}_n^t)} d_m} w_m^{t+1}. \quad (6)$$

After every T_c time steps, i.e., $t \bmod T_c = 0$, all edges upload the new edge models to the cloud server (Lines 11). The cloud server aggregates these edge models to a new global model w_c^{t+1} :

$$w_c^{t+1} = \sum_{n \in \mathbb{N}} \frac{\hat{d}_n^t}{\sum_{n \in \mathbb{N}} \hat{d}_n^t} w_n^{t+1}, \quad (7)$$

where the weight of each edge model \hat{d}_n^t is the number of data samples on devices participating in the training process of edge n , and $\hat{d}_n^t = \sum_{t'=t-T_c}^t \sum_{m \in S(\mathbb{M}_n^t)} d_m$. Then, the cloud distributes w_c^{t+1} to edges and devices to update the edge models and local models. Finally, the cloud, edges, and devices repeat the above procedure until the preset number of time steps is reached.

Figure 5 also summarizes the MIDDLE framework over the entire training process, and the process of “on-device aggregation” and “in-edge device selection” are shown on the right. The cloud communicates with edges to perform global synchronization every T_c time steps. In each single time step, each edge performs in-edge device selection and distributes its edge model to each device. Then, the moved device performs on-device model aggregation, which aggregates the downloaded edge model with its local model to a new initial local model \hat{w}_m^t as the starting point of its local training.

4.2 On-Device Model Aggregation

To efficiently utilize the various “knowledge” in different edge models, we propose on-device model aggregation to relieve the unbalanced updates to accelerate the convergence of the global model in FL. In traditional multi-level FL, “knowledge” sharing across edges relies on model aggregation on the cloud. However, the incoming devices bring their own local models inherited from the previous edges, which may contain the “knowledge” missing from the current edge model due to the biased update of the model caused by the Non-IID data distribution across edges. The current edge desires this complementary “knowledge” to reduce the biased updating of the edge model for accelerating the convergence of global model

training. For a mobile device m , by aggregating the edge model w_n^t from the current edge and the local model w_m^t inherited from the previous edges, we expect to introduce complementary knowledge to the current edge model training process. However, the local model w_m^t could be stale due to the device participating in model training aperiodically in FL. Furthermore, the inherited local model w_m^t could vary greatly in the gradient descent direction due to the Non-IID data distribution across edges. Thus, simply aggregating the models, which are stale or with large differences, will introduce additional noise into the current edge model training, making the FL model training difficult to converge[26].

With the above consideration, we adopt the cosine similarity between the local model w_m^t and the current edge model w_n^t to measure the utility of gradient updates from different parameter models [10, 35]. Then, we define the similarity utility $U(\cdot)$, and $U(w_m^t, w_n^t)$ is the similarity utility between the previous local model w_m^t and the edge model w_n^t :

$$U(w_m^t, w_n^t) = \text{Max} \left(\frac{\langle w_m^t, w_n^t \rangle}{\|w_m^t\| \|w_n^t\|}, 0 \right). \quad (8)$$

The similarity utility is set to zero when the cosine similarity score is lower than 0 to avoid blind aggregation introducing noise. Then, when device m moves across the edges and participates in the current edge training process, device m calculates the similarity utility $U(w_m^t, w_n^t)$ between the previous local model w_m^t and the downloaded edge model w_n^t , uses it as the weight of the model aggregation on device, and gets a new initial local model \hat{w}_m^t :

$$\begin{aligned} \hat{w}_m^t &= \text{OnDevice_ModelAggregation}(w_n^t, w_m^t) \\ &= \frac{1}{1 + U(w_m^t, w_n^t)} w_n^t + \frac{U(w_m^t, w_n^t)}{1 + U(w_m^t, w_n^t)} w_m^t. \end{aligned} \quad (9)$$

In this way, the new initial local model \hat{w}_m^t is still dominated by the current edge model, but also introducing the complementary knowledge of other edge models.

4.3 In-Edge Device Selection

The devices' mobility makes each edge cover different devices under various time steps, and each edge needs to select devices under an uncertain device set at each time step. The historical training experiences of the newly entered devices are based on the previous edge models, and thus cannot be directly used as the criteria for the device selection in the current edge. We need to design a new model quality metric, jointly considering local data privacy and global optimization objective.

Considering the global optimization objective, the edge should select the devices that can regulate the edge model's update direction closer to the global optimum w_c^* . Data privacy requirement in FL makes it fail to directly use the device's data distribution to select devices. Thus, we turn to the available local model parameters for device selection. Let the accumulative updating of local model w_m^t with respective to the global w_c^t be Δw_m^t :

$$\Delta w_m^t = w_m^t - w_c^t, \quad (10)$$

which is optimized towards the local optimum of the device m . We can calculate the similarity utility $U(w_c^*, \Delta w_m^t)$ between w_c^* and Δw_m^t as the criterion for device selection within the edge. However, the optimal cloud model parameters w_c^* cannot be obtained during

the training process. Since the update direction of w_c^t in each iteration attempts to approach the optimal global model w_c^* due to Eq. (4), We can approximate $U(w_c^*, \Delta w_m^t)$ by:

$$U(w_c^*, \Delta w_m^t) \approx U(w_c^t, \Delta w_m^t). \quad (11)$$

Finally, to avoid getting stuck in the local optimum, the edge should select the devices with data samples which are not sufficiently learned by the global cloud model[14, 18], meaning the local model with less similarity to the cloud model should be assigned with a high probability to select. With the above consideration, the k devices are selected from the candidate devices \mathbb{M}_n^t to participate in the edge model training:

$$\begin{aligned} S(\mathbb{M}_n^t) &= \text{InEdge_DeviceSelection}(w_c^t, \{w_m^t, \forall m \in \mathbb{M}_n^t\}) \\ &= \text{TOPK}(\{-U(w_c^*, \Delta w_m^t), \forall m \in \mathbb{M}_n^t\}), \end{aligned} \quad (12)$$

where $\text{TOPK}(\cdot)$ is the function to output a set of items with the K highest values.

5 THEORETICAL ANALYSIS

In this section, we provide an analysis of the convergence bound on the FL under devices' mobility to demonstrate that on-device model aggregation can relieve unbalanced updates across edges.

We stick to the following assumptions, which are widely used in the literature.

ASSUMPTION 1. $F_m(w_m)$ is β -Lipschitz smoothness for each device $m \in \mathbb{M}$, i.e., $F_m(w_m) \leq F_m(w'_m) + (w_m - w'_m)\nabla F_m(w'_m) + \frac{\beta}{2}\|w_m - w'_m\|^2$ for any two parameter model w_m and w'_m .

ASSUMPTION 2. $F_m(w_m)$ is μ -strongly convex for each device $m \in \mathbb{M}$, i.e., $F_m(w_m) \geq F_m(w'_m) + (w_m - w'_m)\nabla F_m(w'_m) + \frac{\mu}{2}\|w_m - w'_m\|^2$ for any two parameter model w_m and w'_m .

ASSUMPTION 3. Let ξ_m^t be a randomly selected data samples from the device m at time step t , the variance of stochastic gradients in each device $m \in \mathbb{M}$ is bounded, i.e., $\mathbb{E}\|\nabla F_m(w_m^t, \xi_m^t) - \nabla F_m(w_m^t)\| \leq c_m^2$.

ASSUMPTION 4. The expected squared norm of stochastic gradients is uniformly bounded, i.e., $\mathbb{E}\|\nabla F_m(w_m^t, \xi_m^t)\|^2 \leq G^2$ for $\forall m \in \mathbb{M}$ and $\forall t \in \mathcal{T}$.

Assumptions 1 and 2 are standard, which are satisfied in the linear regression, logistic regression, and softmax classifier. Assumptions 3 and 4 are typical in the general theoretical analysis of FL algorithms. Furthermore, to simplify the analysis, all devices are assumed to participate in the training. An additional virtual variable \bar{w}^{t+1} is introduced to represent the aggregation of local models after time step $t+1$:

$$\bar{w}^{t+1} = \sum_{m \in \mathbb{M}} \frac{d_m}{\sum_{m \in \mathbb{M}} d_m} w_m^{t+1}. \quad (13)$$

\bar{w}^{t+1} is equal to w_c^{t+1} at the time step when the edge communicates with the cloud server. From the β -Lipschitz smoothness of loss function F , we have:

$$\begin{aligned} &F(w_c^{T+1}) - F(w_c^*) \\ &\leq (w_c^{T+1} - w_c^*)\nabla F(w_c^*) + \frac{\beta}{2}\|w_c^{T+1} - w_c^*\|^2, \end{aligned} \quad (14)$$

w_c^{T+1} is the final cloud model on the cloud server. By taking expectation at both sides:

$$\begin{aligned} \mathbb{E}[F(w_c^{T+1})] - F(w_c^*) &\leq \frac{\beta}{2} \mathbb{E}[\|w_c^{T+1} - w_c^*\|^2] \\ &= \frac{\beta}{2} \mathbb{E}[\|\bar{w}^{T+1} - w_c^*\|^2]. \end{aligned} \quad (15)$$

LEMMA 1. Under Assumptions 1 and 2, if $\eta_t \leq \frac{1}{4\bar{P}}$, it has:

$$\begin{aligned} &\mathbb{E}[\|\bar{w}^{t+1} - w_c^*\|^2] \\ &\leq (1 - \eta_t \mu) \mathbb{E}[\|\bar{w}^t - w_c^*\|^2] + \eta_t^2 \mathbb{E}[\|g^t - \bar{g}^t\|^2] \\ &+ 6\beta\eta_t^2\Gamma + 2\mathbb{E}\left[\sum_{m \in \mathbb{M}} h_m \|\bar{w}^t - w_m^t\|^2\right], \end{aligned} \quad (16)$$

where $g^t = \sum_{m \in \mathbb{M}} h_m \nabla F_m(\hat{w}_m^t, \xi_m^t)$, $\bar{g}^t = \sum_{m \in \mathbb{M}} h_m \nabla F_m(\hat{w}_m^t)$, $h_m = \frac{d_m}{\sum_{m \in \mathbb{M}} d_m}$, and $\Gamma = F^* - \sum_{m \in \mathbb{M}} h_m F_m^*$. F^* and F_m^* are the minimum value of F and F_m , respectively.

PROOF. Please see the Lemma 1 in [8, 20] for the proof. We omit the similar proof due to space limitation. ■

To facilitate the analysis, we set a fixed on-device model aggregation coefficient α , instead of the dynamic value in the on-device model aggregation, i.e., $\hat{w}_m^t = (1 - \alpha)w_m^t + \alpha w_m^t$.

THEOREM 1. Under Assumptions 1 to 4 and assuming $\alpha \in (0, 1)$, by selecting $P_m = \bar{P}(\forall m \in \mathbb{M})$, $\bar{P} \in (0, 1]$, $\gamma = \max\{\frac{8\beta}{\mu}, 1\}$, and the learning rate $\eta_t = \frac{2}{\mu(\gamma+1)}$, the convergence bound of MIDDLE at time step t with full device participation satisfies:

$$\begin{aligned} &\mathbb{E}[F(w_c^{T+1})] - F(w_c^*) \\ &\leq \frac{\beta}{\gamma+T+1} \left(\frac{2B}{\mu^2} + \frac{(\gamma+1)}{2} \mathbb{E}[\|\bar{w}^{(1)} - w^*\|^2] \right) + \frac{8\beta I^2 G^2}{\mu^2 \gamma^2 \alpha (1 - \alpha) \bar{P}}, \end{aligned} \quad (17)$$

where

$$B = \sum_{m \in \mathbb{M}} h_m^2 \sigma_m^2 + 6\beta\Gamma. \quad (18)$$

PROOF SKETCH. We now give an outline of the proof for Theorem 1. According to Lemma 1 and Eq. (15), we can prove Theorem 1 by proving the upper bound of $\mathbb{E}[\|\bar{w}^{t+1} - w_c^*\|^2]$. Specifically, we first determine upper bounds for $\mathbb{E}[\sum_{m \in \mathbb{M}} h_m \|\bar{w}^t - w_m^t\|^2]$ and $\eta_t^2 \mathbb{E}[\|g^t - \bar{g}^t\|^2]$. For $\mathbb{E}[\sum_{m \in \mathbb{M}} h_m \|\bar{w}^t - w_m^t\|^2]$, it can be expanded into:

$$\begin{aligned} &\mathbb{E}\left[\sum_{m \in \mathbb{M}} h_m \|\bar{w}^t - w_m^t\|^2\right] \\ &\leq \mathbb{E}\left[\sum_{m \in \mathbb{M}} h_m \|w_m^t - \hat{w}_m^{t-1}\|^2\right] + \mathbb{E}\left[\sum_{m \in \mathbb{M}} h_m \|\hat{w}_m^{t-1} - \bar{w}^{t-1}\|^2\right], \end{aligned} \quad (19)$$

which can be proofed from $\mathbb{E}[\|X - \mathbb{E}[X]\|^2] \leq \mathbb{E}\|X\|^2$ and $\mathbb{E}[\hat{w}_m^t] = \bar{w}^t$. $\mathbb{E}\left[\sum_{m \in \mathbb{M}} h_m \|w_m^t - \hat{w}_m^{t-1}\|^2\right]$ is general in typical federated learning convergence analysis, which indicates the local updating of device m . Moreover, $\mathbb{E}\left[\sum_{m \in \mathbb{M}} h_m \|\hat{w}_m^{t-1} - \bar{w}^{t-1}\|^2\right]$ is unique, which indicates the divergence between the changed local updating starting point and the global average after the on-device model aggregation and can be bounded with fixed value α and global mobility \bar{P} . The proof of $\eta_t^2 \mathbb{E}[\|g^t - \bar{g}^t\|^2]$ is also typical [8, 20]. Finally, Theorem 1 can be proofed by mathematical induction. ■

REMARK 1. We next investigate how the devices' mobility affects the convergence bound of FL. By taking the first-order derivative of $(\mathbb{E}[F(w_c^{T+1})] - F(w_c^*))$ over the global mobility \bar{P} , we have

$$\frac{\partial (\mathbb{E}[F(w_c^{T+1})] - F(w_c^*))}{\partial \bar{P}} = -\frac{8\beta I^2 G^2}{\mu^2 \gamma^2 \alpha (1 - \alpha) \bar{P}^2}. \quad (20)$$

Thus, it can be observed that $\frac{\partial (\mathbb{E}[F(w_c^{T+1})] - F(w_c^*))}{\partial \bar{P}} < 0$ over $\bar{P} \in (0, 1]$ and $\alpha \in (0, 1)$. These results show that the error between the final cloud model w_c^{T+1} and the optimal cloud model w_c^* can be always reduced under any global mobility \bar{P} . Moreover, according to the first-order derivative of $(\mathbb{E}[F(w_c^{T+1})] - F(w_c^*))$, this model error can gradually decrease with the increase of the global mobility \bar{P} .

6 EVALUATION RESULTS

In this section, we evaluate MIDDLE through extensive numerical experiments. We first introduce the experiment settings, and then provide the experimental results with corresponding analysis.

6.1 Experiment Settings

6.1.1 **Dataset.** We aim at two typical applications in mobile computing, i.e., image classification and speech recognition. For image classification tasks, we adopt three open source datasets, including MNIST [15], EMNIST [3] and CIFAR10 [13]. In MNIST and CIFAR10, there are 10 classes of grayscale and color images, respectively. In EMNIST, the “Letters” track containing 26 classes of grayscale images from letters ‘A’ to ‘Z’ is used as the learning task. For the speech recognition, we use the open source dataset SpeechCommands [34] to detect the voice ‘zero’ to ‘nine’. All training tasks are divided into training and test sets. We use the ONE simulator [12] to generate the traces of mobile devices, which is a common simulator to generate user movement traces using different mobility models. All devices move between edges with different probabilities, and the expectation meets the value of global mobility \bar{P} .

6.1.2 **Parameter Settings.** In MIDDLE, we simulate 10 edges and 100 mobile devices. We assume 50% of the devices participating in training at each time step, and the number of selected devices K within each edge is set as 5. The local data samples in each device are supposed to have large Non-IID distribution: there exists a major class for the device' data samples (more than 80% of all samples). The local training epochs I and the time step interval T_c of communication between cloud and edge are both set as 10. The expected mobility probability \bar{P} is set as 0.5. The MNIST and EMNIST are trained on the convolutional neural network (CNN) with 2 convolutional layers and 2 fully connected layers. The CIFAR10 and SpeechCommands are trained on the convolutional neural network with 3 convolutional layers and 2 fully connected layers. For three image classification tasks, the optimizer is SGD, and the stochastic gradient descent with momentum is used for training, which has an initial learning rate of 0.01 and a momentum term of 0.9. For the speech recognition task, the optimizer is Adam, and the learning rate is 0.001. The convergence speed of different algorithms is reflected in the time steps of reaching the target accuracy, which are set as 0.95, 0.80, 0.55, and 0.85 for MNIST, EMNIST, CIFAR10, and SpeechCommands, respectively.

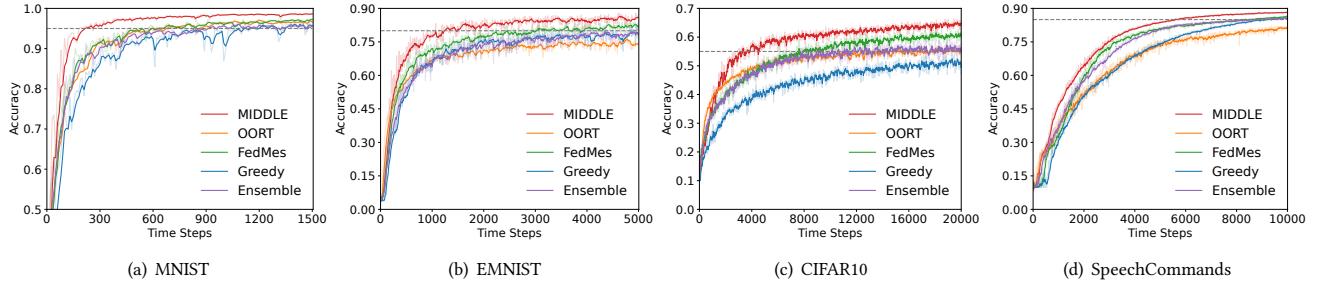
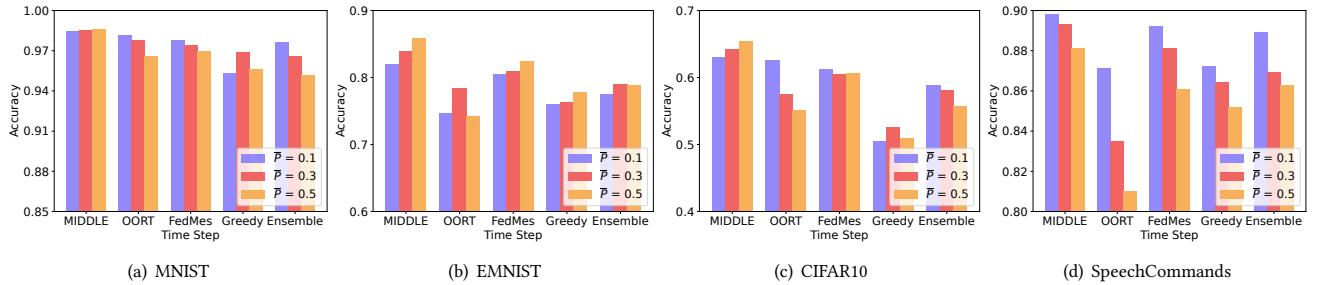


Figure 6: Time-to-accuracy performance over all learning tasks.

Figure 7: Final accuracy of global models on various global mobility \bar{P} .

6.1.3 **Baselines.** We compare MIDDLE with four existing baselines, which do not specifically deal with the devices' mobility. Some adaptive adjustments are made to these baselines to be applied to our problem, and the details are as follows:

OORT: The OORT is the latest device selection strategy in [14, 18]. The system utilities of devices are set as the same, and each edge selects the devices with the top K highest statistical utilities for model training. It is without on-device model aggregation.

FedMes: The FedMes utilizes the devices located in the overlap of two edges to accelerate the model convergence of FL, where each two edge models are aggregated on the devices in an averaged way [8]. In the experiments, devices moving across edges are regarded as the overlapped devices. FedMes adopts the random device selection for edges.

Greedy: When a device moves across edges, it greedily keeps the previous local model as the initial local model for local updates. As in OORT, Greedy selects the devices with the K highest statistical utility to participate in the training.

Ensemble: An ensemble approach combines the OORT and FedMes, which aggregates on-device in an averaged way and selects the devices with the K highest statistical utility.

To show the experiment results more clearly, all results are smoothed and presented by their averages, and the shades are the actual experimental results.

6.2 Experimental Results and Analysis

6.2.1 *Overall Performance.* First, a set of experiments is conducted to verify the performance of MIDDLE over various training tasks. In Figure 6, the MIDDLE outperforms all the baselines over both the model accuracy and the convergence speed in all learning tasks. It shows that MIDDLE can effectively improve model accuracy and speed up model convergence through dynamic device selection within edge and model sharing between edges. However, in the

early stage of training, OORT has a faster convergence speed than the other approaches. Because each edge in the OORT does not introduce the parameter models of other edges, OORT avoids introducing the noise brought by model aggregation on devices. By observing the results of FedMes, Ensemble, and Greedy, they all have higher final model accuracy than OORT in Figure 6(b) and Figure 6(d). This is because EMNIST and SpeechCommands are two more complex learning tasks, which contain more classes and more complex input data, and the aggregation of different models on devices can make the best of the advantages of introducing complementary knowledge, and improve the final model accuracy. It is also shown that introducing other edge models can assist each edge in learning the global data distribution, which can effectively avoid the local optimum of edge models due to the Non-IID data distribution across edges. Furthermore, with the learning task becoming complex, all other approaches achieve a higher final model accuracy than OORT on the EMNIST, CIFAR10, and SpeechCommands, which is shown in Figure 6(b) to Figure 6(d). In Figure 6(a) and Figure 6(c), the time-to-accuracy curve of Greedy shows certain oscillations. This suggests that Greedy approach could introduce too much noise from low-quality local models, leading to gradient drift. The results show that MIDDLE can effectively learn from abundant data samples and the complementary knowledge from the other edge models by the devices' mobility, which can reduce the time steps to the target accuracy and outperform all competitive baselines by $1.51 \times - 6.85 \times$ in terms of model convergence speed while improving the model accuracy.

6.2.2 *Performance of the various global mobility \bar{P} .* We compare the results of MIDDLE on different global mobility \bar{P} , in Figure 7. First, MIDDLE outperforms the other baselines for various global mobility. From our theoretical analysis in Remark 1, the final global model would be closer to the optimal one with the increase of \bar{P} . However,

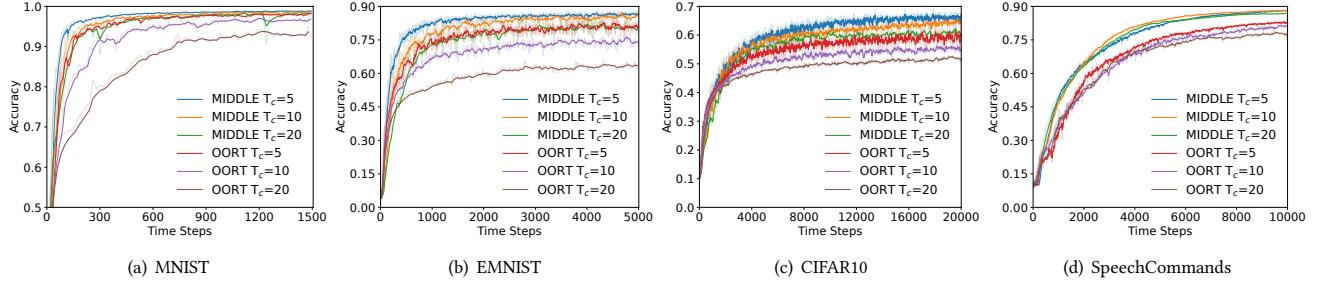


Figure 8: Effects of the different time step interval T_c of communication between cloud and edges.

only in Figure 7(a) to Figure 7(c), the final model accuracy of MIDDLE increases with the increase of global mobility \bar{P} . But for most baselines, the experimental results do not follow our theoretical analysis. This is because the device mobility will cause the dynamic edge optima, resulting in certain oscillations in the edge update direction. Moreover, our theoretical analysis focuses on proving the on-device model aggregation leading to the starting point of the device’s local updating closer to the global model, assuming all devices’ participation. In FL training, all devices’ participation is unrealistic, so an in-edge device selection strategy is necessary for MIDDLE, which creates a bit of deviation between the theoretical analysis and experiment results. The input of the SpeechCommands is long sparse vectors, which presents challenges in training and leads to larger discrepancies between experimental results and expectations. In Figure 7, we can observe an approximate conclusion that the final model accuracy of all methods presents a trend of rising first and then falling. When the peak comes too early or too late, the final model accuracy shows continuously falling or rising. Moreover, in Figure 7(a) to Figure 7(d), with the global mobility \bar{P} increasing, the improvement of MIDDLE in model accuracy is more obvious, which shows that MIDDLE has good robustness and can reduce noise interference.

6.2.3 Performance under the different edge-cloud communication interval T_c . As shown in Figure 8, we expect to illustrate further the importance of model sharing between edges by comparing the performance on different time step intervals T_c of the communication interval between the cloud and edges. Specifically, we compare MIDDLE with OORT, which is a baseline introducing no knowledge from the other edges. We can observe that OORT has a larger reduction on the final model accuracy with the increase of T_c , especially on complex training tasks in Figure 8(b) to Figure 8(d). Although OORT can strategically select devices within each edge, the Non-IID data distribution across edges causes the edge models to update in different directions, which makes the cloud model hard to converge to a stable final global model. After the cloud model has converged, the time-to-accuracy curves of the MIDDLE have smaller oscillations, indicating that its edge models have lower differences. At the same time, in Figure 8(b), the model accuracy on EMNIST drops more obviously, with the increase of T_c . This is because the EMNIST has more training sample classes, resulting in larger differences between edge models. With the above results, we can conclude that MIDDLE can make edge models effectively benefit from local models on mobile devices by exchanging complementary information.

7 RELATED WORK

As a typical HFL framework, device-edge-cloud FL is implemented to improve communication efficiency in wireless networks [33, 37], in which the master aggregator dynamically schedules multiple aggregators to scale with the number of devices and update size in training [1]. Castiglia et al. [2] first proposed multi-level SGD and analyzed the convergence of multi-level SGD. Wang et al. [33] analyzed the impact of the number of edges on the communication resource and convergence speed of FL training. Zhong et al. [37] proposed parallelizing federated learning to speed up learning efficiency. However, edges dividing all devices into subsets bring more complex heterogeneity on both system overhead and data distribution. Yang et al. [36] and Feng et al. [7] focused on the resource allocation in HFL, which formulated the optimization problem and solved it. They ignore the imbalance updating of edge models caused by Non-IID data distribution across devices and edges, which hinders the convergence of the global model and requires longer training iterations to reach the target accuracy.

Each edge coordinates the training of devices within the edge independently, and existing literature has made efforts to study how the interaction between devices and edges affects the convergence of the global model. Wang et al. [31] defined the concepts of upward divergence and downward divergence, which represent the model updating deviation between cloud and edges, edges and devices, respectively, and pointed out that reducing the heterogeneity between edges is more conducive to convergence. Qu et al. [28] and Han et al. [8] proposed to leverage the devices in the overlapping edge areas acting as bridges connecting different edges, and the devices in the overlapping areas download models from multiple edges and aggregate these models on the devices. However, the same device participates in the training process of different edges simultaneously, which causes these devices’ local models to be repeatedly calculated by multiple edges and actually results in the biased updating of the global model. Hu et al. [9] and Li et al. [19] divided devices into multiple clusters in advance to meet the requirements of system scalability and various heterogeneity. Ng et al. [24, 25] designed incentive mechanisms for HFL to encourage devices to participate in the training of edges, which were more conducive to cloud model convergence. However, these pre-divide and subjective incentive methods guide devices to participate in the training process of different edges, ignoring the objective geographical distribution of devices. Feng et al. [5, 6] studied the uncertain mobility of devices, but did not consider how to leverage the mobile devices to accelerate the convergence of the global model.

8 CONCLUSION

In this work, to deal with the Non-IID data distribution across devices and edges in device-edge-cloud FL, we propose a mobility-driven FL, namely MIDDLE, which utilizes the characteristic of unpredictable mobility of devices and the differences between models to accelerate convergence. Specifically, MIDDLE contains two components, on-device model aggregation and in-edge device selection. We define a similarity utility metric to measure the differences in gradient descent directions between different models as the basic component for on-device model aggregation and in-edge device selection, which are used to learn complementary information across edges and process dynamic training samples within edges, respectively. Theoretical analysis demonstrates that MIDDLE can effectively correct the bias of the local training process through on-device model aggregation to accelerate global training. Finally, extensive evaluation results confirm that MIDDLE can effectively improve the convergence speed while improving the model accuracy.

ACKNOWLEDGMENTS

This work was supported in part by National Key R&D Program of China No. 2020YFB1707900, in part by China NSF grant No. 62132018, U2268204, 62272307, 61902248, 61972254, 61972252, 6202504, 62072303, in part by Shanghai Science and Technology fund 20PJ1407900, in part by Huawei Noah’s Ark Lab NetMIND Research Team. The opinions, findings, conclusions, and recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies or the government. Zhenzhe Zheng is the corresponding author.

REFERENCES

- [1] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečný, Stefano Mazzocchi, Brendan McMahan, et al. 2019. Towards federated learning at scale: System design. *Proceedings of MLSys* 1 (2019), 374–388.
- [2] Timothy Castiglia, Anirban Das, and Stacy Patterson. 2021. Multi-level local SGD: Distributed SGD for heterogeneous hierarchical networks. In *Proceedings of ICLR*.
- [3] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. 2017. EMNIST: Extending MNIST to handwritten letters. In *Proceedings of IJCNN*. 2921–2926.
- [4] Yongheng Deng, Feng Lyu, Ju Ren, Yi-Chao Chen, Peng Yang, Yuezhi Zhou, and Yaoxue Zhang. 2021. Fair: Quality-aware federated learning with precise user incentive and model aggregation. In *Proceedings of INFOCOM*. 1–10.
- [5] Chenyuan Feng, Howard H Yang, Deshun Hu, Tony QS Quek, Zhiwei Zhao, and Geyong Min. 2021. Federated Learning with User Mobility in Hierarchical Wireless Networks. In *Proceedings of GLOBECOM*. 01–06.
- [6] Chenyuan Feng, Howard H. Yang, Deshun Hu, Zhiwei Zhao, Tony Q. S. Quek, and Geyong Min. 2022. Mobility-Aware Cluster Federated Learning in Hierarchical Wireless Networks. *IEEE Transactions on Wireless Communications* 21, 10 (2022), 8441–8458.
- [7] Jie Feng, Lei Liu, Qingqi Pei, and Kebin Li. 2021. Min-max cost optimization for efficient hierarchical federated learning in wireless edge networks. *IEEE Transactions on Parallel and Distributed Systems* 33, 11 (2021), 2687–2700.
- [8] Dong-Jun Han, Minseok Choi, Jungwuk Park, and Jaekyun Moon. 2021. FedMes: Speeding Up Federated Learning With Multiple Edge Servers. *IEEE Journal on Selected Areas in Communications* 39, 12 (2021), 3870–3885.
- [9] Chuang Hu, Huang Huang Liang, Xiao Ming Han, Bo An Liu, Da Zhao Cheng, and Dan Wang. 2022. Spread: Decentralized Model Aggregation for Scalable Federated Learning. In *Proceedings of ICPP*. 1–12.
- [10] Wonyong Jeong and Sung Ju Hwang. 2022. Factorized-FL: Personalized Federated Learning with Parameter Factorization & Similarity Matching. In *Proceedings of NeurIPS*.
- [11] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Ben-nas, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2021. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning* 14, 1–2 (2021), 1–210.
- [12] Ari Keränen, Jörg Ott, and Teemu Kärkkäinen. 2009. The ONE simulator for DTN protocol evaluation. In *Proceedings of SimuTools*. 1–10.
- [13] A Krizhevsky. 2009. Learning Multiple Layers of Features from Tiny Images. *Master’s thesis, University of Tront* (2009).
- [14] Fan Lai, Xiangfeng Zhu, Harsha V Madhyastha, and Mosharaf Chowdhury. 2021. Oort: Efficient federated learning via guided participant selection. In *Proceedings of OSDI*. 19–35.
- [15] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.
- [16] Jong-Kwon Lee and Jennifer C Hou. 2006. Modeling steady-state and transient behaviors of user mobility: formulation, analysis, and application. In *Proceedings of MobiHoc*. 85–96.
- [17] Ang Li, Jingwei Sun, Pengcheng Li, Yu Pu, Hai Li, and Yiran Chen. 2021. Hermes: an efficient federated learning framework for heterogeneous mobile clients. In *Proceedings of MobiCom*. 420–437.
- [18] Chennng Li, Xiao Zeng, Mi Zhang, and Zhichao Cao. 2022. PyramidFL: A fine-grained client selection framework for efficient federated learning. In *Proceedings of MobiCom*. 158–171.
- [19] Guanghao Li, Yue Hu, Miao Zhang, Ji Liu, Quanjin Yin, Yong Peng, and Dejing Dou. 2022. FedHiSyn: A hierarchical synchronous federated learning framework for resource and data heterogeneity. In *Proceedings of ICPP*. 1–11.
- [20] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2020. On the Convergence of FedAvg on Non-IID Data. In *Proceedings of ICLR*.
- [21] Bing Luo, Xiang Li, Shiqiang Wang, Jianwei Huang, and Leandros Tassiulas. 2021. Cost-effective federated learning design. In *Proceedings of INFOCOM*. 1–10.
- [22] Xincheng Lyu, Chenshan Ren, Wei Ni, Hui Tian, Ren Ping Liu, and Eryk Dutkiewicz. 2019. Optimal online data partitioning for geo-distributed machine learning in edge of wireless networks. *IEEE Journal on Selected Areas in Communications* 37, 10 (2019), 2393–2406.
- [23] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of AISTATS*. 1273–1282.
- [24] Jer Shyuan Ng, Wei Yang Bryan Lim, Zehui Xiong, Xianbin Cao, Jiangming Jin, Dusit Niyato, Cyril Leung, and Chunyan Miao. 2021. Reputation-aware hedonic coalition formation for efficient serverless hierarchical federated learning. *IEEE Transactions on Parallel and Distributed Systems* 33, 11 (2021), 2675–2686.
- [25] Jer Shyuan Ng, Wei Yang Bryan Lim, Zehui Xiong, Xianbin Cao, Dusit Niyato, Cyril Leung, and Dong In Kim. 2021. A hierarchical incentive design toward motivating participation in coded federated learning. *IEEE Journal on Selected Areas in Communications* 40, 1 (2021), 359–375.
- [26] Kenta Niwa, Guoqiang Zhang, W Bastiaan Kleijn, Noboru Harada, Hiroshi Sawada, and Akinori Fujino. 2021. Asynchronous decentralized optimization with implicit stochastic variance reduction. In *Proceedings of ICML*. 8195–8204.
- [27] Pubudu N Pathirana, Andrey V Savkin, and Sanjay Jha. 2003. Mobility modelling and trajectory prediction for cellular networks with mobile base stations. In *Proceedings of MobiHoc*. 213–221.
- [28] Zhe Qu, Xingyu Li, Jie Xu, Bo Tang, Zhuo Lu, and Yao Liu. 2022. On the Convergence of Multi-Server Federated Learning with Overlapping Area. *IEEE Transactions on Mobile Computing* (2022). <https://doi.org/10.1109/TMC.2022.3200016>
- [29] Lina Su, Ruiting Zhou, Ne Wang, Guang Fang, and Zongpeng Li. 2022. An Online Learning Approach for Client Selection in Federated Edge Learning under Budget Constraint. In *Proceedings of ICPP*. 1–11.
- [30] Hao Wang, Zakhary Kaplan, Di Niu, and Baochun Li. 2020. Optimizing federated learning on non-iid data with reinforcement learning. In *Proceedings of INFOCOM*. 1698–1707.
- [31] Jiayi Wang, Shiqiang Wang, Rong-Rong Chen, and Mingye Ji. 2022. Demystifying Why Local Aggregation Helps: Convergence Analysis of Hierarchical SGD. In *Proceedings of AAAI*. 8548–8556.
- [32] Yidan Wang, Zahir Tari, Xiaoran Huang, and Albert Y Zomaya. 2019. A network-aware and partition-based resource management scheme for data stream processing. In *Proceedings of ICPP*. 1–10.
- [33] Zhiyuan Wang, Hongli Xu, Jianchun Liu, He Huang, Chunming Qiao, and Yangming Zhao. 2021. Resource-efficient federated learning with hierarchical aggregation in edge computing. In *Proceedings of INFOCOM*. 1–10.
- [34] Pete Warden. 2018. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209* (2018).
- [35] Xinyi Xu, Lingjuan Lyu, Xingjun Ma, Chenglin Miao, Chuan Sheng Foo, and Bryan Kian Hsiang Low. 2021. Gradient driven rewards to guarantee fairness in collaborative machine learning. *Proceedings of NeurIPS* (2021), 16104–16117.
- [36] Lei Yang, Yingqi Gan, Jiannong Cao, and Zhenyu Wang. 2022. Optimizing Aggregation Frequency for Hierarchical Model Training in Heterogeneous Edge Computing. *IEEE Transactions on Mobile Computing* (2022). doi: 10.1109/TMC.2022.3149584.
- [37] Zhicong Zhong, Yipeng Zhou, Di Wu, Xu Chen, Min Chen, Chao Li, and Quan Z Sheng. 2021. P-FedAvg: parallelizing federated learning with theoretical guarantees. In *Proceedings of INFOCOM*. 1–10.