mathematics

Article

Federated Learning with Efficient Aggregation via Markov Decision Process in Edge Networks

Tongfei Liu [1], Hui Wang [1,*,†] and Maode Ma [2,†]

[1] College of Computer Science and Technology, Zhejiang Normal University, Jinhua 341000, China; afly2021@zjnu.cn
[2] College of Engineering, Qatar University, Doha 974, Qatar; mammdsg@ieee.org
* Correspondence: hwang@zjnu.cn
† These authors contributed equally to this work.

Abstract: Federated Learning (FL), as an emerging paradigm in distributed machine learning, has received extensive research attention. However, few works consider the impact of device mobility on the learning efficiency of FL. In fact, it is detrimental to the training result if heterogeneous clients undergo migration or are in an offline state during the global aggregation process. To address this issue, the Optimal Global Aggregation strategy (OGAs) is proposed. The OGAs first models the interaction between clients and servers of the FL as a Markov Decision Process (MDP) model, which jointly considers device mobility and data heterogeneity to determine local participants that are conducive to global aggregation. To obtain the optimal client participation strategy, an improved σ-value iteration method is utilized to solve the MDP, ensuring that the number of participating clients is maintained within an optimal interval in each global round. Furthermore, the Principal Component Analysis (PCA) is used to reduce the dimensionality of the original features to deal with the complex state space in the MDP. The experimental results demonstrate that, compared with other existing aggregation strategies, the OGAs has the faster convergence speed and the higher training accuracy, which significantly improves the learning efficiency of the FL.

Keywords: federated learning; Markov Decision Process; aggregation strategy; user mobility

MSC: 68T01

Citation: Liu, T.; Wang, H.; Ma, M. Federated Learning with Efficient Aggregation via Markov Decision Process in Edge Networks. *Mathematics* **2024**, *12*, 920. https://doi.org/10.3390/math12060920

Academic Editor: Michael Voskoglou

Received: 27 February 2024
Revised: 12 March 2024
Accepted: 14 March 2024
Published: 20 March 2024

1. Introduction

The rapid advancement and expansion in the number of IoT devices result in an exponential growth in data, posing two key challenges to traditional centralized learning approaches [1,2]. Firstly, centralized methods relying on cloud-based architectures no longer fit the 4G/5G era due to the high storage requirements and communication costs involved in collecting data from millions or even billions of IoT devices (such as large amounts of time-sensitive and high-frequency data from drones or in-vehicle radar sensors). Secondly, user data privacy has become a sensitive topic.c. The regulations have been developed and enacted to ensure the protection of user privacy by many countries [3]. Traditional centralized methods require uploading local data, which undoubtedly exposes user data privacy and puts users at risk of information leakage. Therefore, to improve efficiency and protect data privacy, providing distributed models and training methods for data-driven learning under the premise of privacy preservation has become a hot topic.

To meet this urgent need, Federated Learning (FL) has been proposed as a distributed machine learning paradigm [4]. It enables multiple devices or organizations to collaboratively learn a shared model without compromising the privacy of local data. Specifically, each device trains a local model using its computational capacity and datasets, while a central server is responsible for maintaining global model updates. Through communication between the server and clients, the model accuracy converges to an optimal result.

Without the need for centralized uploading of local data from clients, the distributed and collaborative characteristics of FL provide an effective mechanism for privacy preservation and model training.

Although progress has been made in improving the learning efficiency and privacy protection of FL, distributed training in heterogeneous edge networks still faces three key challenges. (1) Device mobility: Edge devices often move out of the communication area of the current base station during model training, which renders the global model in the previous round stale [5]. Aggregation strategies should consider both heterogeneity and mobility, making optimal decisions based on the local model training results and device mobility. (2) Non-Independent and Identically Distributed (Non-IID) datasets: In real scenarios, the data distribution on most edge devices is diverse and personalized, resulting in variations in local training results and subsequently impacting the global mode. Designing appropriate aggregation strategies can mitigate the effects of Non-IID data to the maximum extent. (3) Long-deadlines: The difference in device computing power causes FL to need to wait for the slowest device to complete the training task, which not only incurring high computational and communication costs when processing such tasks but also results in huge latency.

To deal with the abovementioned challenges, it is an effective strategy to select appropriate local models to join the global aggregation in the each global round. In general, the design of the aggregation strategy needs to consider the real situation of the training device. By solving the resource scheduling problem to decide the aggregation strategy, ref. [6] succeeded in energy consumption minimization. Also, a fair scheduling mechanism regarding the number of local iterations and resource allocation was introduced into the aggregation strategy [7]. However, most of the existing works are based on the assumption of static scenes; device mobility is a significant aspect that cannot be overlooked in real scenarios. In a heterogeneous environment, we establish a Markov Decision Process (MDP) framework, which incorporates client heterogeneity and device mobility status to derive the optimal client participation decisions. Client heterogeneity is usually manifested in the difference in data distribution information, and the occurrence of mobility often makes the client offline from the current cluster server. The global aggregation takes a considerable amount of time when some devices are offline. The main contributions of this paper can be summarized as follows:

1. This article investigates the impact of device heterogeneity and mobility on the learning efficiency of FL in an edge environment. We propose OGAs to maximize the learning efficiency of FL. In addition, this article analyzes multiple possibilities for location changes of mobile devices when they participate in the global aggregation.
2. The OGAs use the MDP model to describe the interaction between clients and the server. In order to obtain the optimal strategy, an improved σ-value iteration algorithm is employed to address the MDP problem. In addition, to overcome the curse of dimensionality caused by the high-dimensional state space in the MDP model, dimensionality reduction is performed on the original features using the Principal Component Analysis (PCA) scheme, which utilizes the reduced feature space to represent the state set. This approach is simple and efficient, because it can improve efficiency without sacrificing learning accuracy.
3. Extensive simulations were conducted to evaluate the performance of the proposed strategy and investigate the impact of different environment settings on learning efficiency. The simulation results demonstrate that, compared to several other client participation strategies, the proposed strategy can significantly improve the learning efficiency of FL.

The remainder of this paper is organized as follows. The related work of the FL is introduced in Section 2. And the motivating experiment is described in Section 3. Section 4 presents the system model. And in Section 5, we provide a description of the OGA strategy. Section 6 discusses the simulation results. Finally, Section 7 concludes the paper.

2. Related Work

Recently, extensive studies of FL have been conducted in the literature. These works can be primarily categorized into the following three areas: (1) communication efficiency optimization; (2) computation efficiency improvement; (3) client selection/scheduling.

In FL, frequent communication between clients and the server led to significant communication costs [8]. To improve communication efficiency, ref. [9] proposed a bandwidth allocation and scheduling strategy that adapts to channel conditions and device computational capabilities. Ref. [10] introduced an asynchronous FL framework that considers the differences in communication link quality, computational capabilities, and data distribution. This framework helps alleviate the issue of model staleness caused by asynchronous aggregation. Similarly, ref. [11] proposed a two-stage algorithm to mitigate model staleness and conducted convergence analysis. Furthermore, ref. [12] investigated the problem of minimizing FL communication latency and theoretically proved that the total delay is a convex function of learning accuracy. They utilized the method of bisection to obtain the optimal solution.

In improving computation efficiency, existing works tend to focus on designing adaptive solutions. Ref. [13] proposed an adaptive control algorithm that dynamically adjusts the FL global aggregation frequency based on convergence bounds. To achieve better learning performance on Non-IID data, ref. [14] introduced an experience-driven algorithm based on deep reinforcement learning (DRL) to adaptively determine hyperparameters during model training. Ref. [15] extended the FedAvg algorithm and proposed an adaptive weighting strategy to mitigate the negative impact of Non-IID data on FL performance. Ref. [16] designed an adaptive optimizer for sparse general gradients. Ref. [17] developed a weighted aggregation heuristic algorithm that utilizes lossy compression techniques to reduce communication costs without compromising model accuracy.

Moreover, the client selection problem is a hot topic in improving FL efficiency. Joint client selection and resource allocation optimization were studied in Refs. [6,18,19]. Ref. [20] introduced a deadline-aware aggregation approach to set deadlines at specific stages of FL to aggregate as many clients' local models as possible, making the overall training process more efficient. Ref. [21] employed multi-armed bandit (MAB) techniques to adaptively determine clients for global aggregation and demonstrated its effectiveness. Ref. [22] transformed a trade-off problem in FL client selection into an optimization problem and formulated it as a total communication cost minimization problem. Ref. [7] utilized resource-aware techniques and availability to select clients.

Indeed, users are highly likely to be mobile in edge environments. However, most prior works have overlooked the impact of user mobility. To the best of our knowledge, there is limited research on the effects of device mobility on FL efficiency. Only Refs. [23,24] considered user mobility in the study of FL. The work of Ref. [24] primarily focused on proving the convergence of FL in the presence of user mobility, while the work of Ref. [23] overlooked the degradation of the global model due to the lack of effective training data caused by Non-IID data when selecting mobile vehicles for the aggregation process.

3. Motivating Experiment

Previous studies have shown that Non-IID data can slow down the convergence speed of FL due to the divergence in the distribution of local data samples [25,26]. However, these studies are just based on assumptions made in static scenario settings and without considering the possibility of device mobility. In this section, we design an experiment to investigate the impact of device mobility on FL model training. Specifically, we deployed 100 clients in a simulated environment. In each experiment, the clients were randomly migrated with different proportions in the environment, and the probability of the client migration can be modeled as a Markov chain [27]. Each mobile client possessed 600 image data samples belonging to five different classes. The FL server selected mobile devices to ensure that a certain number of clients participated in each round of training, thus ensuring model convergence.

Considering device mobility, the aggregation of the current global round may generate disconnected clients accompanied by the access of unfamiliar clients, which undoubtedly poses challenges to the model training. On one hand, user mobility can lead to a reduction in the number of participants in each global aggregation round, which may result in missing training data and underfitting of the training model. On the other hand, user mobility can also reshape the data, helping to reduce the differences between clusters and constructing representative samples at each stochastic gradient descent (SGD) step of the global model. Figures 1 and 2 illustrate the accuracy of FL in model training with different mobility and Non-IID ratios of clients. In these results, as the ratios of mobile devices and Non-IID increase, the training results tend to be less ideal. Moreover, it can be observed in the figures that the impact of client mobility on FL efficiency is far greater than the impact of Non-IID on FL efficiency. Therefore, this motivates us to design rational and efficient approaches to enhance FL efficiency.

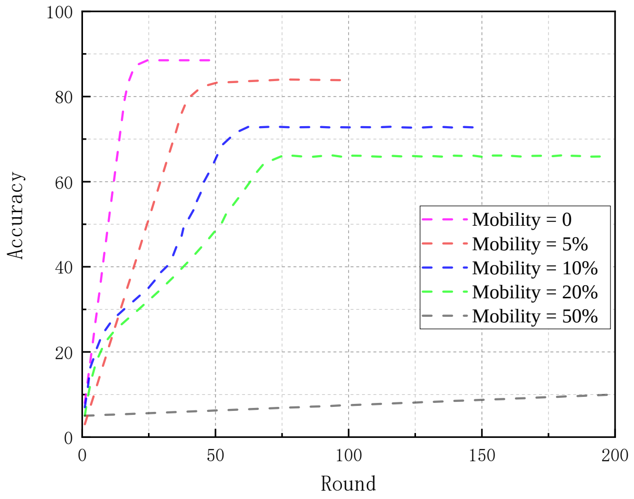

Figure 1. Model training accuracy under different mobility ratios.

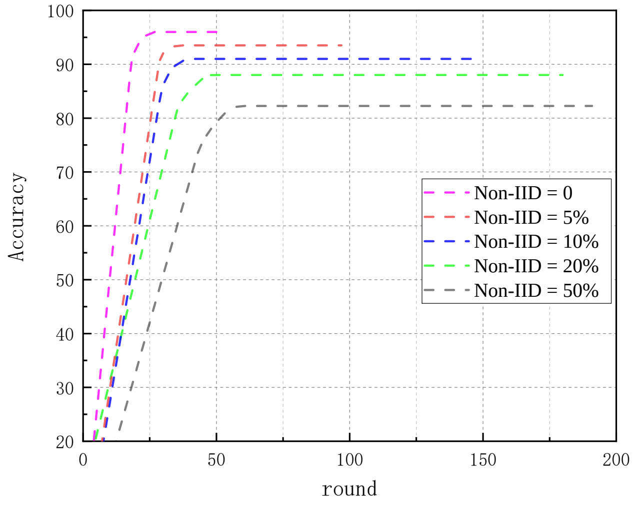

Figure 2. Model training accuracy under different Non-IID ratios.

4. System Model

We consider an edge system for FL consisting of N mobile clients and M edge servers $(N >> M)$, where N clients are randomly divided into M clusters. Each client is indexed as i, $i \in \{1, 2, \cdots, N\}$, and each edge server is indexed as j, $j \in \{1, 2, \cdots, M\}$. A mobile client i connects to the nearest edge server based on its real-time location to participate in the current round of global model aggregation. To describe mobility, we use $\{c_j\}$ to represent the set of mobile clients in a cluster, noting that the size of $|c_j|$ may vary with the number of mobile users. For convenience, we assume that multiple clients participate in the model training in each cluster (i.e., $c_j \geq 1$) to ensure that each cluster can independently complete the learning task (Figure 3).

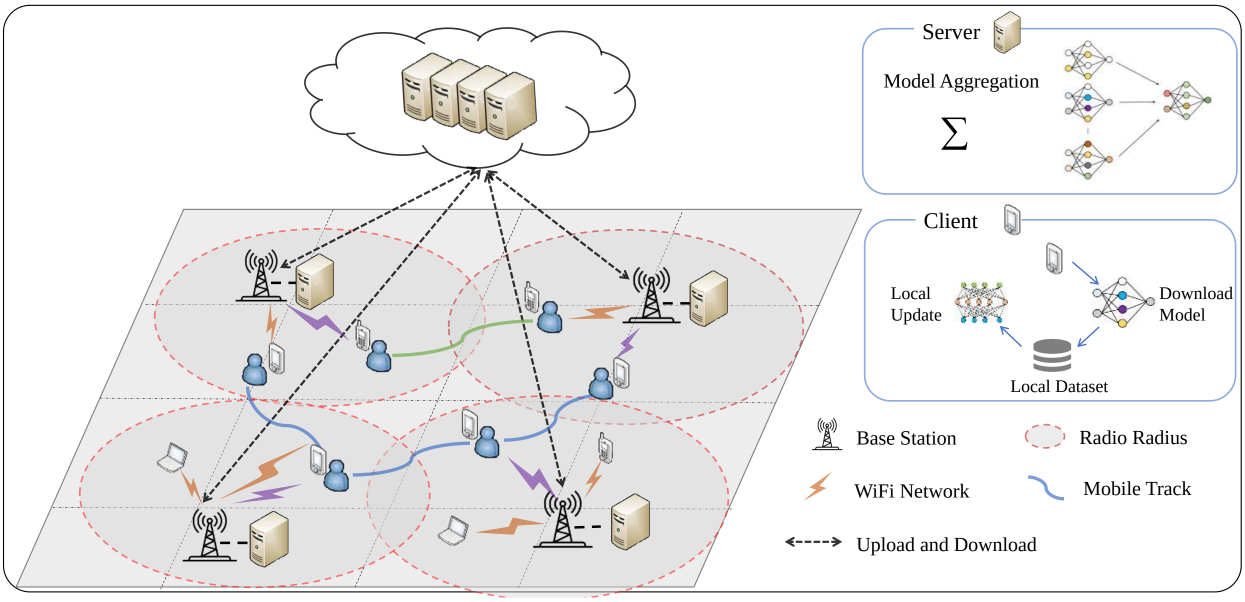

Figure 3. System model.

4.1. The FL Process

In general, FL is built upon a machine learning paradigm, where a typical machine learning task consists of a dataset D and a model parameter vector ω that is trained based on the data samples. In order to reach the desired objective, each client i trains a local model based on its local data samples $d_i = \{x_i, y_i\}, d_i \in D$. The cluster server j aggregates the local models from all nodes and maintains the updates of the global model.

For each set of data samples d_j from client i, we define a loss function $f_j(\omega, x_j, y_j)$, which captures the error between the predicted value $\hat{y}_j$ and the actual value y_j. For instance, in a linear model, the squared loss function $f_j = \frac{1}{2}(\omega x_j - y_j)^2$ is commonly used. For all data samples $d_j \in D_i$ on client i, the local loss function is defined as follows:

$$F_i(\omega) = \frac{1}{|D_i|} \sum_{d_j \in D_i} f_j(\omega, x_j, y_j) \tag{1}$$

The global loss function on the distributed dataset from all devices can be defined as follows:

$$F(\omega) = \frac{\sum_i D_i}{|D|} F_i(\omega) \tag{2}$$

To minimize $F(\omega)$, the SGD technique is commonly used to search for the optimal ω^*. At each time slot t, local updates based on gradient descent are performed according to the following rule:

$$\omega_i^t \leftarrow \omega_i^{t-1} - \eta \nabla F_i\left(\omega_i^{t-1}\right) \tag{3}$$

where $\eta > 0$ is the learning rate, and $\nabla F_i(\cdot)$ is the local model gradient.

4.2. The Mobility Model

Due to the mobility of the devices, the set of clients participating in the current round of global aggregation in each cluster is not fixed. We assume that all mobile clients are uniformly distributed across the entire network. And the mobile clients can randomly move to neighboring clusters with an certain probability. We use a Markov chain model to describe the migration trace of different mobile clients as in the same scenario in Ref. [24]. Specifically, we use an indicator factor α^r to capture the connection status between client i and cluster c_j at the round r, when $\alpha^r = 1$ indicates that the client i is connected to the current cluster. Otherwise, it connects to a neighboring cluster. Figure 4 describes the transition probability of mobile devices between different clusters. For the cluster topology of the entire network, we define a directed connection graph $G = \langle V, E \rangle$, where V represents the set of nodes and E represents the set of edges. The adjacency matrix of the graph G is defined as $A = (\mathrm{a}_{ij})_{n \times n}$, and its elements can be defined as follows:

$$\mathrm{A}_{ij} = \begin{cases} 1, & \text{if} c_j \in V(c_j) \\ 0, & \text{if} c_j \notin V(c_j) \end{cases} \tag{4}$$

where $V(c_j)$ represents the set of adjacent clusters of c_j, and the size of $V(c_j)$ can be calculated as $\sum_{j=1}^{M} A_{ij}$.

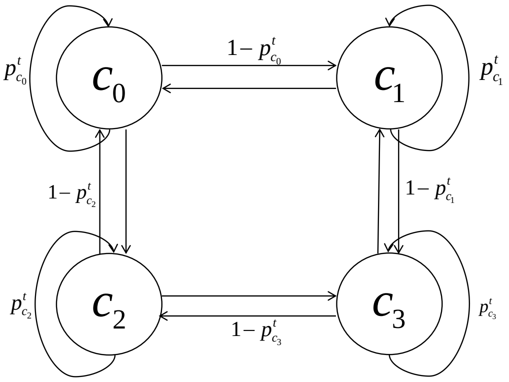

Figure 4. The transition probability of mobile devices between different clusters.

Before the global update, the mobile client i in any cluster has two possible transition states, e.g., stay in the current cluster with a probability $p^r_{c_j}$, and moving to a neighboring cluster is $1 - p^r_{c_j}$, as shown in Figure 4. To simplify the model, we assume that a mobile client migrates to its adjacent cluster with a certain and same probability, and all mobile clients can randomly roam over the whole cluster space. The transition probability of the mobile device moving from cluster c_i to cluster c_j at the r-round is defined as p^r_{ij}, which can be computed as

$$p^r_{ij} = \begin{cases} p^r_{c_j}, & \text{if} \quad \mathrm{i} = \mathrm{j} \\ \frac{1 - p^r_{c_j}}{\sum A_{ij} - 1}, & \text{if} c_j \in V \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

when $p^r_{c_j} = 1$, it indicates that the system is in a stationary state, which is equivalent to a conventional setup of FL [4]. For the sake of generality, we assume that a client may always move about before the global round of aggregation, which facilitates the synchronous training of the global model.

4.3. The MDP Model

After all clients have completed uploading their local training results for the current round, the cluster server needs to immediately aggregate and broadcast the new global model. However, due to the heterogeneity of local data distributions and the uncertainty of client mobility, it is challenging to dynamically select a set of local models that will yield desirable results for the global aggregation in the current round. Moreover, in a distributed training environment, frequent and meaningless interactions between clients and the server can only increase communication costs and degrade model training results.

In this part, we employ the MDP to model the interaction between the cluster server and mobile clients, perceive the mobility and the knowledge of data distribution of the mobile clients, and optimize the client selection strategy to maximize the learning efficiency of FL.

By the MDP, the system remains in a certain state s, $s \in S$. The agent selects an action a, $a \in A$ to take in the current state. After the action is performed, the agent receives an immediate reward R, and the system transitions to the next state s' according to the transition probability P. In the FL system, the decision of which clients participate in the global aggregation for the current round is based on the mobility, described as a standard MDP. The details are described as follows.

4.3.1. State Set

The state set S includes all possible states in the FL system, including the clients' computational capabilities, data distributions, and the moving trail of the devices. S can be defined as

$$S = K \times D \times L \tag{6}$$

where K and D represent the local computing resources and local dataset information of the client i, respectively. L represents the geographical location (i.e., the possible cluster) of the client i. Additionally, × represents the Cartesian product. These can be further described as follows:

$$D = \{d_1, d_2, \cdots\cdots, d_{\max}\} \tag{7}$$

$$K = \{k_1, k_2, \cdots\cdots, k_{\max}\} \tag{8}$$

$$L = \{l_{c_1}, l_{c_2}, \cdots\cdots, l_{c_m}\} \tag{9}$$

Specifically, $k_i \in K$ represents the local computational resources of the mobile client, which depends on the device's computational capabilities. And, $k_1 \to k_{\max}$ is a gradually increasing order that represents different clients' computational resources. The dataset $d_i \in D$ is derived from the client's local data distribution. Since the training results of Non-IID data will ultimately reflect on the model's learning accuracy, the system's decision-making process in each round t aims to minimize the impact of Non-IID data. Lastly, the value of l_{c_j} reflects the real-time location of the mobile client i. For example, in round r, the system state can be represented as $s^r = (k_i^r, d_i^r, l_{c_i}^r)$, $s^r \in S$.

4.3.2. Action Set

The action set A includes all possible actions that the system agent can take. Therefore, A can be described as

$$A = \{a_1, a_2, \cdots\cdots a_n\} \tag{10}$$

where $a_i = \{0, 1\}$ and $i = \{1, 2, \cdots\cdots, n\}$. When $a_i = 0$, it indicates that the client i has not participated in the current global round of model updates. Otherwise, it has some actions. Based on the current state s, the mobile client i will make a decision by selecting a specific action a.

4.3.3. Transition Probability

The transition probability $P(s'|s,a)$ represents the probability of moving from the current state s to the next state s' by taking action a. Assuming that the state transitions of each mobile client i are independent of each other, the following equation holds:

$$P(s'|s,a) = \begin{cases} P(K'_i|K)P(D'_i|D_i)P(L'_i|L_i), a'_i = a_i \\ 0, a'_i \neq a \end{cases} \tag{11}$$

The conditional probability $P(\cdot|\cdot)$ represents the probability of transitioning from the current state to the next state, where K_i, D_i, and L_i represent the local computing resources, data distributed information, and moving trail of the mobile devices in the current state.

Also, K_i', D_i', and L_i' represent the device information of the mobile clients in the next state. In this article, we assume that the data samples and computing resources of the mobile device do not change until the current global round finishes, so that $P(K'_i|K)$ and $P(D'_i|D_i)$ can be defined in a statistical pattern:

$$P\left(K'_i|K_i\right) = \begin{cases} 1, \text{if} K'_i = K_i \\ 0, \text{otherwise} \end{cases} \tag{12}$$

$$P\left(D'_i|D_i\right) = \begin{cases} 1, \text{if} D'_i = D_i \\ 0, \text{otherwise} \end{cases} \tag{13}$$

The conditional probability $P(L'_i|L_i)$ represents the probability that the mobile device moves from cluster L to next cluster L'. $P(L'_i|L_i)$ can be derived by

$$P\left(L'_i|L_i\right) = \begin{cases} \delta, & \text{ifL}'_i = L_i \\ (1-\delta)/\left|c_j\right|, \text{otherwise} \end{cases} \tag{14}$$

where the size of $\delta(0 \leq \delta \leq 1)$ indicates the probability of the mobile device staying in the same cluster in two sequential decision rounds, and the exact value can be computed by Equation (5). Specially, the transition probability depends only on the action performed in the current state and has an affect on the expected reward of each client. The design of expected rewards will be introduced in the next part.

4.3.4. Reward Function

A reward function $R(s, a)$ describes an immediate reward for the mobile device when it selects an action at the current state s. The completion time of various stages in FL describes the computational efficiency of the chosen action, aiming to achieve efficient training results in a shorter time scale. In the scenario of this paper, we assume that the migration of the device occurs during the model training of the device and ignore the migration time between two consecutive decision epochs. Here, the reward function is defined as follows:

$$R(s, a) = T_{comm.} + max(T_{train.}, T_{migr.}) \tag{15}$$

where $T_{\text{comm.}}$ represents the time cost for model communication (including upload and download). By deploying the cluster server on the base station, the mobile client can communicate with the server over a wireless network (such as a cellular network). Consequently, the model communication time can be computed as $T_{\text{comm.}} = \Phi(\omega)/\gamma$. The $\Phi(\omega)$ represents the data size of the local model, and the γ is the wireless network transmission rate, which depends on the wireless network channel conditions. Furthermore, when the CPU frequency f_i of the mobile client i is given, the model training time $T_{train.}$ can be calculated as $T_{train.} = W_i/f_i$, where the W_i represents the total CPU cycles of the client i for training the learning model. Additionally, the value of the migration time $T_{migr.}$ of the mobile client can be obtained from the ratio of distance to velocity. In our simulation experiments, all clients were assumed to move at a 1 km/h speed.

By the MDP, the mobile client will choose an action within a specific period to connect to the cluster server in the current state and upload the local model. Based on it, the server can obtain the transition probability related to the current state and action of the client, and then give the action reward in the period. Similarly, the cluster server is responsible for maximizing the average reward, considering the uncertainty of client mobility and the indeterminacy of task completion time in each stage. Therefore, within the specific period, it is critical for the server to select a group of the optimal clients to join the model training and update in the current round. By analyzing the MDP over an infinite time horizon, we designed the optimal client selection strategy to solve this problem.

5. The Optimal Global Aggregation Strategy

In this section, we consider the problem of optimal client selection in each round of FL, which is formulated as an MDP over an infinite time horizon. By jointly considering the mobility of clients and knowledge of data distribution, the optimal decision can be derived for each round to achieve efficient learning results. Generally, based on the current system state, the MDP generates a policy π, which reflects the mapping between states and actions. The MDP ultimately finds the optimal policy π^* in a steady state, maximizing the expected discounted total reward. By introducing the value function v_π to describe the expected reward of policy π, the formula is defined as follows:

$$v(\pi) = \mathbb{E}_\pi\left[\sum_{t=1}^{N} \mu^{t-1} R(s,a)\right] \tag{16}$$

where $\mathbb{E}[\cdot]$ represents the expected function of the desired return for policy π, μ is the discount factor, and $r \in (0,1]$. Our goal is to solve the MDP over an infinite time horizon to obtain the OGA decision for each global round in FL. Therefore, the problem can be formulated as follows:

$$\pi^* = \arg\min_{\pi}\left\{\sum_{S}\sum_{A} v(\pi)\varphi(s,a)\right\} \tag{17}$$

where $\varphi(s,a) \in (0,1]$ represents the stationary probability of the system in the steady state. In general, the MDP is a mathematical model of reinforcement learning that learns the mapping of states to actions to maximize the rewards.

The problem of solving the optimality equation of the MDP can be characterized as a dynamic programming problem, which is typically solved by using value iteration or policy iteration to seek the optimal solution in a polynomial time. Policy iteration can be time-consuming when solving multi-objective optimization problems, while plain value iteration without proper control may result in difficulty in convergence. Therefore, in our iterative process of solving the MDP, we set a stopping criterion σ, where the iteration stops immediately when the following condition holds true:

$$\left\| v_\pi^t(s) - v_\pi^{t-1}(s) \right\| < \sigma \tag{18}$$

where $\|\cdot\|$ represents the Euclidean norm, and σ is a positive real number that ensures the convergence of the algorithm. Given a threshold value σ for stopping the iteration, the σ-value iteration algorithm can be executed by the cluster servers or the central cloud. Then, the mobile clients can compute the optimal policy based on their geographical locations and data distributions, including each state in S and the corresponding action in A.

We propose the σ-value iteration algorithm to solve the MDP and find the optimal decision that maximizes the efficiency of FL, as shown in Algorithm 1. The outer loop iterates based on the expected system performance, including the moving trail and data distributions, to derive the optimal decision and ensure the selection of a certain number of mobile clients for aggregation in each global round. The inner loop is used to update the value function, which consists of two steps. (1) Policy improvement: the policy is improved based on the previous round's value function to obtain the greedy policy for the current round. (2) Policy evaluation: the greedy action in state is selected based on the greedy policy to update the corresponding value function.

Firstly, the algorithm initializes A to an empty set and creates a probability transition matrix for all mobile clients. Then, the algorithm establishes the migration model for mobile clients and selects the corresponding client data distribution in each cluster. It computes the individual polling time for mobile clients based on $T_{\text{comm.}}$, $T_{training}$, and $T_{trans.}$. At this point, since the selected clients are unknown, the algorithm estimates a corresponding decision based on the real-time geographic location of the mobile clients and their local data distribution. After it, the iterative process begins. During each policy evaluation, the initial values of the value function are the value function from the previous iteration, obtained by solving the following Bellman's optimality equation [28]:

$$v_\pi(s) = \max_{a\in A} \sum\nolimits_S p(s'|s,a)\left[\mu + r v_\pi(s')\right] \tag{19}$$

In practice, it is possible to obtain the optimal policy through value iteration before the algorithm converges. Therefore, it is necessary to change the termination condition of the value iteration algorithm. Specifically, when the current two estimates of the value function satisfy Equation (18), it is time to terminate the iteration to speed up the program execution time.

Algorithm 1 Optimal Global Aggregation (OGA).

```
Require: $N, M, D, t = 0$;
  For each client.
  ES initializes globle model $w(0)$;
  for client $i \in N$ in parallel do
     Maintain the transition probability matrix P;
     if $c_i \in V$ then
        Local model update with SGD according to (3);
        Send local model $w_i^t$ to $\{c_i\}_j$;
     else
        Download globe model $w^{t+1}$
        Send local model $w_i^{t+1}$ to $\{c_i\}_{j+1}$;
     end if
  end for
  For Cluster Server.
  Initialize the sets ClientList[];
  Iterate MDP export $\pi^*$;
  Calculating expected reward according to (15)
  for $\left\|v_\pi^t(s) - v_\pi^{t-1}(s)\right\| < \sigma$ do
     $v_\pi(s) \leftarrow \max_{a\in A} \sum_S p(s'|s,a)[\mu + \mu v_\pi(s')]$
  end for
  Add the mobile client to ClientList[];
```

To ensure the efficiency of FL model training, it is preferable to include as many mobile clients as possible in the aggregation process. To prevent the training from deteriorating, clients in A that have the shortest migration path will be selected. Assuming that the local data distribution of all mobile clients does not change with client migration, the global round can be updated by adding and removing decisions from the decision set. This process is repeated until the model converges. Additionally, after a client migration, clients should update their local models based on the correlation between the local data distribution and the global data distribution. Therefore, the MDP decision can be made by jointly considering client migration and data distribution.

Dimensionality Reduction

Since the state space is combined with the local computing resources, data distribution, and geographical location of mobile clients, these results could occupy a huge state space. Moreover, the state space continues to grow over time, making it challenging to solve the MDP. To tackle the dimensionality challenge, the Principal Component Analysis (PCA) scheme is employed to perform dimensionality reduction on the original features and use the reduced information to represent the state set.

Since there is some correlation among the multiple-dimensional variables in the state space, such as the dependence of data distribution information on the geographical location of the client, it is possible to capture the information among the original variables using a smaller set of comprehensive variables known as principal components. Principal components are linearly independent variables obtained by the orthogonal transformation of the observed data represented by linearly correlated variables. The principal components are mutually uncorrelated, meaning that the represented information does not overlap with itself.

The PCA can be used to map high-dimensional data into a low-dimensional space by using a linear projection, with an expectation that the data are to be most informative (with the largest variance) in the projected dimension, thus using fewer data dimensions and retaining more of the characteristics of the original data. The specific process of the PAC can be described as follows. In the initialization phase, it is necessary to input the initial state space set $s^0 = (k_i^0, d_i^0, l_{c_i}^0)$ and calculate the covariance matrix $C = \{\mathrm{cov}(X,Y)\}_{m\times n}$. After obtaining the input sample set, the data are normalized by de-centralization (such as Z-score normalization) in order to put the center of the data sample in the zero position. Subsequently, the projection matrix is constructed according to the feature size sequence $\{w_1, w_2, \cdots, w_n\}$, and the data are transformed into the new space constructed by the projection matrix. According to the maximum variance theory, the projection plane with a larger variance is selected to project the original feature, so as to ensure that the eigenvalue corresponding to each feature vector is the variance of the original feature projected to this projection plane, to achieve the purpose of dimensionality reduction.

A simple experiment was designed to prove the effectiveness of using PAC to reduce the dimension of the state space. On the basis of the experiment in Section 3, a CNN model with 26050 parameters (simulating a high-dimensional state space) was trained on 100 edge devices, and the Non-IID data sample settings were the same as the experiment in Section 3. After 50 local SGD training iterations, the normalized parameter weights were projected onto the new 2D space and the projection matrix continued to be constructed. Figure 5 shows the feature distribution of parameters that trained on devices with different class labels. For example, all red "·" indicates a compressed version of the parameter weights from the device with label "2" during training, i.e., reduced to four dimensions from the original 26,050. In this case, the PAC can be proved to be effective and able to speed up the MDP solution.

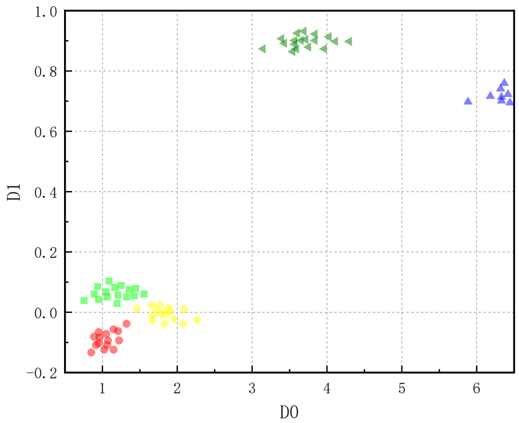

Figure 5. The PAC of model weights on the MNIST datasets.

6. Simulation Results

In this section, we configure the FL system in a heterogeneous environment for experimental evaluation of the proposed scheme, and provide sufficient controlled experiments to verify the effectiveness of our proposed scheme.

6.1. Simulation Setting and Dataset

In the simulation experiments, we configured a cluster of 200 mobile clients and four centered edge servers; the size of the cluster was determined by the radio coverage, and each cluster was responsible for maintaining an FL task process. In the initial phase, all clients were uniformly randomly assigned to four clusters. And clients randomly roamed in the cluster with a specific probability p during each round of FL. We trained a classical Convolutional Neural Network (CNN) model on the open source MNIST dataset for an image classification task. The MNIST dataset is composed of 10 different classes of handwritten digits with 60,000 samples for training and 10,000 samples for testing. The 60,000 data samples are randomly divided into 100 clients, and each client subset independently holds 600 samples. Each client trains a CNN model with a cross-entropy loss function; the mini-batch size of the local SGD is set to 50, and the learning rate η is set to 0.01. Each interaction between the cluster server and the mobile clients is a standard MDP solving process, and the discount factor is set to $r = 0.9$ to describe the expected reward $v(\pi)$ of the strategy π.

Greedy and random strategies were considered controlled trials. Two greedy-based client aggregation strategies were taken into consideration. The first one is the minimum distance strategy (G-MDs) [29], which selects a client to participate in model aggregation according to the minimum Euclidean distance between the mobile client and the current geographic location of the cluster server. Another is the maximum reward strategy (G-MRs) [30], which aims to minimize the learning loss within the limited time and energy budget and select the appropriate client to obtain the overall maximum benefit. The random client aggregation strategy (RCAs) was set up similarly to that in Ref. [4], that is, clients were randomly selected in each cluster to participate in the global aggregation of the current round.

To evaluate the training performance of FL under different strategies, the impact of different proportions of mobile devices and data distribution on the model training effect was investigated. In the simulation experiments, each client locally iterated 300 times to generate an experiment result.

6.2. Effect of Non-IID on Learning Efficiency

We further investigated the impact of different client selection strategies on FL training accuracy under different data distributions. To make it easier to analyze the impact of data heterogeneity, we divided data samples with the same label into the same class (e.g., all images with the label "0"). In this part of the experiment, we set up four data distribution modes of 0% Non-IID, 10% Non-IID, 20% Non-IID, and 50% Non-IID and carried out four rounds of experiments. Each round of the experiments adopted one type of data distribution. Figure 6 plots the performance of training accuracy by the four strategies under different data distribution patterns.

As shown in Figure 6, since the neural network structure is not complex, the convergence rate is very fast in the early stage of training. In particular, when the Non-IID data are more than half, the convergence speed of FL will be greatly reduced, and high learning accuracy cannot be achieved. This is because the participation of Non-IID in training makes the local model gradient diverge from the full gradient, and it is possible to produce wrong gradient directions, which leads to the training falling into a local optimum, which will reduce the final learning accuracy.

Figure 6 also shows that compared with the other three schemes, OGA can achieve a faster convergence speed and a higher learning accuracy, especially in the two extreme cases where there are data samples in the same category and data samples with large differences. Specifically, under the training of the similar data samples, OGA can improve the learning accuracy by about 12.5% compared with the greedy strategy and improve the learning accuracy by about 28.4% compared with the random strategy. For Non-IID sample training, the training results are uncertain due to the randomness of the data division. But in general, the training performance of OGA on Non-IID data samples is better than that by the greedy strategy and the random strategy. This is because OGA considers the local data sample distribution information of the client, and the MDP can make the best decision in the state space, so that the similarity of the client data sample distributions determined in each global round tends to be consistent. It is worth noting that in the 50%

Non-IID setting, the training results of the random strategy become extremely terrible. According to the experimental results, the similarity of local data samples can be increased to improve the learning accuracy.

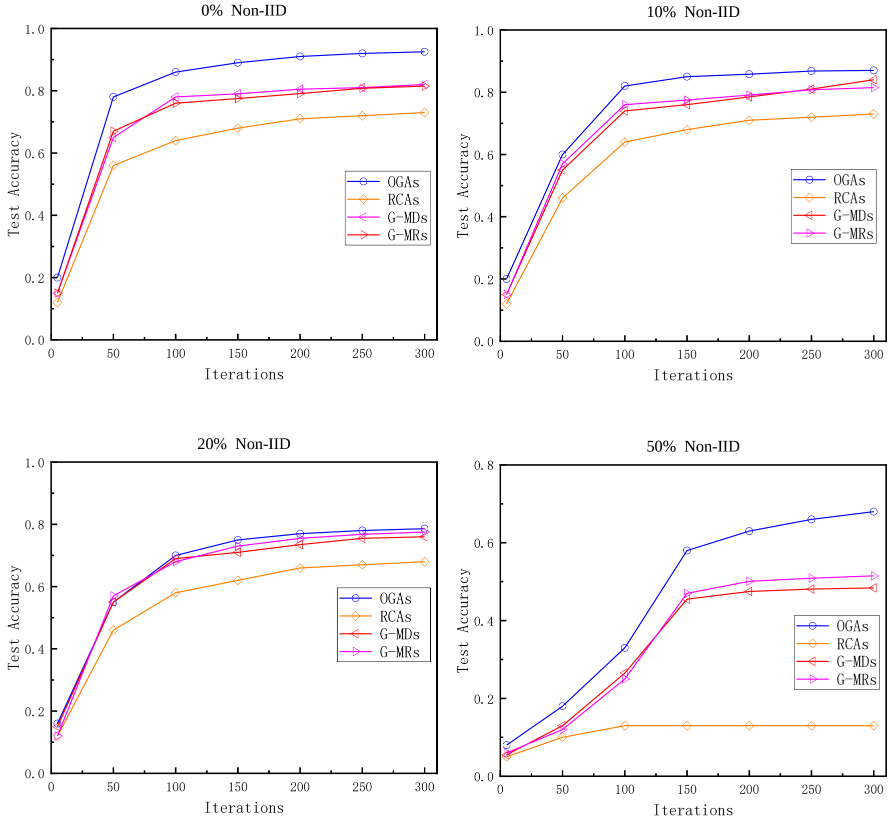

Figure 6. Model training accuracy under different Non-IID ratios.

6.3. *Effect of Mobility on Learning Efficiency*

The impact of four different strategies on model training under different proportions of mobile users has been investigated at present. Figure 7 plots the time spent on model training for the static, 10% mobile client, 20% mobile client, and 50% mobile client scenarios, respectively.

As shown in Figure 7, in the static scenario, the convergence time increases linearly and smoothly with the increase in communication frequency between the client and the server. This result is consistent with the theoretical analysis, because when the number of clients participating in the training and the training accuracy are determined, the time spent on model training is positively correlated with the number of communication rounds. However, when there are mobile clients in the clusters, the clients participating in each round of training start to become uncertain, and the training time to achieve the desired accuracy is also expected to rise, leading to exponential increases with the increase in the number of communication rounds. It is worth noting that when half of the mobile clients are present in the scenario, the convergence time will become very long.

In addition, the simulation results show that the convergence time of the OGA strategy is shorter than that of the other three strategies regardless of whether there is a mobile client involved. This is because the OGA strategy considers the mobility of devices, and the adopted MDP scheme can predict the migration probability of clients before each global aggregation. It can avoid the risk of client absence from aggregation to a certain extent, thus significantly reducing the convergence time of the OGA scheme. In other words, as the value of s and/or a increases, the convergence time of OGA is notably reduced. The OGA strategy can select clients to participate in aggregation more accurately and reduce unnecessary time consumption. Moreover, it is clear in Figure 7 that the convergence time of the MRS is smaller than that of the G-MDs. This is because the pure participation criterion measured by the minimum Euclidean distance may lead to an insufficient number of participating clients, which leads to a long convergence time. Although the random scheme can quickly determine the participating clients for the aggregation, the heterogeneity between clients and servers is the main reason to produce a long convergence time. Therefore, in summary, the OGA scheme is the best in terms of the convergence time when the number of communication rounds increases.

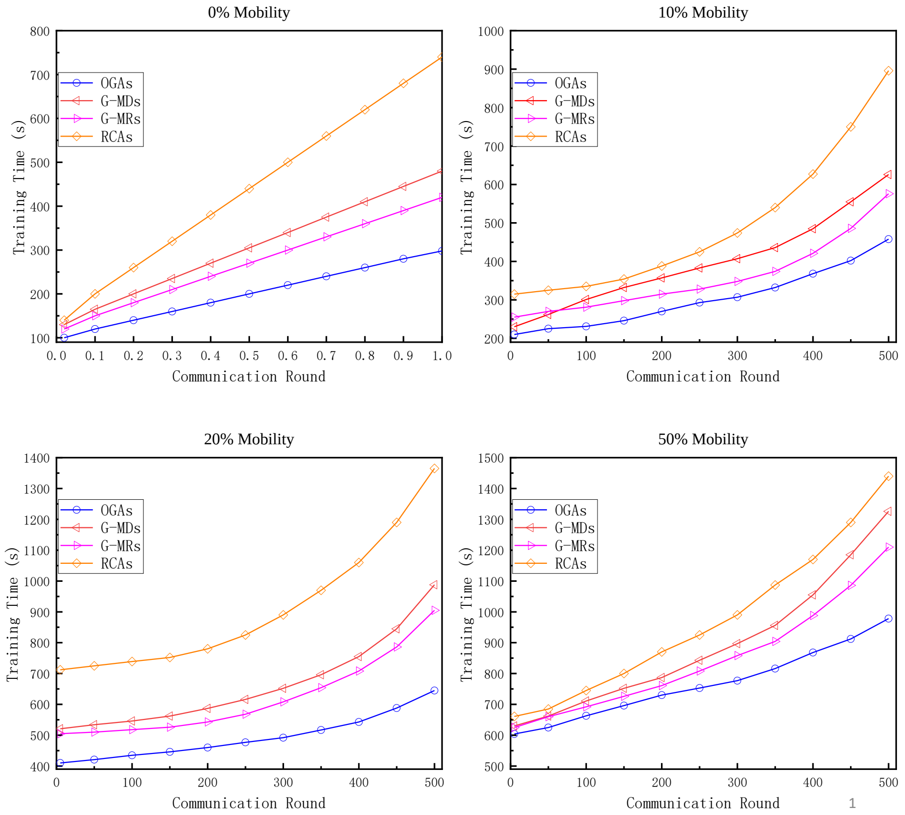

Figure 7. Model training accuracy under different mobility ratios.

6.4. *Effect of N on Learning Efficiency*

By varying the total number of clients in the experiment and including another 50 clients to the simulation environment each time, we aimed to explore the relationship between the number of participating clients and learning efficiency. Figures 8 and 9 show the effect of different total numbers of clients on FL efficiency. It can be observed in Figure 8a that the training time is minimized when the total number of clients is about 150, which indicates that when $N = 150$, there will be enough clients in the system to satisfy the client selection strategies, with each client only participating in the current round of model training, resulting in significant time savings. When $N > 150$, each client may hold an insufficient amount of local data, which will need more clients to participate in each round of training, resulting in longer training times. However, the results are different in the Non-IID setting, as shown in Figure 8b, where the training time decreases as the total number of clients increases. Furthermore, the time required in the model training by the OGA scheme is less than that using the greedy strategy and the random strategy.

Additionally, in Figure 9, the IID setting is fixed to investigate the impact of the total number of clients on learning efficiency. In Figure 9a, the simulation environment is set to a static state without a device for mobility to occur. In contrast, in Figure 9b, the environment is set to a dynamic state where clients can move around. It is clear that in the static state, the training time decreases as the total number of clients increases. And once the total number of clients exceeds 150, the training time remains unchanged. However, in the dynamic state (shown in Figure 9b), the training time increases due to client mobility.

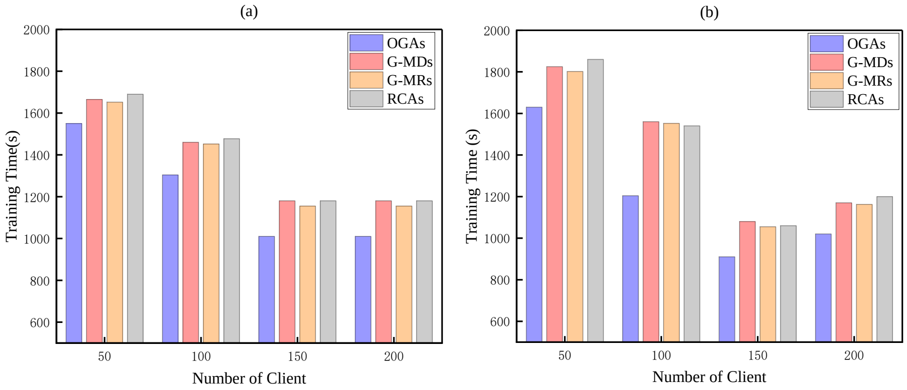

Figure 8. *IID* vs. *Non-IID*, the time cost of model training. (**a**) is the training result in the IID setting, and (**b**) is the result in the Non-IID setting.

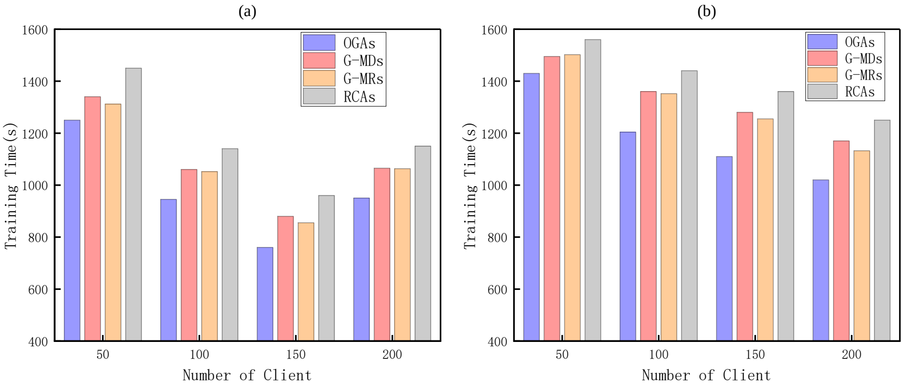

Figure 9. *Mobility* vs. *no mobility*, the time cost of model training. (**a**) is the training result in the no mobility setting, and (**b**) is the result in the mobility setting.

7. Conclusions

In this paper, we investigated the learning efficiency optimization issue for FL in heterogeneous edge networks, with the proposal of the OGA strategy, which establishes an MDP model based on device mobility and data heterogeneity. Then, an improved value iteration algorithm was designed to solve the MDP to obtain the optimal policy. Furthermore, we effectively addressed the issue of the curse of dimensionality in the MDP and optimized the operating efficiency of the OGA scheme. Extensive simulation results demonstrate that the proposed OGA scheme outperforms the other three existing strategies in terms of learning efficiency under given mobility ratios and Non-IID ratios. In the future work, we will investigate an incentive mechanism to encourage more mobile clients to participate in local contributions.

Author Contributions: Writing—original draft preparation, T.L.; writing—review and editing,T.L.; supervision, H.W. and M.M.; funding acquisition, H.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Zhejiang Normal University, grant number 62171413, 2022-4-063.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, T.; Gao, L.; He, C.; Zhang, M.; Krishnamachari, B.; Avestimehr, A.S. Federated learning for the internet of things: Applications, challenges, and opportunities. *IEEE Internet Things Mag.* **2022**, *5*, 24–29. [CrossRef]
2. Lim, W.Y.B.; Luong, N.C.; Hoang, D.T.; Jiao, Y.; Liang, Y.C.; Yang, Q.; Niyato, D.; Miao, C. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 2031–2063. [CrossRef]
3. Jain, P.; Gyanchandani, M.; Khare, N. Big data privacy: A technological perspective and review. *J. Big Data* **2016**, *3*, 1–25. [CrossRef]
4. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the 20th Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 20–22 April 2017; pp. 1273–1282.
5. Wang, Z.; Xu, H.; Liu, J.; Huang, H.; Qiao, C.; Zhao, Y. Resource-efficient federated learning with hierarchical aggregation in edge computing. In Proceedings of the IEEE INFOCOM 2021—IEEE Conference on Computer Communications, Vancouver, BC, Canada, 10–13 May 2021; pp. 1–10.
6. Yu, L.; Albelaihi, R.; Sun, X.; Ansari, N.; Devetsikiotis, M. Jointly optimizing client selection and resource management in wireless federated learning for internet of things. *IEEE Internet Things J.* **2021**, *9*, 4385–4395. [CrossRef]
7. Eslami Abyane, A.; Drew, S.; Hemmati, H. MDA: Availability-Aware Federated Learning Client Selection. *arXiv* **2022**, arXiv:2211.14391.
8. Konečný, J.; McMahan, B.; Ramage, D. Federated optimization: Distributed optimization beyond the datacenter. *arXiv* **2015**, arXiv:1511.03575.
9. Zeng, Q.; Du, Y.; Huang, K.; Leung, K.K. Energy-efficient radio resource allocation for federated edge learning. In Proceedings of the 2020 IEEE International Conference on Communications Workshops (ICC Workshops), Dublin, Ireland, 7–11 June 2020; pp. 1–6.
10. Wang, Z.; Zhang, Z.; Tian, Y.; Yang, Q.; Shan, H.; Wang, W.; Quek, T.Q. Asynchronous federated learning over wireless communication networks. *IEEE Trans. Wirel. Commun.* **2022**, *21*, 6961–6978. [CrossRef]
11. Zhou, Z.; Li, Y.; Ren, X.; Yang, S. Towards efficient and stable k-asynchronous federated learning with unbounded stale gradients on Non-IID data. *IEEE Trans. Parallel Distrib. Syst.* **2022**, *33*, 3291–3305. [CrossRef]
12. Yang, Z.; Chen, M.; Saad, W.; Hong, C.S.; Shikh-Bahaei, M.; Poor, H.V.; Cui, S. Delay minimization for federated learning over wireless communication networks. *arXiv* **2020**, arXiv:2007.03462.
13. Wang, S.; Tuor, T.; Salonidis, T.; Leung, K.K.; Makaya, C.; He, T.; Chan, K. Adaptive federated learning in resource constrained edge computing systems. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 1205–1221. [CrossRef]
14. Liu, J.; Xu, H.; Wang, L.; Xu, Y.; Qian, C.; Huang, J.; Huang, H. Adaptive asynchronous federated learning in resource-constrained edge computing. *IEEE Trans. Mob. Comput.* **2021**, *22*, 674–690. [CrossRef]
15. Xie, C.; Koyejo, S.; Gupta, I. Asynchronous federated optimization. *arXiv* **2019**, arXiv:1903.03934.
16. Sun, H.; Li, S.; Yu, F.R.; Qi, Q.; Wang, J.; Liao, J. Toward communication-efficient federated learning in the Internet of Things with edge computing. *IEEE Internet Things J.* **2020**, *7*, 11053–11067. [CrossRef]
17. Chai, Z.; Chen, Y.; Zhao, L.; Cheng, Y.; Rangwala, H. Fedat: A communication-efficient federated learning method with asynchronous tiers under non-iid data. *arXiv* **2020**, arXiv:2010.05958.
18. Shi, W.; Zhou, S.; Niu, Z.; Jiang, M.; Geng, L. Joint device scheduling and resource allocation for latency constrained wireless federated learning. *IEEE Trans. Wirel. Commun.* **2020**, *20*, 453–467. [CrossRef]
19. Ko, H.; Lee, J.; Seo, S.; Pack, S.; Leung, V.C. Joint client selection and bandwidth allocation algorithm for federated learning. *IEEE Trans. Mob. Comput.* **2021**, *22*, 3380–3390. [CrossRef]
20. Nishio, T.; Yonetani, R. Client selection for federated learning with heterogeneous resources in mobile edge. In Proceedings of the ICC 2019—2019 IEEE International Conference On Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–7.
21. Wang, N.; Zhou, R.; Su, L.; Fang, G.; Li, Z. Adaptive clustered federated learning for clients with time-varying interests. In Proceedings of the 2022 IEEE/ACM 30th International Symposium on Quality of Service (IWQoS), Oslo, Norway, 10–12 June 2022; pp. 1–10.
22. Hosseinzadeh, M.; Hudson, N.; Heshmati, S.; Khamfroush, H. Communication-Loss Trade-Off in Federated Learning: A Distributed Client Selection Algorithm. In Proceedings of the 2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 8–11 January 2022; pp. 1–6.
23. Yu, Z.; Hu, J.; Min, G.; Zhao, Z.; Miao, W.; Hossain, M.S. Mobility-aware proactive edge caching for connected vehicles using federated learning. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 5341–5351. [CrossRef]
24. Feng, C.; Yang, H.H.; Hu, D.; Zhao, Z.; Quek, T.Q.; Min, G. Mobility-aware cluster federated learning in hierarchical wireless networks. *IEEE Trans. Wirel. Commun.* **2022**, *21*, 8441–8458. [CrossRef]
25. Li, X.; Huang, K.; Yang, W.; Wang, S.; Zhang, Z. On the convergence of fedavg on non-iid data. *arXiv* **2019**, arXiv:1907.02189.
26. Wu, H.; Wang, P. Fast-convergent federated learning with adaptive weighting. *IEEE Trans. Cogn. Commun. Netw.* **2021**, *7*, 1078–1088. [CrossRef]
27. Alsheikh, M.A.; Hoang, D.T.; Niyato, D.; Tan, H.P.; Lin, S. Markov decision processes with applications in wireless sensor networks: A survey. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 1239–1267. [CrossRef]
28. Bellman, R. Dynamic programming. *Science* **1966**, *153*, 34–37. [CrossRef] [PubMed]

29. Yang, M.; Wang, X.; Zhu, H.; Wang, H.; Qian, H. Federated learning with class imbalance reduction. In Proceedings of the 2021 29th European Signal Processing Conference (EUSIPCO), Dublin, Ireland, 23–27 August 2021; pp. 2174–2178.
30. Sun, R.; Tao, M. A Greedy Control Policy for Latency and Energy Constrained Wireless Federated Learning. In Proceedings of the 2021 IEEE/CIC International Conference on Communications in China (ICCC), Xiamen, China, 28–30 July 2021; pp. 1119–1124.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.