Joint Client Selection and Resource Allocation for Federated Learning in Mobile Edge Networks

Long Luo, *Member, IEEE,* Qingqing Cai, Zonghang Li, and Hongfang Yu, *Member, IEEE*
University of Electronic Science and Technology of China, P.R. China

2022 IEEE Wireless Communications and Networking Conference (WCNC) | 978-1-6654-4266-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/WCNC51071.2022.9771771

***Abstract*—Federated Learning (FL) has received widespread attention in 5G mobile edge networks (MENs) due to its ability to facilitate collaborative learning of machine learning models without revealing user privacy data. However, FL training is both time and energy consuming. Constrained by the instability and limited resources of clients in MENs, it is challenging to optimize both learning time and energy consumption for FL. This paper studies the problem of client selection and resource allocation to minimize the energy consumption and learning time of multiple FL jobs competing for resources. Because minimizing learning time and minimizing energy consumption are conflicting objectives, we design a decoupling algorithm to optimize them separately and efficiently. Simulations based on popular models and learning datasets show the effectiveness of our approach, reducing up to 75.7% energy consumption and 38.5% learning time compared to prior work.**

I. Introduction

Federated Learning (FL) emerges as an attractive learning paradigm to enable edge intelligence in 5G and beyond while protecting data privacy [1, 2]. We consider the state-of-the-art FL system, Hierarchical FL, in mobile edge networks (MENs) and call this system HFLMEN in this work. As illustrated in Fig. 1, HFLMEN distributes the computing tasks of machine learning (ML) jobs across many mobile user equipments (UEs) and uses a cloud server as parameter server (PS) to orchestrate the iterative learning process with the help of base stations (BSs). In each global training round, HFLMEN lets clients, the participating UEs, download global model parameters or gradients from the cloud server via BSs, train model using local datasets, and upload their model updates to BSs after a given number of local training rounds. BSs aggregate model updates of associated clients and send the results back to cloud server for global model synchronization.

However, HFLMEN is both time and energy consuming due to the following reasons. First, to ensure model convergence, FL typically requires a large number of clients to perform an iterative and long-term on-device model training, which incurs significant energy consumption on low-power devices. Second, given the very limited computing capacities and communication resources of UEs, how clients use computing and communication resources to perform on-device training also impacts learning time of FL jobs and energy consumption of devices a lot.

Despite many efforts on resource management for FL, existing research is insufficient in optimizing the learning time and energy consumption, especially when multiple FL jobs coexist in the HFLMEN system. With the rapid development of FL, an increasing number of applications (*i.e.*, Gboard and QuickType) are adopting FL to improve service quality while protecting user privacy. As a result, concurrent multiple FL jobs is very common in the HFLMEN system. Recent work [3]–[5] focuses on optimizing client selection for single FL job, which is difficult to obtain the optimal performance due to the inability to handle the resource competition among multiple FL jobs. Other work on resource allocation [6]–[9] assumes that the clients participating in each FL job are always available throughout training process. However, this assumption is unrealistic. Because of UE mobility and unreliable 5G wireless channel, clients who are previously available may be disconnected and become unavailable at a later time before training completes.

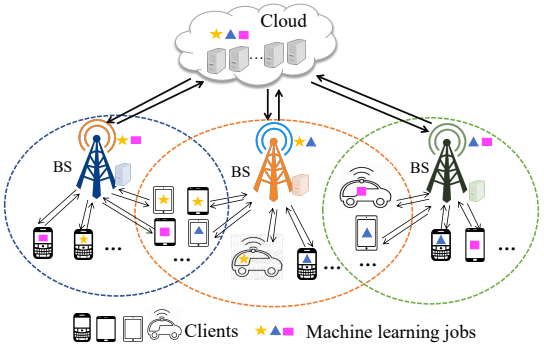


Fig. 1: Illustration of hierarchical Federated Learning in a MEN.

In this paper, we provide "joint resource allocation and client selection for multiple FL jobs" problem design, which is summarized as follows:

- We study the joint resource allocation and client selection problem for multiple FL jobs, aiming to minimize energy consumption and learning time by optimizing: (i) the CPU frequency and transmission power of UEs, and (ii) the client selection of FL jobs as well as the wireless connections between UEs and BSs (§IV-B).
- We design a decoupling algorithm to optimize learning time and energy consumption separately. We present a slowest job first policy based algorithm to find the achievable minimum learning time and use an optimization based algorithm to minimize the energy consumption by optimizing client selection and resource allocation under the learning time constraint. (§V).
- We evaluate the effectiveness of our approach (§VI) under a wide-spectrum of simulation scenarios. The experimental results show that compared to prior work, our approach achieves a significant better performance, reducing up to 75.7% energy consumption and 38.5% learning time, respectively.

§II reviews the related work and §VII concludes this paper.

This work was funded by the National Key Research and Development Program of China (2019YFB1802800) and the National Natural Science Foundation of China (62102066). This project is also supported by CNKL-STISS.

II. Related Work

Federated Learning is an emerging technique. A lot of recent work has particularly focused on developing new platforms [10], protocols [3, 11], and algorithms [6, 12] to enhance the performance of FL in MENs. Due to the very limited resource capacity and instability of clients, some more recent work studies how to improve training performance and reduce training costs (*i.e.*, energy consumption) for FL jobs by optimizing client selection and resource allocation. For example, Nishio *et al.*, [3] first study the client selection problem and propose a protocol for client selection. Jin *et al.*, [4] and Xu *et al.*, [5] study the client selection optimization under long-term resource usage and energy constraints. Tran *et al.*, [6, 7] propose an algorithm to reduce the FL learning time and energy consumption of UEs with heterogeneous power and computing resources. Yang *et al.*, [8], Xu *et al.*, [5] and Shi *et al.*, [9] focus on the allocation of bandwidth and channel resource for achieving long-term performance guarantee, minimizing FL delay, and maximizing model accuracy within a given total learning time budget, respectively. All the above work concentrates on a single FL job scenario. However, multi-job coexistence has become the norm with the large-scale deployment of FL, and prior work will become less efficient and inapplicable in this scenario. Our paper differs from prior work in that we specially consider multiple concurrent FL jobs and study a client selection and resource allocation problem under resource constraints.

III. Model, Protocol and System

We consider a MEN consisting of one cloud server, a set of BSs $\mathcal{M} = \{1, 2, \ldots, M\}$ and a set of UEs that can participate in training a set of FL jobs $\mathcal{J} = \{1, 2, \ldots, J\}$.

A. FL model

We use the same FL model for all jobs, and for simplicity, we describe the adopted FL model below by using one FL job. We consider a FL job j that involves N clients, where each client i has a local dataset $\mathcal{D}_{i,j}$ for training FL job j. Typically, the learning task of client i is to train the model parameter w over its local dataset with loss function $F_{i,j}(w)$.

$$F_{i,j}(w) = \frac{1}{|\mathcal{D}_{i,j}|} \sum_{d \in \mathcal{D}_{i,j}} f_{i,j}(w, d) \tag{1}$$

$f_{i,j}(w, d)$ is the loss function of client i for a sample data $d \in \mathcal{D}_{i,j}$, which captures the FL performance over input data and corresponding output.

FL training problem is to compute a global FL model with the minimum global loss function, which is formulated as

$$\min F_j(w) = \sum_{i=1}^{N} \frac{|\mathcal{D}_{i,j}|}{|\mathcal{D}_j|} F_{i,j}(w) \tag{2}$$

where $\mathcal{D}_j = \bigcup_{i=1}^{N} \mathcal{D}_{i,j}$.

B. FL protocol

To solve problem (2), we adopt the FEDL protocol proposed in [6]. Roughly speaking, FEDL uses an iterative training scheme, requiring K global communication rounds for global model updates. In each global round, the cloud server broadcasts the latest global model parameters and gradients to all clients via BSs. Each client performs κ_j local computation rounds and uploads the locally learned model parameters and gradients back to the cloud for global updating.

More specifically, UEs use their local datasets to update local model $w_{i,j}^k$ in global training round k. To this end, each client i computes the local FL problem (3) and can obtain a θ-approximation solution $w_{i,j}^k$ of (3) satisfying $\|J_{i,j}^k(w_{i,j}^k)\| \leq \theta \|J_{i,j}^{k-1}(w_j^{k-1})\|$, where $\theta \in (0, 1)$.

$$\min J_{i,j}^k(w) = F_{i,j}(w) + \langle \eta \nabla F_j(w_j^{k-1}) - \nabla F_{i,j}(w_j^{k-1}), w \rangle \tag{3}$$

The server updates the global model and gradient as follows:

$$w_j^k = \sum_{i}^{N'} \frac{|\mathcal{D}_{i,j}|}{|\mathcal{D}_j|} w_{i,j}^k, \quad \nabla F_j(w^k) = \sum_{i}^{N'} \frac{|\mathcal{D}_{i,j}|}{|\mathcal{D}_j|} \nabla F_{i,j}(w_{i,j}^k)$$

The problem (2) achieves a global model convergence of w^k when its satisfies

$$F_j(w_j^k) - F_j(w_j^*) \leq \epsilon, \quad \forall k \geq K_j \tag{4}$$

where $\epsilon > 0$ is an arbitrary small constant and w_j^* is the optimal solution to (2).

The convergence analysis of FEDL shows that FEDL convergence is independent of the number of clients and has no restriction on the heterogeneity of UE data under assumptions of strongly convex and smooth loss functions. We refer readers to [6] for more details. Given this superior property of FEDL, we apply FEDL protocol to train each FL job to ensure convergence and focus on resource management to improve training performance when multiple FL jobs compete for resources.

C. System overview

We proposed CEEFL, a communication and energy efficient federated learning paradigm, which optimizes the client selection and resource allocation for multiple concurrent FL jobs to reduce overall training time and energy consumption. Fig. 2 shows the FL with client selection and resource allocation in each global round. Before each global training round, the MEN scheduler optimizes the client selection and resource allocation for all active FL jobs according to client availability and available resource status. To avoid the latency of the optimization from blocking the main training process, the scheduler runs the optimization module offline because it does not rely on information of the main training process. Then, when a new global round starts, FL jobs can immediately use the decision result for the client selection and resource allocation and follow the FEDL protocol to proceed with the training.

IV. Problem Description and Formulation

We consider the energy consumption and learning time minimization (ECLTM) problem for multiple FL jobs in MEN. To this end, we optimize resource allocation (computation and communication resources) and client selection for FL jobs in every global round. In the following, we first introduce the computation and communication models used in this work and then present optimization formulation for the studied problem.

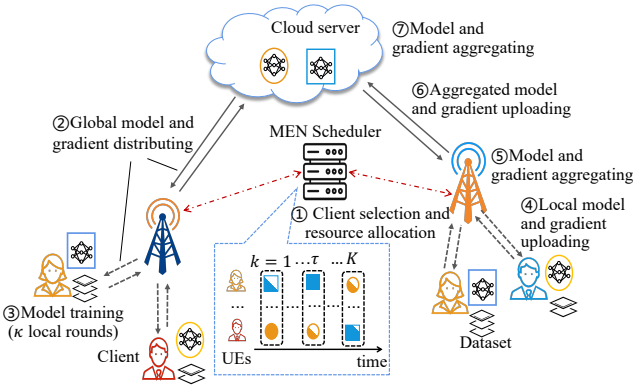


Fig. 2: FL with client selection and resource allocation.

A. Computation and Communication Model

In the training procedure, clients of FL need to train and share ML models, which incur computing and communication resources consumption. As shown in Fig. 2 , clients perform the three steps in each global round: 1) Downloading the global model and gradients from the cloud server via BSs, 2) Training ML model through κ local computation rounds over the local dataset, and 3) Uploading the local model and gradients to the server via BSs. The following introduces the computation model used in step 2) and the communication model adopted in steps 1) and 3).

Computation Model: Let $c_{i,j}$ denote the number of CPU cycles required to process one sample of FL job j on client i and $f_{i,k}$ denote the CPU-cycle frequency of client i in global round k. Then, the time for client i to train the local model of FL job j using its dataset $\mathcal{D}_{i,j,k}$ with a size of $D_{i,j,k}$ in global round k is

$$T^{\text{cp}}_{i,j,k} = \kappa_j \frac{c_{i,j} D_{i,j,k}}{f_{i,k}} \tag{5}$$

where κ_j is the number of local computation rounds required by job j. As in [6], we denote the effective capacitance coefficient of the computing chipset of client i by $\alpha_i/2$. Then, the corresponding CPU energy consumption of client i for local computation of job j in global round k can be computed by

$$E^{\text{cp}}_{i,j,k} = \kappa_j \frac{\alpha_i}{2} c_{i,j} D_{i,j,k} {f_{i,k}}^2 \tag{6}$$

Communication Model: Regarding the global model downloading phase from the cloud to each selected UE, we use Shannon capacity to estimate the downlink transmission rate $r^{\downarrow}_{i,m,k}$, which can be calculated by $b^{\downarrow}_m \log_2[1+\frac{p^B_m h_{i,m,k}}{N_0 b^{\downarrow}_m}]$, where $b^{\downarrow}_m$, p^B_m, $h_{i,m,k}$ and N_0 are the BS broadcast bandwidth, BS transmission power, average channel gain, and background noise power spectral density, respectively. Then, the time for client i to download a unit of data from the server via BS m is calculated by

$$T^{\downarrow}_{i,m,k} = \frac{1}{r^{\downarrow}_{i,m,k}} + \frac{1}{r^C_m} \tag{7}$$

where r^C_m is the data rate between BS m and the cloud server.

Regarding the local model uploading phase of UEs, we consider an OFDMA system with B sub-channel in total. The uplink transmission rate $r^{\uparrow}_{i,m,k}$ can be computed by $b_0 \log_2[1 + \frac{p_{i,k} h_{i,m,k}}{N_0 b_0}]$, where b_0, and $p_{i,k}$ denote bandwidth of each sub-channel and transmission power of UE i, respectively. The time for UE i to upload a unit of data to the server via BS m can be computed by

$$T^{\uparrow}_{i,m,k} = \frac{1}{r^{\uparrow}_{i,m,k}} + \frac{1}{r^C_m} \tag{8}$$

Accordingly, the communication time for UE i to upload and download a unit of data via BS m in global round k is

$$T^{\text{cm}}_{i,m,k} = T^{\downarrow}_{i,m,k} + T^{\uparrow}_{i,m,k} \tag{9}$$

The energy consumption of client i in transmitting a unit of data is the product of transmission power and transmission time, which is computed by

$$E^{\text{cm}}_{i,m,k} = \frac{p_{i,k}}{r^{\uparrow}_{i,m,k}} \tag{10}$$

Note that we only consider the energy consumption of uploading data on the UE side and omit that on the BS side because BS usually has sufficient power supplement like prior work [13].

TABLE I: Summary of key notations

Symbol	Description
$\mathcal{M}$	Set of base stations
$\mathcal{J}$	Set of FL jobs
$\mathcal{U}_k$	Set of available UEs at global round k
$\mathcal{A}_{j,k}/\mathcal{B}_{m,k}$	Set of UEs having training data for job j/covered by BS m in round k
B	Capacity of each BS, *i.e.*, the number of sub-channels
W_j	Minimal number of clients required by FL job j
V_j	Size of model parameters (gradients) of FL job j
K_j	Number of global communication rounds of FL job j
$p^{\min}_i/p^{\max}_i$	Minimum/Maximum transmission power of UE i
$f^{\min}_i/f^{\max}_i$	Minimum/Maximum CPU frequency of UE i
$w^k_{i,j}/w^k_j$	Model parameters learned by UE i of FL jobs j in the global round k
$\nabla F_{i,j}/\nabla F_j$	Model gradients learned by UE i of FL jobs j
Internal and decision variables	
$x_{i,j}/x_{i,j,k}$	Whether UE i is selected to train job j in round k or not
$y_{i,m}/y_{i,m,k}$	Whether UE i is connected to BS m in round k or not
$p_{i,k}$	Transmission power of UE i in round k
$f_{i,k}$	Number of CPU cycles of UE i in round k
$T_k/T_{j,k}$	Learning time elapsed till global round k (for job j)
$E_{j,k}$	Energy consumption for job j in global round k

B. Problem Formulation

Base on the models in §III-A and IV-A, the resource allocation and client selection optimization problem with the objective of minimizing learning time and energy consumption can be formulated as the following mixed-integer nonlinear programming (11).

$$\text{Minimize} \quad (\sum_j T_{j,K_j}, \sum_j \sum_{k \le K_j} E_{j,k}) \tag{11}$$

$$T_{j,k} \ge \max_j T_{j,k-1} + (T^{\text{cp}}_{i,j,k} + 2V_j \sum_m T^{\text{cm}}_{i,m,k} y_{i,m,k}) x_{i,j,k}, \quad \forall i \in \mathcal{U}_k, j \in \mathcal{J}, k \le K_j \tag{12}$$

$$E_{j,k} = \sum_i (E^{\text{cp}}_{i,j,k} + 2V_j \sum_m E^{\text{cm}}_{i,m,k} y_{i,m,k}) x_{i,j,k}, \quad \forall j \in \mathcal{J}, k \le K_j \tag{13}$$

$$\sum_{i \in \mathcal{A}_{j,k}} x_{i,j,k} = W_j, \qquad \forall j \in \mathcal{J}, k \le K_j \tag{14}$$

$$\sum_j x_{i,j,k} \le 1, \qquad \forall i \in \mathcal{U}_k, k \le K_j \tag{15}$$

$$\sum_{i \in \mathcal{B}_{m,k}} y_{i,m,k} \leq B, \quad \forall m \in \mathcal{M}, k \leq K_j \quad (16)$$

$$\sum_{m} y_{i,m,k} = \sum_{j} x_{i,j,k}, \quad \forall i \in \mathcal{U}_k, k \leq K_j \quad (17)$$

$$x_{i,j,k} \in \{0,1\}, \quad \forall i \in \mathcal{U}_k, j \in \mathcal{J}, k \leq K_j \quad (18)$$

$$y_{i,m,k} \in \{0,1\}, \quad \forall i \in \mathcal{U}_k, m \in \mathcal{M}, k \leq K_j \quad (19)$$

$$p_i^{\min} \leq p_{i,k} \leq p_i^{\max}, \quad \forall i \in \mathcal{U}_k, k \leq K_j \quad (20)$$

$$f_i^{\min} \leq f_{i,k} \leq f_i^{\max}, \quad \forall i \in \mathcal{U}_k, k \leq K_j \quad (21)$$

The optimization objectives is expressed as (11). The learning time and the energy consumption for each global training round is defined by (12) and (13), where binary variables $x_{i,j,k} = 1$ if UE i participates in the training of job j in round k and binary variables $y_{i,m,k} = 1$ if UE i connects to BS m in round k. $T_{j,k}$ denotes the time elapsed till global round k. Note that as many existing work [7], we consider the model parameters and the gradients of an FL job have the same size, denoted by V_j for job j, because they generally have the same dimension. Constraints (14) ensure that the number of selected UEs should meet the minimal number of clients required by each FL job, where $\mathcal{A}_{j,k}$ denote the set of UEs that have the training dataset of job j in round k. Constraints (15) ensure that each UE can only participate in at most one FL job in each global round. Constraints (16) ensure that the number of connections to BS should not exceed capacity, where $\mathcal{B}_{m,k}$ is the set of UEs in the coverage range of BS m in round k. Constraints (17) ensure that the participating UEs should connect to BS. Constraints (18)-(21) give the feasible range of variables.

Problem (11) has complex resource constraints and multi-dimensional decision variables, which is non-convex and known to be intractable to solve in real-time [5]. Therefore, we resort to designing efficient heuristics to solve this problem as explained in §V.

V. Algorithm Design

Since learning time and energy consumption are generally two conflicting objectives, it is difficult to optimize them at the same time. So, we propose a two-step solution. First, we optimize the learning time and propose a greedy algorithm based on SJF policy to calculate the minimum achievable learning time required to complete the next global round of all FL jobs. Then, we use this calculated learning time as a budget, and optimize client selection and resource allocation through a heuristic based on LP rounding technique to minimize energy consumption.

A. Optimizing learning time

We use a greedy algorithm based on the slowest job first (SJF) policy to compute the achievable minimum learning time required to complete a specific global training round of all pending FL jobs. To this end, we tentatively assign clients for all pending jobs at the current training round and compute the minimum learning time. Note that the client selection results in this step are only to determine the achievable minimum learning time, not the final solution. We will further optimize client selection according to the energy consumption, as shown in §V-B.

Algorithm 1 shows the procedure of how to compute the minimum learning time T_k for global training round k. We let UEs train model at maximum CPU frequency and uploading data at maximum transmission power to decrease the learning time. We calculate the learning time for each UE when they participate in different FL jobs and connecting to different BSs. Then, we estimate the learning speed of jobs by calculating the average learning time of all possible clients and BS connection (Line 2) Generally, a job with a longer average learning time indicates that it has a slower learning speed and may become straggler in the system. To alleviate the effect of stragglers, we let slower jobs to select clients with more resources and then reduce their learning time. To this end, we select clients for pending jobs in a slowest job first (SJF) manner, giving priority to FL jobs with slower learning speed, under constraints (14)-(17) (Line 3-8). After obtaining the client selection solution, we can easily compute the minimum learning time T_k according to the computation and communication model in §IV-A.

Algorithm 1: SJF-based greedy algorithm for optimizing learning time ($\forall k$)

Input : A set $\mathcal{J}_k$ of unfinished FL job, a collection of available clients $\mathcal{U}_k = \mathcal{A}_{1,k} \cup \mathcal{A}_{2,k} \cup \cdots \cup \mathcal{A}_{|\mathcal{J}_k|,k}$

Output: Minimum learning time T_k

1 For every client candidate $i \in \mathcal{U}_k$, compute the learning time $T_{i,m,j}$ to participate in current training round for job j with the maximum CPU frequency $f_i^{\max}$ and transmission power $p_i^{\max}$ when connected to BS m;

2 For each job j, compute $t_j^{avg} = \frac{\sum_{i,m} T_{i,m,j}}{\sum_m |\mathcal{B}_{m,k} \cap \mathcal{A}_{j,k})|}$;

3 Sort FL jobs in descending order of $\{t_j^{avg}, \forall j \in \mathcal{J}_k\}$;

4 Initialize $T_k \leftarrow 0$;

5 **for** $j \in \mathcal{J}_k$ **do**

6 Greedily assign clients for job j in a smallest $T_{i,m,j}$ first manner, subject to constraints (14)-(17), and compute the learning time $T_{j,k}$ of job j;

7 Update T_k to $T_{j,k}$ if $T_{j,k} > T_k$.

8 **end**

B. Optimizing energy consumption

Now we optimize the energy consumption by optimizing the client selection, BS connection and resource allocation for all FL jobs, under the time budget T_k. However, this optimization problem is still very hard to solve as it also includes integer variables and nonlinear constraints. Therefore, we determine the client selection and BS connection first and then compute the optimal resource allocation (*i.e.*, CPU frequency and transmission power) for selected clients.

Determine client selection and BS connection. In this step, we take the T_k found by Algorithm 1 as a constraint, and to be consistent, we assume that each client candidate uses the maximum CPU frequency and the maximum transmission power. Let binary variables $z_{i,m,j}$ capture the client selection and BS connection decisions, namely $z_{i,m,j} = 1$ if client i sets up a connection with BS m to participate in training job j. We can formulate the client selection and BS connection optimization problem with the objective of minimizing the energy consumption as an integer linear programming (22).

$$\text{Minimize} \quad \sum_{j} E_{j,k} \quad (22)$$

$$E_{j,k} = \sum_{i} (E_{i,j,k}^{\text{cp}} + 2V_j \sum_{m} E_{i,m,k}^{\text{cm}}) z_{i,m,j}, \quad \forall j \quad (23)$$

$$\max_j T_{j,k-1} + (T^{\text{cp}}_{i,j,k} + 2V_j \sum_m T^{\text{cm}}_{i,m,k}) z_{i,m,j} \le T_k, \forall i, j \tag{24}$$

$$\sum_{i \in \mathcal{A}_j} \sum_m z_{i,m,j} = W_j, \qquad \forall j \tag{25}$$

$$\sum_j \sum_m z_{i,m,j} \le 1, \qquad \forall i \tag{26}$$

$$\sum_{i \in \mathcal{B}_m} \sum_j z_{i,m,j} \le B, \qquad \forall m \tag{27}$$

$$z_{i,m,j} \in \{0,1\}, \quad \forall i, m, j \tag{28}$$

To quickly find the feasible solution, we first relax the integer variables in problem (22) and solve it to get the fractional results $\{\widetilde{z}_{i,m,j}, \forall (i,m,j)\}$ and then compute the satisfied integer results $\{\overline{z}_{i,m,j}, \forall (i,m,j)\}$ under constraints (25)-(27) by using the randomized rounding technique as work [14]. Finally, we can obtain the solution of client selection via $\{x_{i,j} = \sum_m \overline{z}_{i,m,j}, \forall (i,j)\}$ and that of BS connection via $\{y_{i,m} = \sum_j \overline{z}_{i,m,j}, \forall (i,m)\}$.

Optimize CPU frequency and transmission power. Since CPU frequency and transmission power on UEs are independent of each other, we separately optimize them to compute the optimal CPU frequency and transmission power for selected clients. We can know from the communication model and computation model in §III that energy consumption of computation and communication decreases as computation time and communication time increases, respectively. We also know that all jobs will not start the global round $k+1$ before finishing round k, which asks all participating UEs to wait even it finishes the training before the learning time T_k. Therefore, to minimize the energy consumption, we let the learning time of all UEs be as close as possible to T_k. To this end, given the value of T_k, we first fix p_i to $p_i^{\max}$ and calculate f_i^* and then calculate p_i^* according to f_i^* as follows.

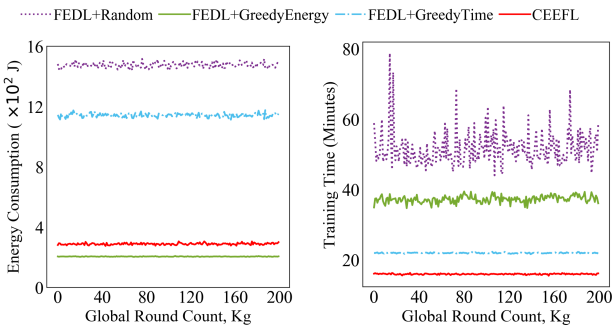
$$f_i^* = \min\{\max\{\frac{c_{i,j} D_{i,j,k} \kappa_j}{T_k - 2V_j T^{\text{cm}}_{i,m,k}}, f_i^{\min}\}, f_i^{\max}\}$$

$$p_i^* = \min\{\max\{(2^{\frac{2V_j}{b_0 (T_k - T^{\text{cp}}_{i,j,k} - 2V_j T^{\downarrow}_{i,m,k} - \frac{2V_j}{r^C_m})}} - 1)\frac{N_0 b_0}{h_{i,m,k}}, p_i^{\min}\}, p_i^{\max}\}$$

VI. Performance Evaluation

A. Simulation Setup

In our simulation, we regularly deploy $M = 9$ base stations (BSs) in a 500m×500m area, where each BS covers a region with a radius of 150m that contains uniformly distributed 1000 UEs. As in previous work [8], the standard deviation of shadow fading is 8 dB and the path loss model follows $128.1 + 37.6 \log d_{i,m}$, where $d_{i,m}$ is the distance between UE i and BS m (in km). The noise power spectral density is -174dBm/Hz. We use the popular computer vision dataset CIFAR-10 to train representative ML models [15], with a size ranging from 14MB to 171MB. The maximum (minimum) CPU frequency of UEs is uniformly distributed in $1.0 - 2.0$ GHz (0.3-0.6 GHz). The parameter $c_{i,j}$ is uniformly distributed in $10 - 30$ cycles/bit, and α_i is 2×10^{-28}. Each UE moves within a range of 150m between two consecutive global training rounds. The number of data samples on UEs for FL jobs is uniformly distributed in $100 - 1000$. The maximum (minimum) transmission power of UEs is uniformly distributed in $0.01 - 0.1$ W ($0.005 - 0.1$ W). The transmission power of BSs is 1W [13]. Each BS can connect with up to 100 UEs, and the uplink (downlink) bandwidth capacity is 180kHz (20MHz) [16]. Besides, we set the bandwidth between BS and the cloud server to 100Mbps.



(a) Energy consumption vs. K_g (b) Learning time vs. K_g

Fig. 3: The performance of different approaches during training process.

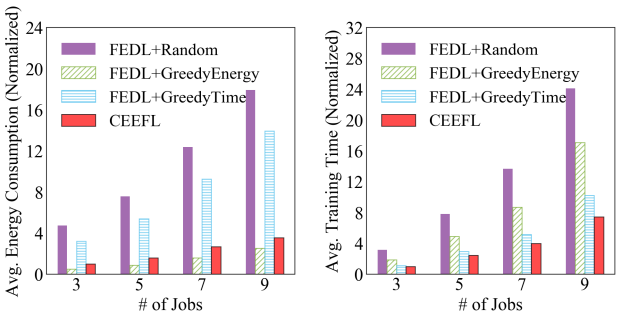
We compare CEEFL with the following client selection approaches: 1) FEDL+Random: it randomly selects clients for FL jobs and feasible CPU frequency and transmission power for UEs, 2) FEDL+GreedyEnergy: it greedily selects clients for FL jobs according to energy consumption by using $p^{\min}$ and $f^{\min}$, and 3) FEDL+GreedyTime: it greedily selects clients for FL jobs according to UEs' learning time by using $p^{\max}$ and $f^{\max}$. To make results comparable, all simulated approaches use the FEDL [6] protocol and the same training parameters (e.g., the number of clients) to train FL jobs to ensure the same convergence performance. The only differences between these compared approaches are the client set and the resource allocation solution they use to train FL jobs.

B. Numerical Results

We first investigate how different approaches perform during the learning process. Fig. 3 report the energy consumption and learning time in each global round. FEDL+Random has the highest energy consumption and the longest learning time, and learning time fluctuates greatly due to randomness. FEDL+GreedyEnergy achieves the lowest energy consumption but a longer learning time than both FEDL+GreedyTime and CEEFL. The energy consumption of FEDL+GreedyTime is much higher than FEDL+GreedyEnergy and CEEFL. It is interesting that the learning time of FEDL+GreedyTime comes after CEEFL. This performance gain of CEEFL benefits from the joint optimization of client selection for multiple FL jobs. However, to achieve a shorter learning time, CEEFL consumes more energy than FEDL+GreedyEnergy because some UEs may use a higher CPU frequency and a larger transmission power for training.

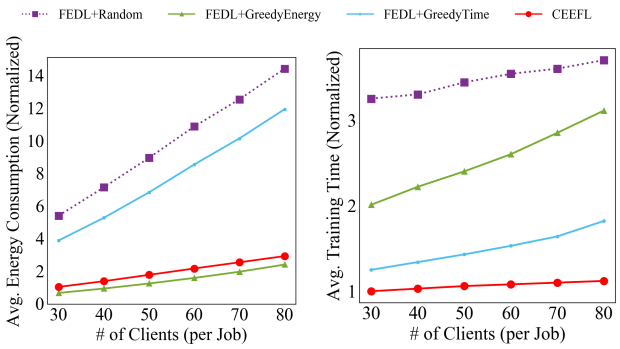
Next, we evaluate the effectiveness of CEEFL under a wide spectrum of experiments by varying the following parameters used to generate workloads: 1) the number of concurrent FL jobs and 2) the number of clients required to participate in each job. We conduct 20 runs under every parameter setting and report the average results.

Now we evaluate the performance of CEEFL under different numbers of concurrent FL jobs. Fig. 4 shows that, as the number of FL jobs increases, the energy consumption of FEDL+Random and FEDL+GreedyTime increases dramatically, and the learning time of FEDL+Random and FEDL+GreedyTime



(a) Energy consumption (b) Learning time

Fig. 4: Impact of number of jobs



(a) Energy consumption (b) Learning time

Fig. 5: Impact of number of clients

increases quickly. However, CEEFL achieves better performance in both energy consumption and learning time. Compared to FEDL+GreedyTime, CEEFL reduces the learning time by as much as 27.2% and the energy consumption by up to 74.5%, respectively. Compare to FEDL+GreedyEnergy, CEEFL improves learning speed by 2.3× while incurring up to 40.7% increase in energy consumption. According to our experiment, CEEFL increases up to 17.3KJ energy consumption compared to FEDL+GreedyEnergy but can reduce learning time as much as 56.2 hours, which is beneficial for FL jobs pursuing fast training. The reason for the poor performance of FEDL+GreedyTime and FEDL+GreedyEnergy is that they separately optimize for each FL job. As the number of FL jobs increases, more UEs are involved in the training process. The FL jobs processed by FEDL+GreedyTime and FEDL+GreedyEnergy early tend to occupy UEs with better computation and bandwidth capacities. As a result, the performance of remaining jobs will be greatly decreased, leading to poor performance on average.

At last, we evaluate the performance of compared approaches by varying the number of clients required to participating in each FL job. Fig. 5 shows that energy consumption and learning time grow almost linearly with the number of clients. This is as expected. However, CEEFL cam always achieves the shortest learning time with rather low energy consumption. More specifically, when per FL job requires 80 clients, CEEFL achieves 38.5% shorter learning time and 75.7% less energy consumption compared to FEDL+GreedyTime, and reduces 64% learning time than FEDL+GreedyEnergy with just 21.9% more energy consumption. Moreover, Fig. 5(b) show that as the number of clients increases, the energy consumption of FEDL+Random and FEDL+GreedyTime increases sharply, while CEEFL grows much slower as the optimal solution FEDL+GreedyEnergy. This should thank CEEFL allows the selected clients to flexibly adjust computation and communication resources to reduce energy consumption under training time constraints. Regarding the learning time, FEDL+GreedyTime and FEDL+GreedyEnergy increases fast as the number of clients increases. While as shown in Fig. 5(b), CEEFL can maintain a much stable performance.

VII. Conclusion

This work studies the energy consumption and learning time minimization problem for multiple Federated Learning (FL) jobs by jointly optimizing client selection and resource allocation. We propose a decoupling algorithm to optimize energy consumption and learning time separately. First, we design a greedy algorithm to find the minimum achievable learning time for per global training round. Then, we take this minimum learning time as a constraint and jointly optimize client selection, BS connection, CPU frequency and transmission power to minimize the energy consumption. Trace-driven simulations show that our algorithm can greatly reduce learning time and energy consumption, compared to prior work on single job optimization.

References

[1] P. Kairouz, H. B. McMahan, B. Avent *et al.*, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[2] W. Y. B. Lim, N. C. Luong *et al.*, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[3] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *IEEE ICC*, 2019, pp. 1–7.

[4] Y. Jin, L. Jiao, Z. Qian, S. Zhang, S. Lu, and X. Wang, "Resource-Efficient and Convergence-Preserving Online Participant Selection in Federated Learning," in *IEEE ICDCS*. IEEE, 2020, pp. 606–616.

[5] J. Xu and H. Wang, "Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective," *IEEE Transactions on Wireless Communications*, vol. 20, no. 2, pp. 1188–1200, 2020.

[6] C. T. Dinh, N. H. Tran, M. N. Nguyen *et al.*, "Federated learning over wireless networks: Convergence analysis and resource allocation," *IEEE/ACM Transactions on Networking*, vol. 29, no. 1, pp. 398–409, 2020.

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

[8] Z. Yang, M. Chen, W. Saad *et al.*, "Delay minimization for federated learning over wireless communication networks," *arXiv preprint arXiv:2007.03462*, 2020.

[9] W. Shi, S. Zhou, Z. Niu, M. Jiang, and L. Geng, "Joint device scheduling and resource allocation for latency constrained wireless federated learning," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 453–467, 2020.

[10] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. B. McMahan *et al.*, "Towards federated learning at scale: System design," *arXiv preprint arXiv:1902.01046*, 2019.

[11] M. S. H. Abad, E. Ozfatura, D. Gunduz, and O. Ercetin, "Hierarchical federated learning across heterogeneous cellular networks," in *IEEE ICASSP*. IEEE, 2020, pp. 8866–8870.

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

[13] M. Chen, Z. Yang, W. Saad *et al.*, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 269–283, 2020.

[14] L. Guo, J. Pang, and A. Walid, "Joint placement and routing of network function chains in data centers," in *IEEE INFOCOM*, 2018, pp. 612–620.

[15] https://keras.io/api/applications/, accessed: 2020-10-01.

[16] B. A. Bjerke, "LTE-advanced and the evolution of LTE deployments," *IEEE Wireless Communications*, vol. 18, no. 5, pp. 4–5, 2011.