An Online Learning Approach for Client Selection in Federated Edge Learning under Budget Constraint

Lina Su
School of Computer Science
Wuhan University
lina.su@whu.edu.cn

Ruiting Zhou
School of Computer Science
School of Cyber Science and Engineering
Wuhan University
ruitingzhou@whu.edu.cn

Ne Wang
School of Computer Science
Wuhan University
ne.wang@whu.edu.cn

Guang Fang
School of Computer Science
Wuhan University
fguang@whu.edu.cn

Zongpeng Li
Insititue for Network Sciences and Cyberspace
Tsinghua University
zongpeng@tsinghua.edu.cn

ABSTRACT

Federated learning (FL) has emerged as a new paradigm that enables distributed mobile devices to learn a global model collaboratively. Since mobile devices (a.k.a, clients) exhibit diversity in model training quality, client selection (CS) becomes critical for efficient FL. CS faces the following challenges: First, the client's availability, the training data volumes, and the network connection status are time-varying and cannot be easily predicted. Second, clients for training and the number of local iterations would seriously affect the model accuracy. Thus, selecting a subset of available clients and controlling local iterations should guarantee model quality. Third, renting clients for model training needs cost. It is necessary to dynamically administrate the use of the long-term budget without knowledge of future inputs. To this end, we propose a federated edge learning (FedL) framework, which can select appropriate clients and control the number of training iterations in real-time. FedL aims to reduce the completion time while reaching the desired model convergence and satisfying the long-term budget for renting clients. FedL consists of two algorithms: i) the online learning algorithm makes CS and iteration decisions according to historic learning results; ii) the online rounding algorithm translates fractional decisions derived by the online learning algorithm into integers to satisfy feasibility constraints. Rigorous mathematical proof reveals that dynamic regret and dynamic fit have sub-linear upper-bounds with time for a given budget. Extensive experiments based on realistic datasets suggest that FedL outperforms multiple state-of-the-art algorithms. In particular, FedL reduces *at least* 38% completion time compared with others.

ICPP '22, August 29–September 1, 2022, Bordeaux, France

ACM ISBN 978-1-4503-9733-9/22/08…$15.00
https://doi.org/10.1145/3545008.3545062

CCS CONCEPTS

• **Networks** → **Network algorithms**; • **Theory of computation** → *Theory and algorithms for application domains.*

KEYWORDS

federated learning, online learning, client selection, budget

ACM Reference Format:
Lina Su, Ruiting Zhou, Ne Wang, Guang Fang, and Zongpeng Li. 2022. An Online Learning Approach for Client Selection in Federated Edge Learning under Budget Constraint. In *ICPP '22: the 51th International Conference on Parallel Processing, August 29–September 1, 2022, Bordeaux, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3545008.3545062

1 INTRODUCTION

The rapid proliferation of mobile devices comes with an exponential increase of data, promoting recent intelligent applications. Traditional machine learning (ML) approaches require to collect enormous amounts of training data at a server to comprehensively analyze and exploit the data. Such centralized data aggregation and processing are not always feasible, due to data transmission overhead [23] and privacy concerns [10]. Federated learning (FL) [19] has emerged as a promising decentralized learning paradigm by enabling distributed devices to learn a global model collaboratively while preserving data privacy.

In FL, each device (*a.k.a*, a client) participates in global model training using its local data, and then sends model updates to the server. These model updates are aggregated to update the global model, which is distributed to devices for further updating. However, each client is heterogeneous by nature, dramatically impacting the local training quality. Aggregating low-quality model updates can significantly degrade the performance of global model, which has been validated in [20]. Correspondingly, client selection (CS) becomes increasingly essential for high-quality federated learning. In fact, CS still encounters technical challenges:

First, some clients are available due to the battery capacity, while others become unavailable over time. Thus, the available clients are time-varying. Also, the training data of each client is generally time-varying. Taking news recommendations as an example, users are interested in current hot events. However, user interests may change as other news emerges. In addition, the wireless communication between client and server can be asymmetric and unstable. Therefore, the communication status is dynamic. These uncertainties increase the difficulty of selecting the proper clients online and adjusting the selection in real-time. *Second*, as shown in existing studies [7, 19], model training needs thousands of computational iterations. In each iteration, all clients use their own data to update the local model via Gradient Descent for global model aggregation. The trained model should guarantee convergence, which is the most important for using FL. However, selecting a subset of available clients to maintain the convergence of local model and global model is very difficult, especially involving online control over the number of training iterations for each client. *Third*, leasing the clients for training would incur considerable monetary costs. Thus, given the long-term budget, spending strategically in the FL procedure is necessary. Improper spending may lead to premature budget exhaustion and abrupt termination of the FL procedure, significantly affecting model convergence. How to wisely manage the budget without knowledge of future inputs?

Existing research is insufficient to address the above challenges. Several interesting studies only concentrate on client selection [4, 5, 11, 21, 28], but they make a strong assumption that the decision-maker has 1-*lookahead*, which cannot apply to the time-varying and unpredictable inputs. Reinforcement learning (RL) has recently been employed to design the CS strategies [6, 12, 27, 30], while they lack theoretical guarantees. Others [2, 31, 32] investigate only aggregation control. The rest researches [14–17, 22] emphasize on the convergence analysis and do not consider the resource consumption. We will discuss this in detail in Sec.2.

To this end, we propose FedL, a federated edge learning framework to minimize the accumulated computational and communication time, satisfying the long-term budget and convergence requirements for the local and global model. FedL utilizes the relationship between the model convergence and the maximal number of training iterations, and also involves time-varying inputs, including the client's availability, the training data volume, and network bandwidth. Functionally, FedL can utilize the current clients and historical model information to jointly select clients and control local training iterations in an online way. Our contributions are summarized as follows:

- We model the target scenario of training FL models and reformulate the problem to tackle the above challenges. Precisely, the objective reflects the effects of first challenge. The relationship between the desired model convergence and the needed maximum number of training iterations is utilized to overcome the second challenge. Moreover, we estimate the lower bound of FL life cycle for a given budget to overcome the third challenge.
- We design an online learning-based framework, FedL, to solve the proposed minimization problem. FedL makes fractional decisions according to the previous model and current clients, then converts the fractional selection decisions into integer ones. In particular, FedL solves the problem through a convex-concave reformulation and modified gradient-decent-based method only based on the observed input rather than prior knowledge. Moreover, FedL can guarantee the expectation of a rounded integer equal to the corresponding fraction to satisfy the constraints.
- We conduct rigorous theoretical analysis of FedL. We prove that the online decisions of FedL can achieve sub-linear regret about the stopping time for a limited budget. The regret is defined as the expected difference between the delay caused by the online solution of FedL and the offline optimum. In addition, the iteration control strategy of FedL can ensure the convergence of the local model and global model. We should emphasize that such results are non-trivial and distinguish our work from most existing FL methods or online algorithms analyses.
- We carry out extensive evaluation on real-world training datasets, including Fashion-MNIST and CIFAR-10, to verify the practical performance of FedL. Results demonstrate that FedL outperforms multiple state-of-the-art algorithms when adapting to time-varying inputs. In particular, after the same training time, FedL can improve the accuracy by 2% to 15% on average compared with FedCS [21], FedAvg [19], and Pow-d [5]. FedL saves *at least* 38% completion time when reaching the same accuracy. Globally, FedL reveals the learning advantage over other alternatives in global performance and overall latency.

Roadmap. Sec.2 reviews related research and Sec.3 formulates the optimization problem. Sec.4 introduces our proposed federated edge learning algorithm. Sec.6 discusses the evaluation results and Sec.7 concludes the paper.

2 RELATED WORK

Due to the tremendous potential of FL, it has received significant attention recently, such as client selection [5, 21], convergence analysis [14, 17], privacy preservation [29], and incentive mechanism [33]. This section mainly discusses the following two aspects, which are highly relevant to our research.

FL Client Selection. Nishio and Yonetani [21] consider client heterogeneity and design a resource-aware algorithm named FedCS that chooses as many clients as possible to reduce the training time. Wang *et al.* [28] explore an orthogonal method to distinguish irrelevant updates from local models and then exclude the corresponding clients to improve communication efficiency. Considering that the training performance could be affected by the wireless network, Chen *et al.* [4] jointly manage the CS and wireless resource allocation to minimize the loss function. Huang *et al.* [11] explore a long-term fairness CS strategy by quantifying client participating rate, balancing learning efficiency and fairness. Cho *et al.* [5] design a power-of-choice CS strategy to handle the balance between the local loss and selection bias.

Xia *et al.* [30] exploit the multi-armed bandit technique and design an upper-confidence-bound CS policy. Wang *et al.* [27] exploit a double Deep Q-learning Network to choose suitable clients, aiming at counterbalancing the bias caused by heterogeneous data. Jiao *et al.* [12] evaluate the local data quality and capability and design a novel CS mechanism based on RL and reverse auction to maximize the social welfare. Deng *et al.* [6] comprehensively studies the importance of individual data-quality on aggregated model and explore a quality-aware CS framework empowered by the RL technique.

FL Convergence Analysis. The work of [16] is the first study to analyze the convergence of federated averaging (FedAvg) algorithm on heterogeneous data. Liu *et al.* [17] design Momentum Federated Learning by applying the Momentum Gradient Descent method in the local training to speed up the convergence. Karimireddy *et al.* [14] propose a stable and fast convergence algorithm SCAFFOLD that utilizes variance reduction to deal with the 'client-drift' caused by heterogeneous data. By re-parameterizing of FedAvg, Li *et al.* [15] explore FedProx that allows each client to execute a variable number of work and demonstrate FedProx can accelerate the convergence. Reddi *et al.* [22] design a series of adaptive optimizers, including ADAGRAD, ADAM, and YOGI, to improve the convergence.

Comparison with Related Work. The works of [4, 5, 11, 21, 28] assume that the decision-maker has 1-*lookahead* [3]. That is to say, the decision-maker knows the dynamic inputs about the current time at the beginning of the current time. However, in real scenarios, the decision-maker has 0-*lookahead.* Moreover, others [6, 12, 27, 30] cannot theoretically guarantee the model convergence. These studies of [14–17, 22] mainly concentrate on the convergence analysis, and directly neglect the resource consumption from an FL system perspective.

3 SYSTEM MODEL AND PROBLEM FORMULATION

3.1 System Model

System Overview. We consider an edge network with a centralized server (*e.g.,* gateway or base station) and M mobile clients with local data. We assume that all clients can directly access the server. The time is discretized into epochs indexed by $t \in \{1, 2, \ldots, T\}$. In an epoch, we train an FL model. Similar to existing studies [13, 25], FL is applied to wireless networks and runs as follows. In each epoch t, the FL process is further partitioned into multiple training iterations. Each iteration performs the following three sub-steps: each client first downloads the latest global model parameters from the server, then updates its model using the local data of the current epoch, and finally sends its model updates to the server. Table I lists relative important notations.

Table 1: Summary of Notations

$\mathcal{T}$	set of T epoch
$\mathcal{E}_t$	set of available clients in t
$x_{t,k}$	whether client k is selected or not in t
$c_{t,k}$	cost for renting client k in t
l_t	# iterations performed in t
η_t	maximal accuracy of local model in t
$\mathcal{F}_{t,k}(\boldsymbol{w})$	loss function of client k in t
$\mathcal{F}_t(\boldsymbol{w})$	loss function of all clients in t
$\boldsymbol{w}_t^i$	global model on i iteration in t, $\forall i \in [1, l_t]$
θ_0	global convergence accuracy
θ	the desirable upper-bound of global loss
$\tau_{t,k}^{loc}$	computation time for each update on k in t
$\tau_{t,k}^{cm}$	transmission latency of client k in t
$\eta_{t,k}^i$	local convergence accuracy of k on i in t

Decision Variable. Considering clients' uncertainty (*e.g.*, battery failure, device offline), we denote $\mathcal{E}_t, \forall t \in \mathcal{T}$ as the set of available clients in t. Each client is endowed with different computational and communication capabilities, leading to different training and transmitting times. We wish to select clients for model training in each epoch. To this end, we introduce a binary *decision variable* $x_{t,k} \in \{0, 1\}, \forall t \in \mathcal{T}, k \in \mathcal{E}_t$, which denotes whether client k is selected ($x_{t,k} = 1$) or not ($x_{t,k} = 0$) in t. The cost for renting client k in t is $c_{t,k}$.

Information of Clients. Assuming client k hosts a training dataset $\mathcal{D}_{t,k}$ with size $D_{t,k} \triangleq |\mathcal{D}_{t,k}|$. The training data can be generated via the interaction between client and mobile application and used to learn various ML tasks, such as predicting client activity or BS load. Recall that each client runs multiple local iterations using its data; we denote l_t as the number of iterations performed in t. We should emphasize that l_t is the same for all clients in t. Meanwhile, we introduce another *decision variable* $\eta_t \in [0, 1), \forall t \in \mathcal{T}$, which represents the maximal accuracy of local model in t. As discussed later, η_t can determine the number of training iterations performed in t.

Federated Learning Process. In each epoch, each client has a learning objective (*i.e.,* minimizing the loss), and updates its local model based on its loss. At the end of each epoch, each client submits the model update to the server for aggregation. The training process iterates until the global model converges. Now, we discuss the detailed process in the following four parts.

1) Loss. In a standard ML framework, for each data sample $\{a_{kl} \in R^d, b_{kl} \in R\}_{l=1}^{D_{t,k}}$ with a d-th dimensional input vector a_{kl}, the goal is to train model w that characterizes ground-truth output b_{kl} with loss function. Thus, the loss function of client k in t is formulated as

$$\mathcal{F}_{t,k}(\boldsymbol{w}) = \frac{1}{D_{t,k}} \sum_{l=1}^{D_{t,k}} f_k(\boldsymbol{w}; a_{kl}, b_{kl}),$$

where $f_k(\cdot)$ denotes a user-specified loss function on client k. Thus, the loss function of all clients in t is

$$\mathcal{F}_t(\boldsymbol{w}) = \sum_{k \in \mathcal{E}_t} \{\vartheta_k \cdot \mathcal{F}_{t,k}(\boldsymbol{w})\},$$

where $\vartheta_k = \frac{D_{t,k}}{\sum_k D_{t,k}} \geq 0$. We should emphasize that $\mathcal{F}_t(\boldsymbol{w})$ means that all devices have participated in the FL process.

2) Model Training. For iteration $i \in [1, l_t]$, client k trains its local model via the distributed approximate Newton (DANE) method [7] as follows

$$\boldsymbol{w}_{t,k}^i = \boldsymbol{w}_t^{i-1} + \boldsymbol{d}_{t,k}^{i-1} = \boldsymbol{w}_t^{i-1} + arg \min_{\boldsymbol{d}} \mathcal{G}_{t,k}^i(\boldsymbol{d}),$$

where $\boldsymbol{w}_t^{i-1}$ is the global model broadcast by the server, $\boldsymbol{w}_{t,k}^i$ is the updated model of client k, $\boldsymbol{d}^1$ represents a difference between local model trained on client k and current global model, and $\mathcal{G}_{t,k}^i(\cdot)$ is a carefully-designed function concerning $\boldsymbol{d}$. Similar to existing studies [25], we define

$$\begin{aligned}\mathcal{G}_{t,k}^i(\boldsymbol{d}) = & \mathcal{F}_{t,k}(\boldsymbol{w}_t^{i-1} + \boldsymbol{d}) + \frac{\sigma_1}{2}\|\boldsymbol{d}\|^2 \\ & - (\nabla \mathcal{F}_{t,k}(\boldsymbol{w}_t^{i-1}) - \sigma_2 \mathcal{J}_t(\boldsymbol{w}_t^{i-1}))^\top \boldsymbol{w},\end{aligned}$$

where σ_1 and σ_2 are positive constants, and $\mathcal{J}_t(\cdot)$ is the aggregated information broadcast by the server. Note that $\boldsymbol{w}_t^0 = \boldsymbol{w}_{t-1}^{l_t}$.

[1] In this paper, bold symbols represent column vectors.

To solve $\boldsymbol{d}^i_{t,k}$, we adopt the stochastic gradient decent (SGD) method as follows

$$d^{i,j}_{t,k} = d^{i,j-1}_{t,k} - \alpha\nabla\mathcal{G}^i_{t,k}(d^{i,j-1}_{t,k}),$$

where α is a step size, $d^{i,0}_{t,k}$ is set to zero by default, and $d^{i,j}_{t,k}$ represents the model trained on client k after j-th local update. Note that, in each iteration, the maximal value of gradient steps j is a pre-defined constant and can be conducted as the above [18, 19].

***3) Aggregation on Server*.** When iteration $i \in [1, l_t]$ ends, the server collects all updated models and gradients from the participants for aggregation:

$$\boldsymbol{w}^i_t = \boldsymbol{w}^{i-1}_t + \frac{1}{|\mathcal{E}_t|}\sum_k [x_{t,k}\cdot \boldsymbol{d}^i_{t,k}],$$

$$\mathcal{J}_t(\boldsymbol{w}^i_t) = \frac{1}{|\mathcal{E}_t|}\sum_k \mathcal{F}_{t,k}(\boldsymbol{w}^i_t).$$

Thus, the loss function of all participants in t is given by

$$\widetilde{\mathcal{F}}_t(\boldsymbol{w}^i_t) = \sum_{k\in\mathcal{E}_t}(x_{t,k}\cdot\mathcal{F}_{t,k}(\boldsymbol{w}^i_t)).$$

We should emphasize that not all devices are involved in training. Only some devices are participants. Meanwhile, the server cannot access the raw training data, thus preserving data privacy.

***4) Model Convergence*.** In each epoch, we should guarantee the convergence of local model and global model. Suppose $\mathcal{F}_{t,k}(\cdot)$ is L-Lipschitz continuous and γ-strongly convex in [18, 25], we can write the convergence as follows

$$\mathcal{G}^{l_t}_{t,k}(\boldsymbol{d}^{l_t}_{t,k}) - \mathcal{G}^{l_t,*}_{t,k} \le \eta^{l_t}_{t,k}[\mathcal{G}^{l_t}_{t,k}(\boldsymbol{0}) - \mathcal{G}^{l_t,*}_{t,k}],$$

$$\widetilde{\mathcal{F}}_t(\boldsymbol{w}^{l_t}_t) \le \widetilde{\mathcal{F}}^*_t + \theta_0[\widetilde{\mathcal{F}}_t(\boldsymbol{w}^0_t) - \widetilde{\mathcal{F}}^*_t] = \theta,$$

where $\eta^{l_t}_{t,k}$ denotes the local convergence accuracy on client k, and θ_0 is the global convergence accuracy. Here, $\mathcal{G}^{l_t,*}_{t,k}$ and $\widetilde{\mathcal{F}}^*_t$ represent the local optimum and global optimum on training data. θ denotes the desirable upper-bound of global loss. As defined that η_t is the maximal local convergence accuracy in t, we can write

$$\eta_t = \max_k \eta_{t,k} = \max_{k,i}\eta^i_{t,k}. \tag{1}$$

To achieve the desirable global loss θ_0, at least $l_t(\eta_t, \theta_0)$ iterations can be performed as in [13, 25], *i.e.,*

$$l_t(\eta_t,\theta_0) = \frac{O(\log(1/\theta_0))}{1-\eta_t}.$$

Given a value of θ_0, we only adjust η_t to determine the amount of local iterations.

3.2 Latency

For each client, the latency consists of the following two components.

***Local Computation*.** For client k, let π_k represent the CPU frequency on client k and e_k be the amount of CPU cycles for client k to train a sample of data. Therefore, the computational latency for each local update on client k in t is

$$\tau^{loc}_{t,k} = \frac{e_k\cdot|\mathcal{D}_{t,k}|}{\pi_k}.$$

Note that, due to the sufficient computational capability and relatively low complexity of the aggregated model, computation latency of aggregated global model can be ignored.

***Wireless Communication*.** After local training, each participating client uploads its local model to the server using frequency domain multiple access (FDMA) protocol [24, 25]. The achievable transmission rate of participating client k in t is

$$r_{t,k} = b_{t,k}\cdot\log_2(1+\frac{h_k p_k}{N_0 b_{t,k}}),$$

where $b_{t,k}$ denotes the bandwidth allocated to client k, p_k represents the transmit power of client k that stays unchanged in different epochs, h_k is the corresponding channel gain between client k and the server, and N_0 denotes the noise power density. Since edge network has a limited bandwidth, we have $\sum_k b_{t,k} = B$, where B is the bandwidth capacity.

During the uploading process, since the dimension of the local model is fixed for all clients [7, 25], we assume that the data size that each client requires to upload is a constant, denoted by s. Thus, the transmission latency of client k in t is

$$\tau^{cm}_{t,k} = \frac{s}{r_{t,k}}.$$

Due to the transmit power of the server and downlink bandwidth of model broadcast, we also ignore the time of receiving aggregated model.

Therefore, for client k, the latency denoted by $d_k(t)$ can be given as

$$d_k(t) = l_t(\eta_t,\theta_0)\cdot(\tau^{loc}_{t,k}+\tau^{cm}_{t,k}).$$

3.3 Problem Formulation

In each epoch, the server only does model aggregation after all the selected clients execute the local iteration and upload the local model. Therefore, the latency of epoch t depends on the slowest client, *i.e.,*

$$d(\mathcal{E}_t) = \max_{k\in\mathcal{E}_t}\{d_k(t)\}. \tag{2}$$

Our goal is to minimize the overall time of FL process in an online manner while ensuring model convergence and respecting budget constraints. Firstly, the offline optimization problem that all inputs are known can be mathematically formulated:

$$\boldsymbol{P}_1: \quad \text{minimize} \sum_t d(\mathcal{E}_t) \tag{3}$$

subject to:

$$\sum_t\sum_k c_{t,k}\cdot x_{t,k} \le C, \forall t, \tag{3a}$$

$$\sum_k x_{t,k} \ge n, \forall t, \tag{3b}$$

$$x_{t,k}\cdot\max_{k,i}\{\eta^i_{t,k}\} \le \eta_t, \forall t,k, \tag{3c}$$

$$\sum_t[\mathcal{F}_t(\boldsymbol{w}^{l_t}_t) - \theta] \le 0, \tag{3d}$$

$$x_{t,k}\in\{0,1\}, \forall t,k, \tag{3e}$$

$$\eta_t\in[0,1), \forall t, \tag{3f}$$

where $\forall t,k$ represent $\forall t\in\mathcal{T}, k\in\mathcal{E}_t$. Constraint (3a) ensures the long-term cost of renting clients cannot exceed budget C in the FL procedure. To ensure the quality of model training, constraint (3b) guarantees at least n participants should be involved in each epoch. Constraint (3c) and (3d) indicate the convergence requirements of the local model and global model, respectively. Although the model is trained based on the loss $\widetilde{\mathcal{F}}_t$ of selected clients, we should guarantee that the trained model, when applied to all the data in the FL system, the loss $\mathcal{F}_t$ is bounded.

Challenges. Designing an online algorithm to achieve our desired goals is highly non-trivial, in view of the following challenges. *First*, the online solution requires guaranteeing the upper-bound regret. This requirement sharply increases the difficulty of algorithm design. *Second*, some dynamic inputs, such as $\boldsymbol{w}_t^{l_t-1}$, $\boldsymbol{d}_{t,k}^{l_t}$, $\eta_{t,k}^i$ are only observed after control decisions are made in t. Any online algorithm has the potential to violate constraints (3c) and (3d). *Third*, due to the long-term constraint (3a), the time horizon relies on a given budget C. It is difficult to dynamically manage the limited budget while minimizing the overall latency. *Last but most importantly*, our minimization problem is a mixed-integer program with non-linear terms, including $\frac{O(\log(1/\theta_0))}{1-\eta_t}$, $\max_{k\in\mathcal{E}_t} d_k(t)$ and $\max_{k,i}\{\eta_{t,k}^i\}$. Even removing these non-linear terms, the corresponding problem is still NP-hardness in an offline scenario. How can we solve it in an online manner efficiently and effectively?

4 FEDERATED EDGE LEARNING ALGORITHM

4.1 Algorithmic Overview

To tackle the above challenges, we design a federated edge learning framework, FedL. As illustrated in Fig. 1, the basic idea of FedL consists of two stages.

- *Reformulation stage.* To efficiently solve $\boldsymbol{P}_1$, we normalize $O(\log(1/\theta_0))$, and utilize equality (1), (2) and (4) to convert non-linear terms into linear terms. Then, we re-aggregate control decisions Φ for the convenience of solving. After that, we estimate the range of stopping epoch for a given budget to overcome the third challenge. Finally, we reformulate the original problem $\boldsymbol{P}_1$ to a convex program $\boldsymbol{P}_2$ with variable Φ.
- *Decision stage.* To solve $\boldsymbol{P}_2$, we first derive the relaxation of $\boldsymbol{P}_2$, denoted by $\boldsymbol{P}_3$. By solving each one-shot problem $\boldsymbol{P}_{3,t}$, the selection and iteration decisions can be obtained. Specifically, based on current inputs and current decisions, FedL predicts fractional decisions for the next epoch through a novel online learning method, tackling the first and last challenges. Then, the selection decisions can be translated into integers by the rounding algorithm without violating any constraints, overcoming the second challenge.

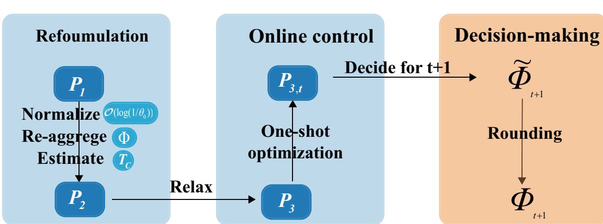

Figure 1: An illustration of FedL framework.

4.2 Problem Reformulation

In this work, due to a fixed accuracy θ_0, we normalize $O(\log(1/\theta_0))$ such that $l_t(\eta_t) = \frac{1}{1-\eta_t}$. Then, we introduce the fact as follows:

$$\max_{k\in\mathcal{E}_t}\{d_k(t)\} \leq \sum_{k\in\mathcal{E}_t} d_k(t). \tag{4}$$

After that, we use the aggregation of control decisions denoted by $\Phi_t = [x_{t,1}, \ldots, x_{t,\mathcal{E}_t}, \rho_t]^\top$ where $\rho_t = \frac{1}{1-\eta_t}$, and present several new notations as follows:

$$f_t(\Phi_t) = \sum_k \rho_t \cdot x_{t,k} \cdot (\tau_{t,k}^{loc} + \tau_{t,k}^{cm}),$$

$$p(\Phi_t) = \sum_t \sum_k c_{t,k} \cdot x_{t,k} - C,$$

$$q(\Phi_t) = n - \sum_k x_{t,k},$$

$$h_t^0 = \mathcal{F}_t(\boldsymbol{w}_t^{l_t-1} + \frac{1}{|\mathcal{E}_t|}\sum_k x_{t,k} \cdot \boldsymbol{d}_{t,k}^{l_t}) - \theta,$$

$$h_t^k = \eta_t \cdot x_{t,k} \cdot \rho_t - \rho_t + 1,$$

$$\boldsymbol{h}_t(\Phi_t) = [h_t^0, h_t^1, \ldots, h_t^M]^\top,$$

where $f_t(\cdot)$ corresponds to the objective function of (3), $p(\cdot)$ and $q(\cdot)$ correspond to constraints (3a) and (3b), and $\boldsymbol{h}_t(\cdot)$ corresponds to constraints (3c) and (3d). Given a long-term budget C, the stopping epoch denoted by T_C has a range, *i.e.,*

$$\frac{C}{n \cdot \max\{c_{t,k}\}} \leq T_C \leq \frac{C}{n \cdot \min\{c_{t,k}\}}.$$

Therefore, we reformulate the original problem as $\boldsymbol{P}_2$:

$$\boldsymbol{P}_2: \qquad \text{minimize} \sum_{t\leq T_C} f_t(\Phi_t) \tag{5}$$

subject to:

$$p(\Phi_t) \leq 0, \forall t \tag{5a}$$

$$q(\Phi_t) \leq 0, \forall t, \tag{5b}$$

$$\sum_{t\leq T_C} \boldsymbol{h}_t(\Phi_t) \leq 0, \tag{5c}$$

$$\Phi_t \in \mathcal{X} = \{\Phi_t | x_{t,k} \in \{0,1\}, \rho_t \geq 1\}, \forall t, \tag{5d}$$

where $\forall t$ represents $\forall t \leq T_C$. Our aim is to make decisions online. However, some system inputs, such as $\boldsymbol{w}_t^{l_t-1}$, $\boldsymbol{d}_{t,k}^{l_t}$, $\eta_{t,k}^i$ can be known after decisions Φ_t are made in t, which is a typical scenario of online learning. Due to the property of such inputs, online algorithms may excessively violate constraints (3c) and (3d). To this end, we develop an efficient online learning algorithm.

Theorem 1. The solution obtained in $\boldsymbol{P}_2$ is a feasible solution to $\boldsymbol{P}_1$.

Proof: Constraint (5a) means inequality $p(\Phi_t) = \sum_t \sum_k c_{t,k} \cdot x_{t,k} - C \leq 0$ holds, equivalent to (3a). In the same way, we can deduce that constraint (5b) is equivalent to constraint (3b). Since $\mathcal{F}_t(\boldsymbol{w}_t^{l_t-1} + \frac{1}{|\mathcal{E}_t|}\sum_k x_{t,k} \cdot \boldsymbol{d}_{t,k}^{l_t})$ is equal to $\mathcal{F}_t(\boldsymbol{w}_t^{l_t})$, inequality $h_t^0 \leq 0$ is equivalent to inequality $\mathcal{F}_t(\boldsymbol{w}_t^{l_t}) - \theta \leq 0$, which corresponds to constraint (3d) in t. As defined in h_t^k, we can deduct $h_t^k = \eta_t \cdot x_{t,k} \cdot \rho_t - \rho_t + 1 = \frac{\eta_t \cdot x_{t,k}}{1-\eta_t} - \frac{1}{1-\eta_t} + \frac{1-\eta_t}{1-\eta_t} = \frac{\eta_t \cdot x_{t,k} - \eta_t}{1-\eta_t}$. Since $1-\eta_t$ is greater than zero, when $h_t^1, h_t^2, \ldots, h_t^M$ are less than or equal to zero, this means that inequality $\eta_t \cdot x_{t,k} - \eta_t \leq 0$ holds, corresponding to constraint (3c). Together with $h_t^0 \leq 0$, constraint (5c) is equivalent to constraints (3c) and (3d). In original problem $\boldsymbol{P}_1$, the FL system operates within the time horizon, limited by the budget. The time horizon is directly converted into the expired time T_C in $\boldsymbol{P}_2$. Thus, constraints (5a) and (5b) are equivalent to constraints (3a) and (3b), respectively. Constraint (5c) is equivalent to constraints (3c) and (3d). Therefore, any feasible solution to $\boldsymbol{P}_2$ is a feasible solution to $\boldsymbol{P}_1$. □

4.3 Online Learning Algorithm

Now, we design an efficient online learning algorithm by alternately optimizing primal variables and Lagrange multipliers using dedicated descent-ascent steps. First, we relax constraint (5d) and then solve a one-shot problem $\boldsymbol{P}_{3,t}$ as follows

$$\boldsymbol{P}_{3,t}: \qquad \text{minimize } f_t(\widetilde{\Phi}_t) \tag{6}$$

subject to:

$$p(\widetilde{\Phi}_t) \leq 0, \tag{6a}$$

$$q(\widetilde{\Phi}_t) \leq 0, \tag{6b}$$

$$\boldsymbol{h}_t(\widetilde{\Phi}_t) \leq 0, \tag{6c}$$

$$\widetilde{\Phi}_t \in \widetilde{\mathcal{X}} = \{\widetilde{\Phi}_t | \widetilde{x}_{t,k} \in [0,1], \rho_t \geq 1\}, \tag{6d}$$

Solving $\boldsymbol{P}_{3,t}$ is actually equivalent to solving the following problem:

$$\boldsymbol{P}_4: \qquad \min_{\widetilde{\Phi}_t} \max_{\boldsymbol{\mu}_t} \left(f_t(\widetilde{\Phi}_t) + \boldsymbol{\mu}_t^\top \boldsymbol{h}_t(\widetilde{\Phi}_t) \right) \tag{7}$$

subject to:

$$(6a), (6b), (6d),$$

$$\boldsymbol{\mu}_t \geq 0, \tag{7e}$$

where the objective function is the representation of Lagrangian function, and $\boldsymbol{\mu}_t$ is the corresponding Lagrange multiplier. Therefore, we solve the following problem online:

$$\mathcal{L}_t(\widetilde{\Phi}, \boldsymbol{\mu}) \triangleq f_t(\widetilde{\Phi}) + \boldsymbol{\mu}^\top \boldsymbol{h}_t(\widetilde{\Phi}).$$

Due to the convexity of $f_t(\cdot)$ and $\boldsymbol{h}_t(\cdot)$, straightforward guidance of online solving $\mathcal{L}_t(\cdot)$ is the gradient incurred in $t-1$. Thus, our online learning algorithm alternately minimizes $\mathcal{L}_t(\widetilde{\Phi}, \boldsymbol{\mu}_{t+1})$ regarding $\widetilde{\Phi}$ through the modified descent step and maximizes $\mathcal{L}_t(\widetilde{\Phi}_t, \boldsymbol{\mu})$ regarding $\boldsymbol{\mu}$ through the standard ascent step. To elaborate, in $t+1$, we derive $\widetilde{\Phi}_{t+1}$ by solving:

$$\min_{\widetilde{\Phi}_{t+1}} \nabla f_t(\widetilde{\Phi}_t)(\widetilde{\Phi}_{t+1} - \widetilde{\Phi}_t) + \boldsymbol{\mu}_{t+1}^\top \boldsymbol{h}_t(\widetilde{\Phi}_{t+1}) + \frac{\|\widetilde{\Phi}_{t+1} - \widetilde{\Phi}_t\|^2}{2\beta} \tag{8}$$

subject to:

$$(6a), (6b),$$

where β is a step size, and $\nabla f_t(\widetilde{\Phi}_t)$ denotes the gradient of $f_t(\cdot)$ at $\widetilde{\Phi}_t$. The first and second term in (8) together approximate $\mathcal{L}_t(\widetilde{\Phi}, \boldsymbol{\mu}_{t+1})$, and the last term represents the proximal term. Note that (8) is not a strict standard but a carefully-modified descent step that can penalize these violations of constraints (3c) and (3d). Lagrange multiplier is optimized as

$$\boldsymbol{\mu}_{t+1} = [\boldsymbol{\mu}_t + \delta \nabla_{\boldsymbol{\mu}} \mathcal{L}_t(\widetilde{\Phi}_t, \boldsymbol{\mu}_t)]^+ = [\boldsymbol{\mu}_t + \delta \boldsymbol{h}_t(\widetilde{\Phi}_t)]^+, \tag{9}$$

where δ also denotes a positive step size, and $\nabla_{\boldsymbol{\mu}} \mathcal{L}_t(\widetilde{\Phi}_t, \boldsymbol{\mu}_t) = \boldsymbol{h}_t(\widetilde{\Phi}_t)$ represents the gradient of $\mathcal{L}_t(\widetilde{\Phi}_t, \cdot)$ at $\boldsymbol{\mu} = \boldsymbol{\mu}_t$. Clearly, updating $\boldsymbol{\mu}_{t+1}$ and $\widetilde{\Phi}_{t+1}$ only requires observable knowledge in t, which is the core idea of our online learning algorithm.

Our federated edge learning algorithm is shown as in Alg. 1. Lines 8 – 9 show that $\boldsymbol{\mu}_{t+1}$ and $\widetilde{\Phi}_{t+1}$ can be updated. To translate these fractions into integers, we introduce the rounding algorithm, which is discussed in detail next.

Algorithm 1 Federated Edge Learning Algorithm (**FedL**)

Input: initial global model $\boldsymbol{w}_0$
Output: global model $\boldsymbol{w}_{T_C}$

1: **while** $C \geq 0$ **do**
2: Server broadcasts global model $\boldsymbol{w}_t$ to all clients;
3: Derive decisions Φ_t by calling Algorithm 2 on $\widetilde{\Phi}_t$;
4: Client k computes $\boldsymbol{w}_{t,k}^i$ and $\boldsymbol{d}_{t,k}^i$ and uploads to the server;
5: Server calculates the latest global model $\boldsymbol{w}_t$;
6: Update $C = C - \sum_k c_{t,k} \cdot x_{t,k}$;
7: Set $t = t + 1$;
8: Update $\boldsymbol{\mu}_{t+1}$ by (9);
9: Update $\widetilde{\Phi}_{t+1}$ by (8);
10: **end while**

4.4 Online Rounding Algorithm

Given constraint (3e), fractional decisions are to be rounded into integer ones. The values of ρ_t are reals, and it thus is not necessary to round. Accordingly, we first divide $\widetilde{\Phi}_t$ into two components, *i.e.*, $\widetilde{x}_{t,k}$ and ρ_t, and only round $\widetilde{x}_{t,k}$. Intuitively, a straightforward approach is the independent rounding method; the fundamental idea is that each selection decision is rounded up or down individually. That is to say, $\widetilde{x}_{t,k}$ is rounded up to the ceiling integer $\lceil \widetilde{x}_{t,k} \rceil$ with a probability of $\lceil \widetilde{x}_{t,k} \rceil - \widetilde{x}_{t,k}$, or rounded down to the floor integer $\lfloor \widetilde{x}_{t,k} \rfloor$ with a probability of $\widetilde{x}_{t,k} - \lfloor \widetilde{x}_{t,k} \rfloor$, while this independent rounding algorithm may generate an infeasible solution or lead to an excessive system latency. Therefore, we design a randomized dependent client selection (RDCS) algorithm motivated by the dependent rounding technique [8]. RDCS explores the interdependence of selection decisions. The basic idea of RDCS is, the rounded-up decision can compensate for a rounded-down decision. The decisions are not rounded up or down aggressively using this dependent rounding algorithm.

The detail of RDCS is presented in Alg. 2. In line 1, a pair of clients are randomly selected. In lines 3 – 4, two reals are obtained according to $\widetilde{x}_{t,i}$ and $\widetilde{x}_{t,j}$. Then, in lines 5 – 8, two reals are used as the probability to round a fraction into an integer. RDCS guarantees its performance: the sum of all-rounded values remains constant, all rounded values must be integers, and the expectation of each round value equals the fractional value.

Algorithm 2 The Randomized Dependent Client Selection Algorithm (**RDCS**)

Input: $\widetilde{\Phi}_t$

1: Randomly choose two clients i and j from $\{\widetilde{x}_{t,k}\}$;
2: **while** $\widetilde{x}_{t,i} \in (0,1) \wedge \widetilde{x}_{t,j} \in (0,1)$ **do**
3: Define $\zeta_1 \triangleq \min\{1 - \widetilde{x}_{t,i}, \widetilde{x}_{t,j}\}$;
4: Define $\zeta_2 \triangleq \min\{\widetilde{x}_{t,i}, 1 - \widetilde{x}_{t,j}\}$;
5: With probability $\frac{\zeta_2}{\zeta_1 + \zeta_2}$,
6: set $\widetilde{x}_{t,i} = \widetilde{x}_{t,i} + \zeta_1$, $\widetilde{x}_{t,j} = \widetilde{x}_{t,j} - \zeta_1$;
7: With probability $\frac{\zeta_1}{\zeta_1 + \zeta_2}$,
8: set $\widetilde{x}_{t,i} = \widetilde{x}_{t,i} - \zeta_2$, $\widetilde{x}_{t,j} = \widetilde{x}_{t,j} + \zeta_2$;
9: **end while**
10: Return $\Phi_t = [x_{t,1}, \ldots, x_{t,M}, \rho_t]^\top$

5 THEORETICAL ANALYSIS

Let us concentrate on the theoretical analysis now. *First*, we introduce the dynamic regret and fit, and present Lemma 1, Lemma 2 and Theorem 2. *Second*, Corollary 1 presents that the dynamic regret and dynamic fit have sub-linear upper-bounds with the time for the given budget. The sub-linear upper-bounds suggest that FedL makes asymptotically optimal control decisions and the increase of dynamic fit is much slower than the progress of time. *Third*, since a fraction is rounded to an integer in a randomized way, RDCS ensures that the expectation of each random integer is equal to its fractional value before rounding by Theorem 3. *Last*, Theorem 4 reveals that FedL runs in polynomial time. In the following, we discuss the theoretical analysis in detail.

Dynamic regret is defined as the difference between the accumulated objective function based on $\{\Phi_t\}$ and that based on $\{\Phi_t^*\}$. We consider the dynamic regret of $\boldsymbol{P}_1$, denoted by Reg_d, and dynamic regret of $\boldsymbol{P}_2$ in the integer domain and real domain, denoted by Reg_o and $\widetilde{Reg_o}$:

$$Reg_d \triangleq \mathbb{E}[\sum_{t\le T_C} d(\Phi_t)] - \sum_{t\le T_C} d(\Phi_t^*),$$
$$Reg_o \triangleq \mathbb{E}[\sum_{t\le T_C} f_t(\Phi_t)] - \sum_{t\le T_C} f_t(\Phi_t^*),$$
$$\widetilde{Reg_o} \triangleq \sum_{t\le T_C} f_t(\widetilde{\Phi}_t) - \sum_{t\le T_C} f_t(\widetilde{\Phi}_t^*),$$

where $d(\Phi_t) = \max_{k\in\Phi_t} d_k(t)$, $\Phi_t^* \in arg\min_{\Phi\in\mathcal{X}} f_t(\Phi)$ and $\widetilde{\Phi}_t^* \in arg\min_{\widetilde{\Phi}\in\widetilde{\mathcal{X}}} f_t(\widetilde{\Phi})$. Due to the algorithm RDCS, we thus introduce the expectation in Reg_o.

Dynamic fit is the accumulated violation of constraints according to online decisions $\{\Phi_t\}$. We adopt the form of $[\cdot]^+ = \max\{\cdot, 0\}$ to characterize such violation,

$$Fit_d \triangleq \|[\mathbb{E}(\sum_{t\le T_C} h_t(d(\Phi_t))]^+\|,$$
$$Fit_o \triangleq \|[\mathbb{E}(\sum_{t\le T_C} h_t(\Phi_t))]^+\|,$$
$$\widetilde{Fit_o} \triangleq \|[\sum_{t\le T_C} h_t(\widetilde{\Phi}_t)]^+\|.$$

Lemma 1. The dynamic regret and dynamic fit can be expressed as:

$$Reg_d \le Reg_o \le \widetilde{Reg_o}, \qquad Fit_d \le Fit_o \le \widetilde{Fit_o} + \Delta, \tag{10}$$

where Δ denotes a positive constant [9].

Proof. Seen Appendix A. □

To facilitate analysis, we begin with two assumptions:

Assumption 1. The objective function of problem $\boldsymbol{P}_{3,t}$ have bounded gradients w.r.t. $\widetilde{\mathcal{X}}$ and $\boldsymbol{h}_t(\widetilde{\Phi})$ is also bounded, *i.e.,* $\|\nabla f_t(\widetilde{\Phi})\| \le G_f, \forall\widetilde{\Phi} \in \widetilde{\mathcal{X}}$ and $\|\boldsymbol{h}_t(\widetilde{\Phi})\| \le G_h, \forall\widetilde{\Phi} \in \widetilde{\mathcal{X}}$. And the radius in a feasible domain is also bounded, *i.e.,* $\|m - n\| \le 2R, \forall m, n \in \widetilde{\mathcal{X}}$.

Assumption 2. There exists a positive constant $\xi > 0$ and one interior point $\mathring{\Phi}_t \in \widetilde{\mathcal{X}}$ so that $\boldsymbol{h}_t(\mathring{\Phi}_t) \le -\xi\mathbf{1}$ and $\xi > \widehat{V}(\boldsymbol{h})$, where $\widehat{V}(h) \triangleq \max_{t\le T_C}\max_{\widetilde{\Phi}\in\widetilde{\mathcal{X}}} \|[\boldsymbol{h}_{t+1}(\widetilde{\Phi}) - \boldsymbol{h}_t(\widetilde{\Phi})]^+\|$.

Assumption 1 shows that the gradients in the primal objective, dual objective and radius in the feasible domain are bounded, which are customarily adopted in online convex optimization. Assumption 2 makes sure that there exists a bounded optimal Lagrange multiplier when the feasible domain of $\boldsymbol{h}_t(\widetilde{\Phi}) \le 0$ is sufficiently large or the trajectory of $h_t(\widetilde{\Phi})$ is adequately smooth.

Lemma 2. Based on the above assumptions and Lagrange multiplier initialization $\boldsymbol{\mu}_1 = 0$, we obtain

$$\frac{\|\boldsymbol{\mu}_{t+1}\|^2 - \|\boldsymbol{\mu}_t\|^2}{2} \le \delta\boldsymbol{\mu}_t^{\top}\boldsymbol{h}_t(\widetilde{\Phi}_t) + \frac{\delta^2}{2}\|\boldsymbol{h}_t(\widetilde{\Phi}_t)\|^2, \tag{11}$$

$$\|\boldsymbol{\mu}_t\| \le \|\widehat{\boldsymbol{\mu}}\| \triangleq \delta G_h + \frac{2G_f R + R^2/(2\beta) + (\delta G_h^2)/2}{\xi - \widehat{V}(h)}. \tag{12}$$

Proof. Seen Appendix B. □

Theorem 2. Based on the above assumptions and Lagrange multiplier initialization $\boldsymbol{\mu}_1 = 0$, the upper-bounds of dynamic regret and dynamic fit can be given as

$$Reg_d \le \widetilde{Reg_o} \le \mathcal{R}_{T_C}, \quad Fit_d \le \widetilde{Fit_o} \le \frac{\boldsymbol{\mu}_{T_C+1}}{\delta} \le \frac{\|\widehat{\boldsymbol{\mu}}\|}{\delta}, \tag{13}$$

where

$$\mathcal{R}_{T_C} = \frac{\beta G_f^2 T_C}{2} + \|\widehat{\boldsymbol{\mu}}\| V(\{\boldsymbol{h}_t\}_{t=1}^{T_C}) + \frac{\delta G_h^2 T_C}{2} + \frac{R\cdot V(\{\widetilde{\Phi}_t^*\}_{t=1}^{T_C})}{\beta} + \frac{R^2}{2\beta}, \tag{13a}$$

$$V(\{\widetilde{\Phi}_t^*\}_{t=1}^{T_C}) \triangleq \sum_{t\le T_C}\|\widetilde{\Phi}_t^* - \widetilde{\Phi}_{t-1}^*\|, \tag{13b}$$

$$V(\{\boldsymbol{h}_t\}_{t=1}^{T_C}) \triangleq \sum_{t\le T_C}\max_{\widetilde{\Phi}\in\widetilde{\mathcal{X}}}\|[\boldsymbol{h}_{t+1}(\widetilde{\Phi}) - \boldsymbol{h}_t(\widetilde{\Phi})]^+\|. \tag{13c}$$

Proof. Seen our technical report [1]. □

Corollary 1. Based on the above assumptions and Lagrange multiplier initialization, we set the step sizes as $\beta = \delta = \max\{\sqrt{\frac{V(\{\widetilde{\Phi}_t^*\}_{t=1}^{T_C})}{T_C}}, \sqrt{\frac{V(\{\boldsymbol{h}_t\}_{t=1}^{T_C})}{T_C}}\}$ and can derive:

$$Reg_d = O(\max\{\sqrt{V(\{\widetilde{\Phi}_t^*\}_{t=1}^{T_C})T_C}, \sqrt{V(\{\boldsymbol{h}_t\}_{t=1}^{T_C})T_C}\}),$$
$$Fit_d \le \frac{\|\widehat{\boldsymbol{\mu}}\|}{\delta} + \Delta = O(\max\{\frac{T_C}{V(\{\widetilde{\Phi}_t^*\}_{t=1}^{T_C}}, \frac{T_C}{V(\{\boldsymbol{h}_t\}_{t=1}^{T_C})}\}) + \Delta.$$

According to such Corollary and $\frac{C}{n\cdot\max\{c_{t,k}\}} \le T_C \le \frac{C}{n\cdot\min\{c_{t,k}\}}$, when we set $\beta = \delta = O(T_C^{-\frac{1}{3}})$, the upper-bounds of dynamic regret and dynamic fit can be derived as

$$Reg_d = O(\max\{V(\{\widetilde{\Phi}_t^*\}_{t=1}^{T_C})T_C^{\frac{1}{3}}, V(\{\boldsymbol{h}_t\}_{t=1}^{T_C})T_C^{\frac{1}{3}}, T_C^{\frac{2}{3}}\}),$$

$$Fit_d = O(T_C^{\frac{2}{3}}) + \Delta.$$

Theorem 3. RDCS guarantees $\mathbb{E}[x_{t,i}] = \widetilde{x}_{t,i}, \forall t, i$.

Proof. Seen our technical report [1]. □

Theorem 4. FedL is a polynomial running time algorithm with complexity $O(T_C K^2)$, where $K = \max_t |\mathcal{E}_t|$.

Proof. Seen our technical report [1]. □

6 PERFORMANCE EVALUATION

In this section, we present results from empirical studies of FedL. The evaluation is according to real learning on real-world datasets: Fashion-MNIST (FMNIST) and CIFAR-10. We first introduce the basic setting of the experimental environment, present the benchmark algorithms, and finally show the numerical results.

6.1 Basic Setting

Implementation and Clients. Our testbed experiments are conducted on an FL simulator[2]. Unless otherwise specified, an FL system is composed of $M = 100$ clients uniformly distributed in an area of radius 500m and a server that resides in the centre of the area. The path loss is modeled as $128.1 + 37.6 \log_{10} d$ (d is distance) as well as standard deviation on shadow fading is 8 dB [24]. Besides, let the power spectrum density of Gaussian noise be $N_0 = -174$ dBm/Hz. The bandwidth is $B = 20$ MHz. For client k, let e_k be uniformly distributed in $[10, 30]$ cycles/bit [25]. We set the maximal transmit power $p_1^{max} = p_2^{max} = \cdots = p_M^{max} = 10$dB and maximal CPU cycle $f_1^{max} = f_2^{max} = \cdots = f_M^{max} = 2$ GHz. Renting the cost of a client are uniformly distributed in $[0.1, 12]$ based on the dynamic price of Amazon. In addition, the availability of all devices obeys the same Bernoulli distribution.

Data. We choose two public datasets: FMNIST (a dataset of Zalando's fashion article images) and CIFAR-10 (a dataset of 32×32 color images). FMNIST and CIFAR-10 consist of 70,000 and 60,000 images, respectively. The images from each dataset can be partitioned into ten classes. We also consider data distribution with a non-iid setting. To simulate non-iid distribution, we choose a number of data from a principal dataset and randomly select the remaining data from another dataset. Note that all data are then transformed into online data followed by Poisson distribution.

Models. We employ two Convolutional Neural Network (CNN) models with different structures to learn the classification results. For FMNIST, the CNN model has two 5×5 convolutional layers (32, 64 channels), followed by 2×2 max pooling, a fully connected layer with 1024 units, and finally a softmax output layer (10 units). For CIFAR-10, a harder task, another CNN model also owns two 5×5 convolutional layers (64, 64 channels), followed by 3×3 max pooling, two fully connected layers with 384 and 192 units, and finally a softmax output layer (10 units).

Baselines. We choose and implement three state-of-the-art schemes for comparing online solutions.

- FedAvg [19] is the most original but has been widely used in client selection. The server randomly selects participants to train the model.
- FedCS [21] selects as many clients as possible to train and terminates the model training upon a fixed deadline in each epoch.
- Pow-d [5] emphasizes selection fairness and then selects clients with larger local losses.

Note that all algorithms mentioned above cannot derive the real available clients and local convergence in advance, as the training process is online. Solving problem (8) is the crucial step of FedL and can be conducted by applying the Interior-Point Filter Line-Search algorithm [26]. We use three metrics to evaluate the training performance of FedL: accuracy, training time, and loss.

6.2 Result

Accuracy vs Training time. In Figs. 2-3, it can be observed that each algorithm needs more training time to reach higher accuracy. Pow-d tends to select clients with lower accuracy, which leads to larger local losses. It thus needs more training time to achieve the same accuracy. The accuracy of FedAvg is lower than that of FedCS since it can access all available clients. In contrast, FedCS has a significant advantage in the beginning. Still, the performance gap between FedL and FedCS is acceptable. As time goes, FedL can catch up with FedCS and gradually surpass FedCS. Finally, the training time of FedL is shorter among the selection strategies. Because FedL can explore the best clients who help FL training converge. On the other hand, FedAvg and FedCS use the random selection strategy for budget consideration, and Pow-d adopts the CS strategy of local loss maximization. Thus they lose the chance to search for the optimal subset of clients. For instance, by Fig. 2, for the IID-FMNIST training, the accuracy of FedL, FedCS, FedAvg and Pow-d is 93.73%, 91.12%, 88.19%, 76.72% after 2000 seconds of training. To achieve an accuracy of 75%, the completion time of those schemes is 460s, 750s, 800s, and 1450s, respectively.

When training the Non-IID data, the accuracy would fluctuate to varying degrees. For example, in Fig. 2, after 2000 seconds of training, the accuracy of Pow-d, FedAvg, and FedCS varies in $[70\%, 73\%]$, $[80\%, 83\%]$, and $[81\%, 85\%]$, respectively. In contrast, the accuracy of FedL is relatively over 86%. The training times that stabilized above 70% accuracy are 390s, 650s, 650s, and 1290s, respectively. Figs. 2-3 illustrate that FedL can reduce training time when reaching target accuracy.

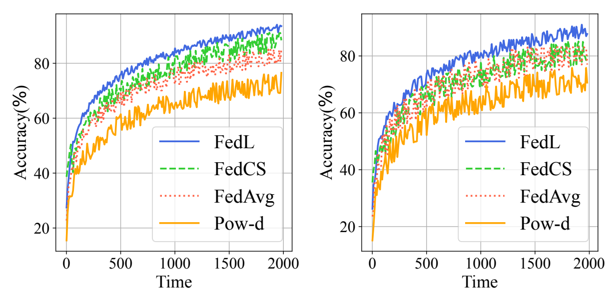

Figure 2: Accuracy vs. time for Fashion-MNIST. Left: IID; Right: Non-IID.

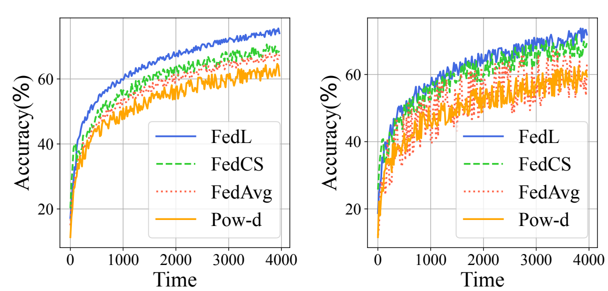

Figure 3: Accuracy vs. time for CIFAR-10. Left: IID; Right: Non-IID.

Accuracy vs Federated round. Figs. 4-5 illustrate how each scheme performs in federated rounds. As expected, Pow-d performs the worst. FedCS performs better than FedAvg and Pow-d since

[2] https://github.com/CharlieDinh/FEDL.

FedCS selects as many clients as possible. At first, the training performance of FedL is slightly worse than FedCS. However, FedCS overemphasizes the current optimization rather than learning from the historical information. The performance gap of FedCS and FedL narrows rapidly until FedL surpasses FedCS. For example, for IID CIFAR-10 training, the accuracy of FedL, FedCS, FedAvg and Pow-d is 74.06%, 70.41%, 68.44%, 62.70% after 200 federated rounds. To achieve an accuracy of 60%, the federated round of those schemes is 52, 72, 86, and 122, respectively. When training Non-IID data, all schemes' accuracy fluctuates to some degrees. But FedL is the most stable.

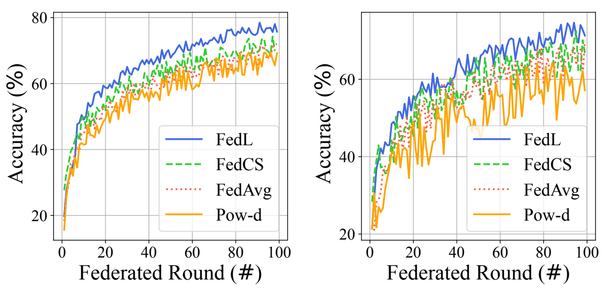

Figure 4: Accuracy vs. Federated round for FMNIST. Left: IID; Right: Non-IID.

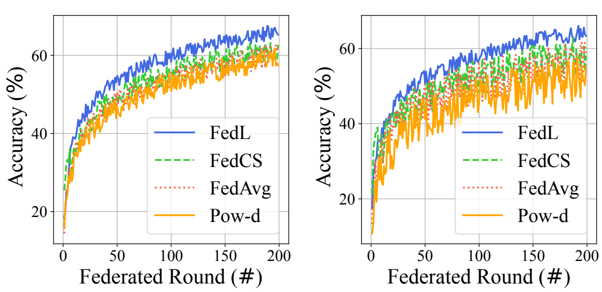

Figure 5: Accuracy vs. Federated round for CIFAR-10. Left: IID; Right: Non-IID.

Impact of Budget. Figs. 6-7 further investigate the constraints' effect when training tasks. When the budget is small, there is a considerable gap between FedL and the benchmarks since they overemphasize the optimization of current task and do not any preparation for future tasks. As the budget increases, the benchmarks can perform more rounds of model training so that their loss gradually decreases. In contrast, FedL has no noticeable decreasing trend and consistently preserves lower losses even with a small budget. That is to say, FedL can finish FL tasks with less budget, which can reap benefits in practical application.

To sum, FedL consistently achieves better performance (*e.g.*, training time, accuracy, loss) than the benchmarks under different datasets and data distribution.

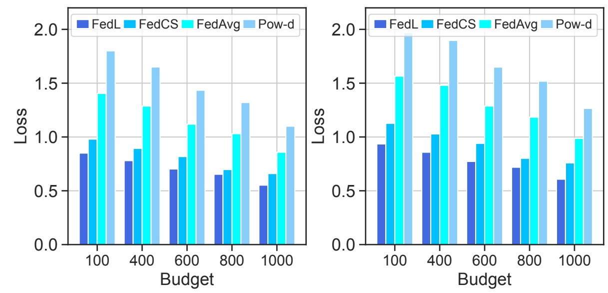

Figure 6: Budget impact for Fashion-MNIST. Left: IID; Right: Non-IID.

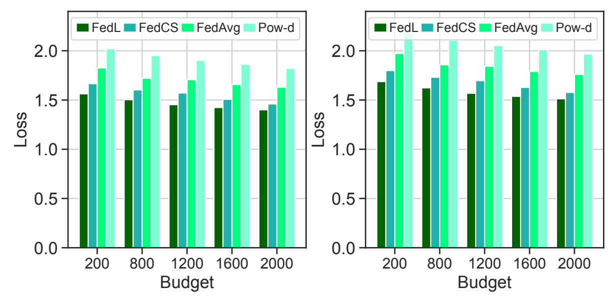

Figure 7: Budget impact for CIFAR-10. Left: IID; Right: Non-IID.

7 CONCLUSION

In FL, extensive model training and transference are time-consuming. To expedite FL, we designed the FedL framework to select appropriate clients and control the number of training iterations in real-time, reaching desirable model convergence and satisfying the long-term budget. FedL consists of two online algorithms: the online learning algorithm learns CS strategies and iteration control, and the online rounding algorithm translates the fractional decisions derived by the online learning algorithm into integer ones. Dynamic regret and dynamic fit are proven to have sub-linear upper-bounds with time for a given budget. Extensive evaluation results verified the advantage of FedL over multiple state-of-the-art algorithms. We will consider selection fairness to further expand the CS capabilities in the future.

8 ACKNOWLEDGEMENT

The corresponding authors are Ruiting Zhou and Zongpeng Li. This work is supported in part by the NSFC Grants (62072344 and U20A20177), and Huawei Project (FA2019071041).

REFERENCES

[1] 2022. *Technical report.* https://www.jianguoyun.com/p/DWevre0Q-8GiChjXgrkEIAA.

[2] Mehdi Salehi Heydar Abad, Emre Ozfatura, Deniz Gunduz, and Ozgur Ercetin. [n.d.]. Hierarchical federated learning across heterogeneous cellular networks. In *ICASSP 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 8866–8870.

[3] Niv Buchbinder, Shahar Chen, and Joseph Naor. 2014. Competitive analysis via regularization. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*. SIAM, 436–444.

[4] Mingzhe Chen, Zhaohui Yang, Walid Saad, Changchuan Yin, H Vincent Poor, and Shuguang Cui. 2020. A joint learning and communications framework for federated learning over wireless networks. *IEEE Transactions on Wireless Communications* 20, 1 (2020), 269–283.

[5] Yae Jee Cho, Jianyu Wang, and Gauri Joshi. 2020. Client selection in federated learning: Convergence analysis and power-of-choice selection strategies. *arXiv*

preprint arXiv:2010.01243 (2020).
[6] Yongheng Deng, Feng Lyu, Ju Ren, Huaqing Wu, Yuezhi Zhou, Yaoxue Zhang, and Xuemin Sherman Shen. 2022. AUCTION: Automated and Quality-Aware Client Selection Framework for Efficient Federated Learning. *IEEE Transactions on Parallel and Distributed Systems* (2022).
[7] Canh T. Dinh, Nguyen H. Tran, Minh N. H. Nguyen, Choong Seon Hong, Wei Bao, Albert Y. Zomaya, and Vincent Gramoli. 2021. Federated Learning Over Wireless Networks: Convergence Analysis and Resource Allocation. *IEEE/ACM Transactions on Networking* 29, 1 (2021), 398–409. https://doi.org/10.1109/TNET.2020.3035770
[8] R Gandhi, S Khuller, S Parthasarathy, and A Srinivasan. 2006. Dependent rounding and its applications to approximation algorithms. *J. ACM* 53, 3 (2006), 324–360.
[9] Xiang Gao, Meera Sitharam, and Adrian E Roitberg. 2017. Bounds on the Jensen gap, and implications for mean-concentrated distributions. *arXiv preprint arXiv:1712.05267* (2017).
[10] White House. 2012. Consumer data privacy in a networked world: A framework for protecting privacy and promoting innovation in the global digital economy. *White House, Washington, DC* (2012), 1–62.
[11] Tiansheng Huang, Weiwei Lin, Wentai Wu, Ligang He, Keqin Li, and Albert Y Zomaya. 2020. An efficiency-boosting client selection scheme for federated learning with fairness guarantee. *IEEE Transactions on Parallel and Distributed Systems* 32, 7 (2020), 1552–1564.
[12] Yutao Jiao, Ping Wang, Dusit Niyato, Bin Lin, and Dong In Kim. 2020. Toward an automated auction framework for wireless federated learning services market. *IEEE Transactions on Mobile Computing* (2020).
[13] Yibo Jin, Lei Jiao, Zhuzhong Qian, Sheng Zhang, Sanglu Lu, and Xiaoliang Wang. [n.d.]. Resource-Efficient and Convergence-Preserving Online Participant Selection in Federated Learning. In *2020 IEEE 40th ICDCS*. 606–616. https://doi.org/10.1109/ICDCS47774.2020.00049
[14] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. 2020. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*. PMLR, 5132–5143.
[15] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems* 2 (2020), 429–450.
[16] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2020. On the convergence of fedavg on non-iid data. *ICLR* (2020).
[17] Wei Liu, Li Chen, Yunfei Chen, and Wenyi Zhang. 2020. Accelerating federated learning via momentum gradient descent. *IEEE Transactions on Parallel and Distributed Systems* 31, 8 (2020), 1754–1766.
[18] Chenxin Ma, Jakub Konečný, Martin Jaggi, Virginia Smith, Michael I Jordan, Peter Richtárik, and Martin Takáč. 2017. Distributed optimization with arbitrary local solvers. *Optimization Methods and Software* 32, 4 (2017), 813–848.
[19] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.
[20] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. 2019. Agnostic federated learning. In *International Conference on Machine Learning*. PMLR, 4615–4625.
[21] Takayuki Nishio and Ryo Yonetani. 2019. Client selection for federated learning with heterogeneous resources in mobile edge. In *IEEE International Conference on Communications (ICC)*. 1–7.
[22] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and H Brendan McMahan. 2020. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295* (2020).
[23] Ju Ren, Deyu Zhang, Shiwen He, Yaoxue Zhang, and Tao Li. 2019. A survey on end-edge-cloud orchestrated network computing paradigms: Transparent computing, mobile edge computing, fog computing, and cloudlet. *ACM Computing Surveys (CSUR)* 52, 6 (2019), 1–36.
[24] Wenqi Shi, Sheng Zhou, Zhisheng Niu, Miao Jiang, and Lu Geng. 2021. Joint Device Scheduling and Resource Allocation for Latency Constrained Wireless Federated Learning. *IEEE Transactions on Wireless Communications* 20, 1 (2021), 453–467. https://doi.org/10.1109/TWC.2020.3025446
[25] Nguyen H. Tran, Wei Bao, Albert Zomaya, Minh N. H. Nguyen, and Choong Seon Hong. 2019. Federated Learning over Wireless Networks: Optimization Model Design and Analysis. In *IEEE INFOCOM*. 1387–1395. https://doi.org/10.1109/INFOCOM.2019.8737464
[26] Andreas Wächter and Lorenz T Biegler. 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming* 106, 1 (2006), 25–57.
[27] Hao Wang, Zakhary Kaplan, Di Niu, and Baochun Li. 2020. Optimizing federated learning on non-iid data with reinforcement learning. In *IEEE INFOCOM*. 1698–1707.
[28] Luping WANG, Wei WANG, and Bo LI. [n.d.]. CMFL: Mitigating Communication Overhead for Federated Learning. In *2019 IEEE 39th ICDCS*. 954–964. https://doi.org/10.1109/ICDCS.2019.00099
[29] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. 2019. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *IEEE INFOCOM*. 2512–2520.
[30] Wenchao Xia, Tony QS Quek, Kun Guo, Wanli Wen, Howard H Yang, and Hongbo Zhu. 2020. Multi-armed bandit-based client scheduling for federated learning. *IEEE Transactions on Wireless Communications* 19, 11 (2020), 7108–7123.
[31] Yufeng Zhan, Peng Li, Wu Leijie, and Song Guo. 2021. L4L: Experience-driven computational resource control in federated learning. *IEEE Trans. Comput.* (2021).
[32] Jie Zhang, Song Guo, Zhihao Qu, Deze Zeng, Yufeng Zhan, Qifeng Liu, and Rajendra A Akerkar. 2021. Adaptive federated learning on Non-IID data with resource constraint. *IEEE Trans. Comput.* (2021).
[33] Ruiting Zhou, Jinlong Pang, Zhibo Wang, John CS Lui, and Zongpeng Li. [n.d.]. A Truthful Procurement Auction for Incentivizing Heterogeneous Clients in Federated Learning. In *2021 IEEE 41st ICDCS*. IEEE, 183–193.

Appendix A

Proof of Lemma 1.

We start with proving dynamic regret as follows

$$
\begin{aligned}
&\mathbb{E}[\sum_{t\le T_C} d(\Phi_t)] - \sum_{t\le T_C} d(\Phi_t^*) \overset{(a)}{\le} \mathbb{E}[\sum_{t\le T_C} f_t(\Phi_t)] - \sum_{t\le T_C} f_t(\Phi_t^*)\\
&\overset{(b)}{=} \sum_{t\le T_C} f_t(\mathbb{E}[\Phi_t]) - \sum_{t\le T_C} f_t(\Phi_t^*)\\
&= \sum_{t\le T_C} f_t(\mathbb{E}[\Phi_t]) - \sum_{t\le T_C} f_t(\widetilde{\Phi}_t) + \sum_{t\le T_C} f_t(\widetilde{\Phi}_t) - \sum_{t\le T_C} f_t(\Phi_t^*)\\
&\overset{(c)}{\le} \sum_{t\le T_C} f_t(\widetilde{\Phi}_t) - \sum_{t\le T_C} f_t(\Phi_t^*) = \widetilde{Reg_o},
\end{aligned}
\tag{14}
$$

where inequality (14a) can be obtained due to inequality (4). Since $f_t(\cdot)$ is linear w.r.t. $x_{t,k}$, inequality (14b) holds. For the minimization problem, since the optimum in the integer domain is bigger than that in the real domain and $\mathbb{E}_x[\Phi_t] = \widetilde{\Phi}_t$, we can obtain inequality (14c).

Similar to the proof of dynamic regret, the dynamic fit can be proved as

$$
\begin{aligned}
&\|[\mathbb{E}(\sum_{t\le T_C} h_t(d(\Phi_t))]^+\| \overset{(a)}{\le} \|[\mathbb{E}(\sum_{t\le T_C} \boldsymbol{h}_t(\Phi_t))]^+\|\\
&\overset{(b)}{\le} \|\mathbb{E}[\sum_{t\le T_C} \boldsymbol{h}_t(\Phi_t)]\| \overset{(c)}{\le} \|\sum_{t\le T_C} \boldsymbol{h}_t(\mathbb{E}[\Phi_t]) + \Delta\|\\
&\overset{(d)}{=} \|\sum_{t\le T_C} \boldsymbol{h}_t(\widetilde{\Phi}_t) + \Delta\| \overset{(e)}{\le} \|\sum_{t\le T_C} \boldsymbol{h}_t(\widetilde{\Phi}_t)\| + \|\Delta\|\\
&= \widetilde{Fit_o} + \Delta,
\end{aligned}
\tag{15}
$$

where inequality (15a) can be obtained due to inequality (4). Since the absolute value can be reduced after using the function $[]^+$ to all dimensions. For example, a non-positive value can be translated into zero when employing $[]^+$, inequality (15b) thus holds. Due to Jensen Gap [9], we can conduct inequality (15c). Since our rounding algorithm ensures the expectation cannot change, inequality (15d) holds. Inequality (15e) is from the triangle inequality. □

Appendix B

Proof of Lemma 2.

As in (9) updated, we obtain

$$
\begin{aligned}
\|\boldsymbol{\mu}_{t+1}\|^2 &= \|[\boldsymbol{\mu}_t + \delta \boldsymbol{h}_t(\widetilde{\Phi}_t)]^+\|^2 \overset{(a)}{\le} \|\boldsymbol{\mu}_t + \delta \boldsymbol{h}_t(\widetilde{\Phi}_t)\|^2\\
&= \|\boldsymbol{\mu}_t\|^2 + 2\delta \boldsymbol{\mu}_t^\top \boldsymbol{h}_t(\widetilde{\Phi}_t) + \delta^2 \|\boldsymbol{h}_t(\widetilde{\Phi}_t)\|^2,
\end{aligned}
\tag{16}
$$

Given the limited space, the remaining proof is given in our technical report [1].