# Energy-Efficient Device Selection in Federated Edge Learning

Cheng Peng, Qin Hu*, Jianan Chen, Kyubyung Kang, Feng Li, Xukai Zou
Indiana University-Purdue University Indianapolis, IN 46202, USA
Email: {cp16, qinhu, jc144, kyukang}@iu.edu, {fengli, xzou}@iupui.edu
* Corresponding author

***Abstract*—Due to the increasing demand from mobile devices for the real-time response of cloud computing services, *federated edge learning* (FEL) emerges as a new computing paradigm, which utilizes edge devices to achieve efficient machine learning while protecting their data privacy. Implementing efficient FEL suffers from the challenges of devices' limited computing and communication resources, as well as unevenly distributed datasets, which inspires several existing research focusing on *device selection* to optimize time consumption and data diversity. However, these studies fail to consider the energy consumption of edge devices given their limited power supply, which can seriously affect the cost-efficiency of FEL with unexpected device dropouts. To fill this gap, we propose a device selection model capturing both energy consumption and data diversity optimization, under the constraints of time consumption and training data amount. Then we solve the optimization problem by reformulating the original model and designing a novel algorithm, named $E^2DS$, to reduce the time complexity greatly. By comparing with two classical FEL schemes, we validate the superiority of our proposed device selection mechanism for FEL with extensive experimental results.**

## I. Introduction

Nowadays, the demand for the timely and reliable response of cloud computing services becomes increasingly extensive due to the prevailing of real-time applications on mobile devices, which reveals multiple critical deficiencies of the traditional central cloud computing. On the one hand, wireless communication channels may suffer from high latency, low bandwidth, and instability. On the other hand, the security of the cloud center has long been criticized, and data privacy can be easily breached under the attacks of malicious third parties. Regarding the first disadvantage, leveraging on the large-scale deployment of 5G technology, edge computing has played an important role in providing flexible and efficient services to end users. To solve the second problem, federated learning (FL) with privacy protection characteristics can effectively assist in aggregating necessary information to improve training efficiency, so as to provide automatic and intelligent computing. Combining the advantages of the above two solutions, federated edge learning (FEL) emerges recently to holistically address the problem of cloud computing for achieving efficient machine learning with the help of edge devices while protecting their data privacy at the same time.

However, due to the uncertain communication conditions and computing capabilities of edge devices without owning independent and identically distributed (IID) data, there exist huge challenges in efficiently implementing FEL. Faced with such a situation, a lot of existing studies are devoted to the research of server-oriented optimization in the FEL process, including *resource allocation* [1]–[3], *intermediate server assistance* [4], [5] and *device participation incentive* [6]–[8]. Although these methods are optimized under the premise of a given set of edge devices contributing to FEL, the participation of some devices lacking computing and communication capabilities or training data can negatively affect the training performance of FEL.

Beyond the server-oriented optimization, many researchers have tried to improve FEL from the perspective of devices, which mainly includes three types of studies, i.e., such as *model optimization* [9]–[11], *resource balance* [12] and *device selection* [13]–[16]. Among them, model optimization may sacrifice the accuracy of the final model to a certain extent, while resource balance usually assigns extra resources to low-quality participants, which can be unfair for other high-quality devices and thus discouraging their future participation. More importantly, involving all available devices in the learning process in the first two types of schemes can heavily degrade the overall performance of FEL systems. Thus, device selection becomes a better solution under the premise of balancing resource consumption and efficiency. In fact, there are several recent works [13]–[16] focusing on device selection for FEL. But they fail to include the impact of devices' energy cost on device selection, except for time consumption and data diversity. As the power of edge devices is usually provided by built-in batteries, excessive energy consumption of participated devices may cause unanticipated dropouts and significantly lower the cost-efficiency of FEL.

To overcome the shortcomings of the existing device selection mechanisms, we consider minimizing the energy consumption of all selected devices in each round of training, which is under the constraints of communication and computing time consumption of each device, as well as the total amount of data for training. In addition, the number of devices selected for FEL in each round is also designed to be maximized so as to increase the diversity of data and thus accelerate the speed of model convergence.

However, it is nontrivial to solve the above minimization problem and the time complexity of intuitively solving this problem increases exponentially with the number of devices, which imposes a burden on the server to determine the

This is the author's manuscript of the article published in final edited form as:

Peng, C., Hu, Q., Chen, J., Kang, K., Li, F., & Zou, X. (2021). Energy-Efficient Device Selection in Federated Edge Learning. 2021 International Conference on Computer Communications and Networks (ICCCN), 1–9. https://doi.org/10.1109/ICCCN52240.2021.9522303

appropriate set of devices in each FEL round and can further reduce the training efficiency of FEL.

In this paper, we solve the above challenge by defining an optimization problem considering the time and energy consumption in the communication and calculation phases, as well as the number of selected devices and the local data sizes of devices. Afterwards, the optimization problem is solved by designing an efficient algorithm named $E^2DS$ with lower time complexity. The main contributions of this work are summarized as follows:

- By elaborating the FEL procedures, we insert a device selection step in the traditional FL process, which is determined by the server through solving an optimization problem.
- Considering the limited waiting time constraint of the server and the required size of training data in FEL, we establish a mathematical model to minimize the total energy consumption of selected devices in the communication and computing processes, as well as maximize the number of selected devices for data diversity consideration.
- To reduce the calculation complexity of the server in the stage of device selection, we reformulate the proposed optimization problem and propose an efficient solution, namely $E^2DS$, that achieves the optimization goal but brings no negative impact to the system performance of FEL.
- We implement a series of experiments to evaluate the performance of our proposed $E^2DS$ scheme via investigating time and communication cost. Besides, by comparing with two classical methods, we demonstrate that $E^2DS$ can perform better to save device-average energy consumption while achieving great learning performance, i.e., high accuracy with fast convergence.

The rest of this paper consists of the following five sections. Section II summarizes the most related work about optimizing FEL. Section III introduces our system model and problem formulation, which is reformulated and solved in Section IV. Then we evaluate our proposed scheme and present experimental results in Section V. Section VI concludes the whole paper.

## II. Related Work

As the applications of FL at the edge become popular, numerous studies have been conducted to improve FEL performance. The existing research on FEL optimization can be roughly divided into two categories, *server-oriented optimization* and *device-oriented optimization*.

Many researchers worked on server-oriented optimization through *resource allocation* [1]–[3], *intermediate server assistance* [4], [5] and *device participation incentive* [6]–[8]. Yang *et al.* [1] optimized communication bandwidth allocation to accelerate global model aggregation via superposing wireless multiple-access channels. Abad *et al.* [2] targeted at reducing communication delay by introducing mobile base stations for resource allocation. Further, Ren *et al.* [3] studied the strategy design using deep reinforcement learning for allocating communication and computing resources to reduce transmission cost during FEL. By introducing intermediate edge servers, an algorithm employing multiple edge servers to aggregate models were proposed in [4] to speed up the convergence of FL, while Ye *et al.* [5] proposed setting up intermediate edge servers for small-scale aggregation to reduce both total communication and computing cost of mobile devices. Besides, through incentive mechanism designs, researchers in [6] and [7] studied the criteria for measuring the contribution of devices to global model updates and motivating devices to participate in alliance learning. Hu *et al.* [8] proposed a device participation decision strategy based on the correlated equilibrium to maximize both the individual and global profits.

As the server-oriented solutions, like resource allocation, will cause extra server load, and the assistance of the intermediate server requires additional stations to be built, many researchers have studied device-oriented optimization schemes in FEL, such as *ML model optimization* [9]–[11], *resource balance* [12] and *device selection* [13]–[16]. Xu *et al.* [9] studied training data preprocessing by introducing the CNN model into FEL to improve learning performance. Yu *et al.* [10] considered reducing training rounds using the proactive content caching algorithm to improve training efficiency. In addition, a model pruning algorithm was proposed in [11] to adjust the model size and reduce training time for devices. Zeng *et al.* [12] investigated resource so as to guarantee that all devices can submit local model updates using similar time cost to device condition, where the weaker channel status and computing capability would be provided with more bandwidth so as to guarantee that all devices can submit local model updates using similar time cost. Based on device selection, Wu *et al.* [13] adjusted the selection set of devices by comparing the efficiency of previous local model training of devices, while Amiri *et al.* [14] considered the channel conditions of devices and the importance of local model updates from each device. The time consumption in local computing and communication was the criteria for device selection in [15], and Zhang *et al.* [16] studied to reduce the impact of non-IID data on FEL performance through device selection.

However, due to the complex environment of edge devices and servers, the aforementioned studies only consider single factors affecting FEL performance, i.e., time consumption or data adversity. In addition, as an indispensable resource, energy consumption is rarely considered in the existing studies, which will lower the cost-efficiency of FEL. To fill the gaps, we propose a comprehensive device selection scheme that takes into account the energy and time consumption in both local calculation and communication process, as well as the diversity of training data. Through the device selection process before each round of learning, the total energy consumed during the FEL process will be significantly reduced with participated devices as many as possible.

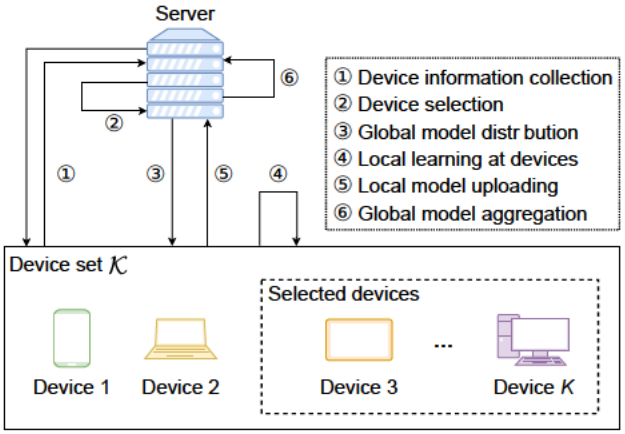



Fig. 1. The working process of an FEL round.

## III. System Model

In this paper, we consider an FEL system, consisting of one edge server and multiple devices that have been activated on the edge server. We denote the set of devices as $\mathcal{K} = \{1, 2, \cdots, k, \cdots, K\}$. Considering that numerous devices could exist, a subset of devices will be selected to participate in each round of FEL for both time and communication efficiency. For simplicity, we define a binary variable $x_k \in \{0, 1\}$ to indicate whether device $k$ is selected or not, where $x_k = 1$ denotes that device $k$ is chosen for local training in this FEL round while $x_k = 0$ means not being picked.

In this case, how to optimize the selection of devices based on the cost-efficiency criteria becomes an essential step to improve the system performance. To that aim, we consider selecting devices with a trade-off between resource consumption and learning performance in this paper, where the selected devices can provide the required number of data samples for model training and satisfy the time limitation of local training and updating while consuming the minimum amount of total energy.

In this section, we model the optimal device selection problem via elaborating the overall FEL working process with device selection considering resource consumption formulating an optimization problem with constraints.

### A. FEL Process

We first define a waiting time limitation, denoted as $T^{wait}$, for the server to collect local model updates from selected devices in each round, which is usually a pre-defined constant parameter for a certain FEL task. Then we present the FEL interaction process between the edge server and candidate devices during each round in Fig. 1, involving six main steps as described below.

① **Device Information Collection.** To determine the best subset of devices satisfying the basic time limitation $T^{wait}$ and other additional requirements, the server needs to collect necessary information from devices, including both computation and communication related parameters, such as local data size, CPU computing power, CPU-cycle frequency, bandwidth and transmission power.

② **Device Selection.** After the server receives requested information from all devices or reaches the maximum waiting time, the server will start to calculate the best selection plan of appropriate devices for optimizing system performance, which turns out to be the research focus of this work and will be solved in the next section.

③ **Global Model Distribution.** After the device selection step, the server sends the parameters of the global model to all selected devices.

④ **Local Learning at Devices.** Once devices receive parameters from the server, they will use their local data to train the local model.

⑤ **Local Model Uploading.** When the local model converges, each device sends the parameter updates back to the server.

⑥ **Global Model Aggregation.** Even some of the devices may fail to submit local updates within $T^{wait}$ due to networking condition, the server aggregates the received results to derive the updated global model.

The above steps will be repeated for multiple rounds until the maximum number of FL rounds or the required model performance is reached.

### B. Time Consumption Model

As we mentioned above, in order to achieve efficient FEL, the server needs to select a subset of devices that can upload their locally learned model updates within $T^{wait}$ instead of endless waiting. As we can see from Fig. 1, the main time consumption for devices in each round of FEL is spent for steps ③ to ⑤, which will be specifically calculated in this subsection. It is worthy noting that since other steps, i.e., steps ①, ②, and ⑥, are mainly operated by the server with relatively powerful computing and communication capabilities, which is clearly independent of devices, so we skip discussing the time consumption during these steps.

Steps ③ - ⑤ describe the parameters updating process of devices. We denote the time spent of device $k$ in the step of global model distribution by $T_k^D$ and that in the step of local model uploading by $T_k^U$, which are generally determined by the value of model parameter size divided by the transmission speed and can be respectively calculated as:

$$T_k^D = \frac{D^p}{V_k^D},$$

$$T_k^U = \frac{D^p}{V_k^U}.$$

In the above equations, $D^p$ is the size of model parameters; $V_k^D$ and $V_k^U$ are respectively the transmission speeds of downloading and uploading that are closely related to the condition of communication channels and can be defined as:

$$V_k^D = B_k^D \log(1 + \frac{P_k h_k^2}{N_0}), \tag{1}$$

$$V_k^U = B_k^U \log(1 + \frac{P_k h_k^2}{N_0}). \quad (2)$$

In (1) and (2), $B_k^D$ and $B_k^U$ are respectively the download and upload bandwidth of device $k$, $P_k$ and $h_k$ are respectively the transmission power and the channel gain of device $k$ [15], and $N_0$ is the background noise. Specifically, $B_k^D$, $B_k^U$ and $N_0$ are constant parameters, while $P_k$ and $h_k$ are device information provided by each candidate device in the step of device information collection.

We denote $c_k$ as the number of CPU cycles for each device $k$ to complete training one sample of data, which can be measured locally and reported to the server. And the total number of CPU cycles needed for device $k$ in each FEL round is $c_k D_k$ with $D_k$ denoting the size of its local dataset. Given the CPU-cycle frequency of device $k$ denoted by $f_k$, the time spent in the local learning step of device $k$, denoted by $T_k^{LC}$, can be calculated as [17]:

$$T_k^{LC} = \frac{c_k D_k}{f_k}.$$

Combining the time spent in each step, the total time consumption of device $k$ in each FEL round, denoted as $T_k$, can be calculated as:

$$T_k = T_k^D + T_k^{LC} + T_k^U.$$

### C. Energy Consumption Model

Similar to the time consumption model defined above, steps ③ to ⑤ consumes the energy of any device mainly in each round of FEL, which will be calculated in this subsection. In the rest of steps, the energy is mainly consumed for the server. Since it is not closely related to devices, and the energy supply of the server is sufficient, we skip to discuss the energy consumption during these steps.

Since the energy consumption during transmission is the product of transmission power and transmission time, we can calculate the energy consumption of device $k$ in the step of global model distribution and local model uploading, denoted by $E_k^D$ and $E_k^U$, respectively, as follows:

$$E_k^D = T_k^D P_k,$$

$$E_k^U = T_k^U P_k.$$

In the step of local learning at devices, the energy consumed of device $k$, denoted as $E_k^{LC}$, is closely related to the data amount and CPU frequency, which is expressed as [18]:

$$E_k^{LC} = \frac{\alpha_k}{2} c_k D_k f_k^2.$$

In the above equation, $\alpha_k$ is the effective capacitance coefficient of the computing chip-set in device $k$.

Thus, the overall energy consumption for device $k$, denoted as $E_k$, in each FEL round can be expressed as:

$$E_k = E_k^D + E_k^{LC} + E_k^U. \quad (3)$$

### D. Problem Formulation

Considering that edge devices are usually battery-powered with limited energy supply, we aim to minimize the total energy consumption of all participated devices so as to avoid unexpected device dropouts for improving the cost-efficiency of FEL. According to (3), the total energy consumption of selected devices can be expressed as $\sum_{k=1}^{K} E_k x_k$.

Besides, the number of participated devices is also an important parameter affecting the overall performance of FEL. In a real federated learning scenario, the quality and quantity of data from different devices can be highly diverse, and the data samples usually cannot meet the assumptions of IID. If the data distribution is severely skewed (i.e, non-IID), the accuracy of the training results will be severely reduced [19]. Thus, in the device selection stage, it is significant to increase the number of devices for improving the diversity of training data, so as to increase the accuracy of the finally trained model. Given $x_k$ denoting the state indicator of each selected device, the number of devices can be calculated by $\sum_{k=1}^{K} x_k$.

As we mentioned at the beginning of this section, there is a time limitation $T^{wait}$ for the server to collect local model updates, which should be a common constraint for all devices. In addition, the total amount of data for training affects the performance of FEL where excessive or extremely few data will affect the convergence speed of the training process and the model accuracy. Here we employ a parameter $a$ to depict the ratio of the required amount of data in FEL to the total amount of data $D_{all} = \sum_{k=1}^{K} D_k x_k$.

To achieve the above goals, we formulate an optimization problem as follows:

$$\min: \ \eta \sum_{k=1}^{K} E_k x_k - \theta \sum_{k=1}^{K} x_k, \quad (4)$$

$$\text{s.t.}: \ x_k \in \{0, 1\}, \quad (4a)$$

$$T_k \leq T^{wait}, \quad (4b)$$

$$\sum_{k=1}^{K} D_k x_k \geq a \cdot D_{all}. \quad (4c)$$

In the above formulation, the objective (4) is to minimize the difference between the total energy consumption and the total number of selected devices, with $\eta$ and $\theta$ for value balancing. Constraint (4a) ensures that the state indicator should only be 1 for being selected, or 0 for not being selected. Constraint (4b) states that the overall time consumption of each selected device should be less than the maximum waiting time $T^{wait}$ set by the server. And the last constraint (4c) guarantees that the overall data amount of selected devices has to be enough for training the model in each round.

## IV. Algorithm Design for Energy-Efficient Device Selection

In this section, we design a specific algorithm to solve the optimization problem defined in (4), which consists of the problem reformulation in Section IV-A and detailed algorithm design in Section IV-B.

### A. Problem Reformulation

From (4), we see that the optimization problem is to find the best subset of devices that minimizes energy consumption and maximizes the number of selected devices, under the constraints of time consumption and training data size. Solving the above optimization problem is nontrivial as it requires a complex combinatorial optimization where the difference between the total energy consumption and the number of selected devices needs to be minimized. An intuitive solution of this problem is to traverse all combinations of devices and then compare them to obtain the optimal one. Since each device has two states of being selected and not being selected, it costs the time complexity of $O(2^K)$ to traverse $K$ devices in total, which will undoubtedly cause a huge time consumption in each FEL round and seriously affect the training efficiency. Therefore, we transform the problem defined in (4) to an efficient maximization problem and propose a solution based on dynamic programming with a lower time complexity.

For simplicity, we denote $\mathcal{K}_1$ as the set of selected devices and $\mathcal{K}_0$ as the set of unselected devices, where $\mathcal{K}_1 \cap \mathcal{K}_0 = \varnothing$ and $\mathcal{K}_1 \cup \mathcal{K}_0 = \mathcal{K}$. Obviously, the size of $\mathcal{K}_1$, denoted as $K_1$, can be calculated by:

$$K_1 = \sum_{k=1}^{K} x_k,$$

and the size of $\mathcal{K}_0$, denoted as $K_0$, can be calculated by:

$$K_0 = \sum_{k=1}^{K} (1 - x_k). \tag{5}$$

Clearly, the state indicator of device $k$ in $\mathcal{K}_0$ is $x_k = 0$. Then, we can use $y_k$ to indicate the state of device $k$ which is not selected for joining FEL in this round, so we have $y_k = 1 - x_k$. In this case, (5) can be rewritten as:

$$K_0 = \sum_{k=1}^{K} y_k.$$

Next, we respectively denote the sum of energy consumption of devices in set $\mathcal{K}_1$ and that in set $\mathcal{K}_0$ as $E_1$ and $E_0$. According to the definitions of two sets mentioned above, $E_1$ and $E_0$ are calculated as:

$$E_1 = \sum_{k=1}^{K} E_k x_k,$$

$$E_0 = \sum_{k=1}^{K} E_k (1 - x_k) = \sum_{k=1}^{K} E_k y_k.$$

Then the total energy consumption of all devices, denoted as $E_{all}$, is the sum of $E_1$ and $E_0$, which is expressed as:

$$E_{all} = E_1 + E_0 = \sum_{k=1}^{K} E_k x_k + \sum_{k=1}^{K} E_k y_k. \tag{6}$$

In the above equation, $E_{all}$ is clearly a fixed value in each FEL round.

In addition, the total data amount $D_{all}$, which appears in one of the constraints in optimization problem (4), can also be rewritten according to the above-defined two device sets $\mathcal{K}_1$ and $\mathcal{K}_0$. Specifically, the data amounts of devices in $\mathcal{K}_1$ and $\mathcal{K}_0$ is respectively denoted by $D_1$ and $D_0$, which are calculated as follows:

$$D_1 = \sum_{k=1}^{K} D_k x_k,$$

$$D_0 = \sum_{k=1}^{K} D_k y_k,$$

and the total amount of data $D_{all}$ is the sum of $D_1$ and $D_0$, which is expressed as:

$$D_{all} = \sum_{k=1}^{K} D_k x_k + \sum_{k=1}^{K} D_k y_k. \tag{7}$$

Since $D_{all}$ is a fixed value that only related to the set of total devices $\mathcal{K}$ in each round, the constraint (4c) can be transferred to:

$$\sum_{k=1}^{K} D_k y_k \leq (1 - a) \cdot D_{all}. \tag{8}$$

According to (6) and the transformed constraint (8), the optimization problem defined in (4) can be rewritten into:

$$\begin{aligned}
\min: \ & (\eta E_{all} - \theta K) - (\eta \sum_{k=1}^{K} E_k y_k - \theta \sum_{k=1}^{K} y_k), && (9)\\
\text{s.t.}: \ & y_k \in \{0, 1\}, && (9a)\\
& T_k \leq T^{wait}, && (9b)\\
& \sum_{k=1}^{K} D_k y_k \leq (1 - a) \cdot D_{all}. && (9c)
\end{aligned}$$

As mentioned above, $\eta E_{all}$ and $\theta K$ are fixed values for each FEL round. Since we aim to minimize the difference between this fixed value $\eta E_{all} - \theta K$ and $\eta \sum_{k=1}^{K} E_k y_k - \theta \sum_{k=1}^{K} y_k$ in the optimization problem (9), it is equivalent to maximize $(\eta \sum_{k=1}^{K} E_k y_k - \theta \sum_{k=1}^{K} y_k)$. Thus, problem (9) can be reformulated as:

$$\begin{aligned}
\max: \ & \sum_{k=1}^{K} y_k (\eta E_k - \theta), && (10)\\
\text{s.t.}: \ & y_k \in \{0, 1\}, && (10a)\\
& T_k \leq T^{wait}, && (10b)\\
& \sum_{k=1}^{K} D_k y_k \leq (1 - a) \cdot D_{all}. && (10c)
\end{aligned}$$

### B. Algorithm Design

In the above, we introduce two fixed values and use the complementary relationship to transform the original minimization problem defined in (4) into a maximization problem defined in (9). To solve it, we propose an Energy-Efficient Device

Selection (E$^2$DS) algorithm which is specified in Algorithm 1.

Generally, we can see that the above maximization problem can be transferred to a 0-1 Knapsack problem. In the 0-1 Knapsack problem, there are a number of different items and a knapsack with limited capacity. Each item has its own value and can be selected to be put into the Knapsack or not, and the purpose is to find out which set of items being packed into the knapsack can maximize the total value under the capacity limitation.

Similarly, in our problem defined in (9), there are $K$ different devices that can be selected or not for joining FEL, the capacity is the maximum remaining data size $(1-a)\cdot D_{all}$ with an extra constraint of time consumption. For the sake of convenience, we denote $D^{cap}=(1-a)\cdot D_{all}$ as the data size capacity of unselected devices, and $v_k=\eta E_k-\theta$ as the value of device $k$. Then we denote a sequence $\mathcal{V}=\{v_1,v_2,\cdots,v_K\}$ to contain the values of all devices. The sequence of data size is denoted by $\mathcal{D}=\{D_1,D_2,\cdots,D_K\}$.

**Algorithm 1** E$^2$DS

**Input:** the value set of devices $\mathcal{V}=\{v_1,v_2,\cdots,v_K\}$, the size set of devices $\mathcal{D}=\{D_1,D_2,\cdots,D_K\}$, the number of devices $K$, the data size capacity of selected devices $D^{cap}$

**Output:** the state indicators of all devices for $\mathcal{K}_1$ $\{x_1,x_2,\cdots,x_K\}$

1: $\{y_1,y_2,\cdots,y_K\}\leftarrow UDD(\mathcal{V},\mathcal{D},K,D^{cap})$
2: **for** $i\leftarrow 1$ to $K$ **do**
3: $\quad x_i\leftarrow 1-y_i$
4: **end for**
5: **return** $\{x_1,x_2,\cdots,x_K\}$

As shown in Algorithm 1, there are four input parameters: the value set $\mathcal{V}$ presents the value of each device; the set of weight $\mathcal{D}$ presents the dataset size of each device; the parameter $K$ presents the number of devices in the edge; the parameter $D^{cap}$ presents the maximum data size. The set of state indicators of all devices $\{x_1,x_2,\cdots,x_K\}$ describes the result of device selection, where the state of device $k$ with $x_k=1$ will be selected to participate in FEL of this round. Specifically, we design a dynamic programming based algorithm named Unselected Device Decision ($UDD$), which will be detailed in Algorithm 2, to calculate the set of unselected devices (Line 1). Then, by traversing devices from 1 to $K$, all the indicators of devices will be converted (Lines 2 - 4) to meet our objective in (4).

In Algorithm 2, a two-dimensional array $DSA(K,D^{cap})$ is defined to store the intermediate results of dynamic programming. After initializing the array $DSA(K,D^{cap})$ (Lines 1 - 6), we complete the calculation of $DSA(K,D^{cap})$ (Lines 7 - 19). By traversing devices from 1 to $K$, and data size from 1 to $D^{cap}$, $DSA(i,D)$ will be filled in with the calculated optimal solution. After the traversal is completed, the value of $DSA(K,D^{cap})$ is the solution of the entire optimization problem. Then, we update the value of state indicator of

**Algorithm 2** $UDD$

**Input:** the value set of devices $\mathcal{V}=\{v_1,v_2,\cdots,v_K\}$, the size set of devices $\mathcal{D}=\{D_1,D_2,\cdots,D_K\}$, the number of devices $K$, the data size capacity of selected devices $D^{cap}$

**Output:** the state indicators of all devices for $\mathcal{K}_0$ $\{y_1,y_2,\cdots,y_K\}$

1: **for** $D\leftarrow 0$ to $D^{cap}$ **do**
2: $\quad DSA(0,D)\leftarrow 0$
3: **end for**
4: **for** $i\leftarrow 1$ to $K$ **do**
5: $\quad DSA(i,0)\leftarrow 0$
6: **end for**
7: **for** $i\leftarrow 1$ to $K$ **do**
8: $\quad$ **for** $D\leftarrow 1$ to $D^{cap}$ **do**
9: $\quad\quad$ **if** $D_i\le D$ **then**
10: $\quad\quad\quad$ **if** $v_i+DSA(i-1,D-D_i)>DSA(i-1,D)$ **then**
11: $\quad\quad\quad\quad DSA(i,D)\leftarrow v_i+DSA(i-1,D-D_i)$
12: $\quad\quad\quad$ **else**
13: $\quad\quad\quad\quad DSA(i,D)\leftarrow DSA(i-1,D)$
14: $\quad\quad\quad$ **end if**
15: $\quad\quad$ **else**
16: $\quad\quad\quad DSA(i,D)\leftarrow DSA(i-1,D)$
17: $\quad\quad$ **end if**
18: $\quad$ **end for**
19: **end for**
20: **for** $i\leftarrow K$ to 1 **do**
21: $\quad$ **if** $DSA(i,D^{cap})>DSA(i-1,D^{cap})$ **then**
22: $\quad\quad y[i]\leftarrow 1$
23: $\quad\quad D^{cap}\leftarrow D^{cap}-D[i]$
24: $\quad$ **else**
25: $\quad\quad y[i]\leftarrow 0$
26: $\quad$ **end if**
27: **end for**
28: **return** $\{y_1,y_2,\cdots,y_K\}$

all devices (Lines 20 - 27). Since the final result is already calculated, whether each device is selected or not can be found by looking up the value of the two-dimensional array $DSA$ in reverse. For example, if $DSA(i,D)=DSA(i-1,D)$, it means choosing or not choosing device $k$ leads to the same optimization result, and then we can know that device $k$ should not be selected to form set $\mathcal{K}_0$, i.e., $y_k=0$ (Line 22). Otherwise, device $k$ should be selected to form set $\mathcal{K}_0$, i.e., $y_k=1$ (Line 25). Finally, after traversing all devices, set $\{y_1,y_2,\cdots,y_K\}$ will be returned to Algorithm 1 (Line 28).

Overall, after obtaining the optimal result in Algorithm 2 and converting it into indicators in Algorithm 1, we can assign all the values in $\{x_1,x_2,\cdots,x_K\}$ as the states of each device, where any device $k$ with $x_k=1$ is the one the server would like to choose to participate in this round of FEL.

Since there are many methods to prove the correctness of applying dynamic programming to solve the 0-1 Knapsack

problem [20], such as the proof by contradiction, optimization problem (10) is also able to be well solved by Algorithm 1. The time complexity of the algorithm is $O(KD^{cap})$, which is obviously less than that of solving it by a brute-force search with the time cost of $O(2^K)$.

## V. Experimental Evaluation

To evaluate the effectiveness and efficiency of our proposed optimization scheme E$^2$DS, we simulate the environment of FEL and perform experimental verification. We also simulate the traditional FL (TFL) process without device selection and the device selection scheme named FedCS in [15]. All three schemes are implemented on a desktop with Intel(R) Core(TM) i7-9750 CPU @2.60GHz and 16GB RAM running Windows 10 OS.

### A. FEL Environment Simulation

We establish a simulated edge computing environment to employ the FEL training process, with $K = 100$ to complete the local learning and a server for device selection as well as model aggregation.

For the wireless communication simulation, a circular area with a radius of 50 meters is used as the covered area of the edge server for the experiment, with the edge server located at the center of the area. Devices are uniformly distributed with a range of 2 meters to 50 meters from the center of the circle. The channel gain $h_k$ of device $k$ follows the exponential distribution with the equation $g_0(d_0/d)^4$, where the reference distance $d_0 = 1$ meter, and $g_0 = -40$ dB [18]. We assume the download bandwidths of devices $B_k^D$ follow the normal distribution with the mean and standard deviation being 5 MHz and 4 MHz, and as a practical bandwidth limitation, the upload bandwidth of devices $B_k^U$ would be lower than the download bandwidth, which follows the normal distribution with the mean and standard deviation of 1 MHz and 0.1 MHz. In addition, we set the transmission power $P_k$ as a normal distribution where the mean is 0.6 W and the standard deviation is 0.2 W, and the background noise $N_0 = 10^{-8}$ W. Furthermore, the data size of model parameters is set as $D^p = 25,000$ nats, which is approximately equal to 4.5 KB.

For the local learning step, the training size $D_k$ of each device is set as a normal distribution with the mean and standard deviation being 5 MB and 4 MB, effective capacitance coefficient $\alpha_k = 2 \times 10^{-28}$, the number of CPU cycles $c_k$ is normally distributed with the mean of 15 cycles/bits and the standard deviation of 10 cycles/bits, and the CPU-cycle frequency $f_k$ follow a normal distribution with the mean of 0.5 GHz and the standard deviation of 0.1 GHz.

### B. Model Training Settings

To verify the accuracy and efficiency of model training in the FEL environment, we use the MNIST dataset to complete the classification task in this subsection. By comparing the convergence speeds and accuracy of training in different FEL schemes, we can analyze the influence of device selection on FEL performance.

MNIST contains 70,000 handwritten digital images with 10 classes. In our experiments, we set 60,000 of the images for training and 10,000 of those for testing. The dataset is already preprocessed that every image in MNIST is gray-scale with 28×28 pixel, with the handwritten numbers displayed in the center. It is closer to the actual situation as the preprocessing process of the FEL can be completed in advance to speed up FEL.

We compare our proposed E$^2$DS algorithm with the other two schemes, i.e., TFL and FedCS [15]. Specifically, the first model randomly selects devices for training. FedCS uses a greedy algorithm to select as many devices as possible at the beginning of each round of training.

For all these schemes, all candidate devices are assigned with the number of $D_k$ images for local training. To simulate the FEL scenario with non-IID data distribution, we distribute 68.2% of data from the same class to each device, and randomly select 31.8% of data from the remaining classes.

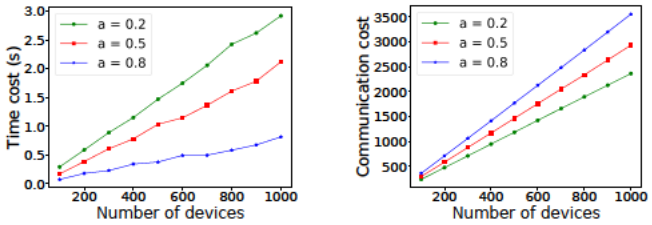
We build a convolutional neural network as the global model, consisting of three linear convolution layers (the first with 32 channels, the second with 64 channels, the third with 128 channels, and each followed with 2×2 max pooling), each of which is activated by the ReLU function, and a final Softmax output layer afterwards.

Then, we set the ratio of the necessary amount of data to the total amount of data for each round of FEL as $a = 0.75$ unless otherwise specified, which means there are three-quarters of the data used in each FEL round. In addition, the maximum waiting time in each round $T^{wait}$ is set as 3 min, 5 min and 10 min. For the TFL scheme, as there is no time limitation, we only set the same amount of data for training. For the FedCS scheme, as there is no data amount limitation, we only set the same time limitation. While for our proposed E$^2$DS scheme, we set the weight of energy consumption as $\eta = 3$ and that of the device number as $\theta = 1$. Other sets of weight parameters are also examined, which return similar changing trends, so we omit them to avoid redundancy.

### C. Evaluation Results

In this subsection, we evaluate the time cost and communication cost of the E$^2$DS algorithm in Section V-C1, which verifies the usability and effectiveness of our proposed scheme. Then, Section V-C2 verifies that the E$^2$DS algorithm performs better by evaluating the data diversity and the energy consumption of the three schemes. Finally, in Section V-C3 we evaluate the accuracy and the speed of convergence comparing in three schemes.

*1) Time and Communication Cost:* To test the efficiency of the device selection algorithm, Fig. 2 shows the time consumption and communication cost when selecting different numbers of devices. In Fig. 2(a), the time consumption increases approximately linearly with the increased number of devices. The growth trend is in line with the time complexity $O(KD^{cap})$ of Algorithm 1 as we mentioned earlier. Specifically, the time consumption is larger when $a = 0.2$ and smaller when $a = 0.8$, due to the inversely-proportional

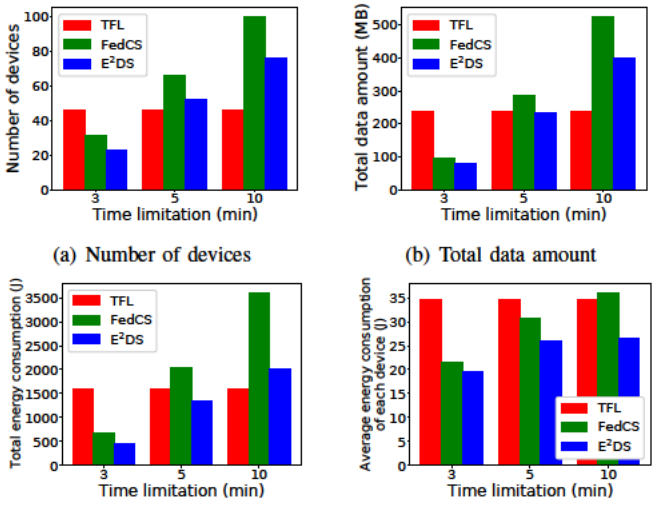



(a) Time cost (b) Communication cost

Fig. 2. The time and communication cost of E$^2$DS.

relationship between the data capacity $D^{cap}$ and $a$ as shown in $D^{cap} = (1-a) \cdot D_{all}$. In contrast, as Fig. 2(b) implies, there is a positive correlation between the number of devices and the communication cost, because more devices indicate more necessary information to be collected. At the same time, $a$ and the communication cost are also positively correlated since a higher $a$ indicates more data for training and hopefully more devices being selected, leading to more communication cost.

*2) Comparison of Device Selection Schemes:* To explore the efficiency of E$^2$DS scheme, we study the performances including the number of devices, total data amount, total energy consumption and energy consumption per device of E$^2$DS scheme, TFL scheme and FedCS scheme with different time limitations $T^{wait}$.




(a) Number of devices (b) Total data amount

(c) Total energy consumption (d) Average energy consumption of each device

Fig. 3. The comparison results of the number of devices, total data amount, total energy consumption and energy consumption per device in different schemes.

In a practical FEL environment, we assume that more devices mean that the model has more diverse data. Therefore, if more devices participating in training, the model will reach better performance. As shown in Fig. 3(a), the number of selected devices does not change with $T^{wait}$ in TFL scheme. Since TFL scheme does not estimate the time consumption of devices in each round of learning, but the total amount of data is limited to $a \cdot D_{all}$. In contrast, E$^2$DS scheme is restricted by $T^{wait}$ when selecting devices. When $T^{wait} = 3$ min, fewer devices can be selected; when $T^{wait} = 5$ min and $T^{wait} = 10$ min, the number of devices selected by E$^2$DS scheme exceeds that of TFL scheme. Besides, it is meaningless to compare the number of devices selected with FedCS scheme, because it selects as many devices as possible due to the greedy algorithm.

TABLE I
COMPARISON OF LEARNING RESULTS.

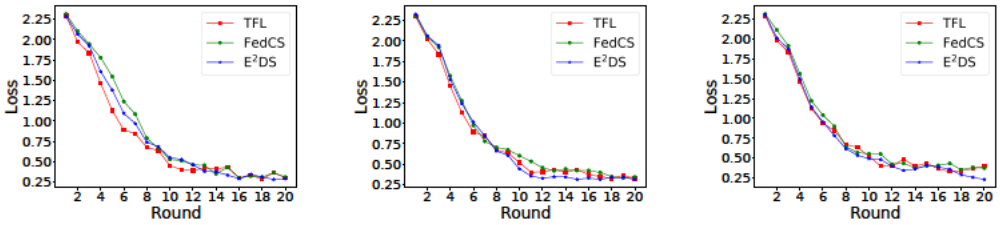
| Method | $T^{wait}$ | Accuracy | | |
|---|---|---|---|---|
| | | Round=5 | Round=10 | Round=20 |
| TFL | - | 0.69 | 0.75 | 0.85 |
| FedCS | 3 min | 0.62 | 0.81 | 0.92 |
| | 5 min | 0.72 | 0.81 | 0.93 |
| | 10 min | 0.72 | 0.85 | 0.94 |
| E$^2$DS | 3 min | 0.68 | 0.88 | 0.93 |
| | 5 min | 0.77 | 0.88 | 0.94 |
| | 10 min | 0.73 | 0.89 | 0.94 |

Regarding the total data amount shown in Fig. 3(b), according to the experimental settings, the upper limit of the total data amount of the devices participating in E$^2$DS scheme is $a \cdot D_{all}$. When $T^{wait} = 3$ min or $T^{wait} = 5$ min, the choice of devices is mainly limited by $T^{wait}$. Compared with TFL scheme, in $T^{wait} = 3$ min, though the total data amount in E$^2$DS scheme is less than that of TFL scheme, E$^2$DS scheme selects more devices. Besides, since $T^{wait}$ is the only restriction, the total data amount increases as $T^{wait}$ becomes larger in the FedCS scheme.

As for the total energy consumption shown in Fig. 3(c), when $T^{wait} = 3$ min and $T^{wait} = 5$ min, E$^2$DS scheme consumes the least energy. In comparison, the energy consumption of TFL scheme is 1.3-3.6 times that of E$^2$DS scheme, while the energy consumption of the FedCS scheme is 1.5-1.8 times that of E$^2$DS scheme. In particular, when $T^{wait} = 10$ min, the energy consumptions of TFL scheme and E$^2$DS scheme are less than 2000 J, while the FedCS scheme consumes nearly 3500 J. This is because FedCS scheme lacks restriction on device selection in the same task, and a load of useless energy consumption is generated.

For the average energy consumption of each device, which is the most intuitive aspect to reflect the effectiveness of the scheme on energy consumption, Fig. 3(c) displays that the average energy consumption of E$^2$DS scheme is the lowest in all cases. Compared with TFL scheme, the energy consumption reduced by 30%-50%, while there is also 20%-30% reduction compared with FedCS scheme.

*3) Comparison of Learning Results:* In this subsection, we compare the learning results of E$^2$DS scheme, TFL scheme and FedCS scheme by changing the value of the time limitation as well as the number of FEL rounds. Table I implies the results of the accuracy of the three schemes after training. All schemes can achieve an accuracy of over 85% after 20 rounds of training. Specifically, FedCS and E$^2$DS schemes can increase the accuracy to 94%. Especially for E$^2$DS scheme, the




(a) $T^{wait}$ = 3 min (b) $T^{wait}$ = 5 min (c) $T^{wait}$ = 10 min

Fig. 4. The convergence trends of the loss function over training data in different schemes.

accuracy is higher than 85% after the fifth round of training, achieving a higher classification accuracy than FedCS and TFL schemes.

Furthermore, in order to show the convergence difference of three schemes in the training process, the convergence trends of loss function over training data are shown in Fig. 4. It indicates that when $T^{wait} = 3$ min, the convergence speed of TFL scheme is faster in the first ten rounds, but the convergence trend becomes roughly the same as the number of rounds grows. When $T^{wait} = 5$ min and $T^{wait} = 10$ min, the convergence trends of the three schemes are almost the same. Referencing to Fig. 3(a), there are more devices participating in TFL scheme when $T^{wait} = 3$ min, which accelerates the convergence. When $T^{wait} = 5$ min and $T^{wait} = 10$ min, the number of devices is no longer a bottleneck that limits the convergence speed, then three schemes show the same convergence trends.

## VI. Conclusion

In this paper, we study the process of FEL in detail and consider the device selection problem optimizing both energy consumption and data diversity, under the constraints of total time consumption and training data size of participated devices in each round. Due to the exponential time complexity of directly solving the optimization problem, we reformulate it to devise an efficient solution for achieving greatly reduced time complexity. Extensive experiments based on simulation and a real-world dataset demonstrate the effectiveness and better performance of our proposed scheme than the other two classical FEL schemes.

## References

[1] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 2022–2035, 2020.

[2] M. S. H. Abad, E. Ozfatura, D. Gunduz, and O. Ercetin, "Hierarchical federated learning across heterogeneous cellular networks," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 8866–8870.

[3] J. Ren, H. Wang, T. Hou, S. Zheng, and C. Tang, "Federated learning-based computation offloading optimization in edge computing-supported internet of things," *IEEE Access*, vol. 7, pp. 69 194–69 201, 2019.

[4] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[5] Y. Ye, S. Li, F. Liu, Y. Tang, and W. Hu, "Edgefed: optimized federated learning based on edge computing," *IEEE Access*, vol. 8, pp. 209 191–209 198, 2020.

[6] L. U. Khan, S. R. Pandey, N. H. Tran, W. Saad, Z. Han, M. N. Nguyen, and C. S. Hong, "Federated learning for edge networks: Resource optimization and incentive mechanism," *IEEE Communications Magazine*, vol. 58, no. 10, pp. 88–93, 2020.

[7] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 700–10 714, 2019.

[8] Q. Hu, F. Li, X. Zou, and Y. Xiao, "Correlated participation decision making for federated edge learning," in *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE, 2020, pp. 1–6.

[9] Z. Xu, Z. Yang, J. Xiong, J. Yang, and X. Chen, "Elfish: Resource-aware federated learning on heterogeneous edge devices," *arXiv preprint arXiv:1912.01684*, 2019.

[10] Z. Yu, J. Hu, G. Min, H. Lu, Z. Zhao, H. Wang, and N. Georgalas, "Federated learning based proactive content caching in edge computing," in *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2018, pp. 1–6.

[11] Y. Jiang, S. Wang, B. J. Ko, W.-H. Lee, and L. Tassiulas, "Model pruning enables efficient federated learning on edge devices," *arXiv preprint arXiv:1909.12326*, 2019.

[12] Q. Zeng, Y. Du, K. Huang, and K. K. Leung, "Energy-efficient radio resource allocation for federated edge learning," in *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2020, pp. 1–6.

[13] W. Wu, L. He, W. Lin, and R. Mao, "Accelerating federated learning over reliability-agnostic clients in mobile edge computing systems," *IEEE Transactions on Parallel and Distributed Systems*, 2020.

[14] M. M. Amiri, D. Gündüz, S. R. Kulkarni, and H. V. Poor, "Update aware device scheduling for federated learning at the wireless edge," in *2020 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2020, pp. 2598–2603.

[15] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–7.

[16] W. Zhang, X. Wang, P. Zhou, W. Wu, and X. Zhang, "Client selection for federated learning with non-iid data in mobile edge computing," *IEEE Access*, vol. 9, pp. 24 462–24 474, 2021.

[17] T. D. Burd and R. W. Brodersen, "Processor design for portable systems," *Journal of VLSI signal processing systems for signal, image and video technology*, vol. 13, no. 2, pp. 203–221, 1996.

[18] N. H. Tran, W. Bao, A. Zomaya, N. M. NH, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 1387–1395.

[19] N. Mhaisen, A. Awad, A. Mohamed, A. Erbad, and M. Guizani, "Analysis and optimal edge assignment for hierarchical federated learning on non-iid data," *arXiv preprint arXiv:2012.05622*, 2020.

[20] P. Gilmore and R. E. Gomory, "The theory and computation of knapsack functions," *Operations Research*, vol. 14, no. 6, pp. 1045–1074, 1966.