Energy-Efficient Distributed Machine Learning at Wireless Edge with Device-to-Device Communication

Rui Hu*, Yuanxiong Guo†, and Yanmin Gong*
* Department of Electrical and Computer Engineering, University of Texas at San Antonio, San Antonio, TX 78249
† Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX 78249
rui.hu@utsa.edu, yuanxiong.guo@utsa.edu, yanmin.gong@utsa.edu

***Abstract*—This paper considers a federated edge learning (FEL) system where a base station (BS) coordinates a set of edge devices to train a shared machine learning model collaboratively. One of the fundamental issues in such systems is maintaining the learning performance with the limited and heterogeneous resource capabilities of edge devices. Our goal is to improve the energy efficiency of edge devices in FEL by mitigating the temporal and spatial heterogeneity of their energy resources. Specifically, to balance the heterogeneous energy levels among edge devices, energy-hungry devices can offload their data to nearby devices that have sufficient energy via device-to-device (D2D) communication links at low transmission overheads. Besides, to mitigate the impact of the time-varying energy level of a device, data collected by edge devices can be queued to be processed when sufficient energy is available. To compute the optimal offloading and queuing strategies, we propose an online control algorithm based on Lyapunov optimization to determine the amount of data to be offloaded, queued, and processed at each time slot. Our simulation results on the real-world dataset demonstrate that our approach achieves a better overall energy efficiency than baselines.**

***Index Terms*—Federated learning, energy efficiency, wireless edge, device-to-device communication, Lyapunov optimization**

I. Introduction

The confluence of Internet of Things (IoT) and machine learning technologies paves the way for intelligence at the network edge. The International Data Corporation (IDC) estimates that there will be 55.7 billion IoT devices by 2025, generating almost 80B zettabytes of data [1], and those data could be processed and analyzed using machine learning approaches to improve human lives. In traditional machine learning approaches, a cloud server first collects those raw data from IoT devices and then uses the collected data to train a machine learning model. However, as the volume of data generated at the edge increases, the traditional cloud-based machine learning approach is no longer suitable for IoT settings [2]. Transmitting the raw data from massive IoT devices to a cloud server incurs expensive communication costs over the core network and causes high transmission delays. In addition, due to the increasing privacy concerns, users will be unwilling to share their data to a cloud server for machine learning purposes [3], [4].

The above drawbacks of traditional cloud-based machine learning approaches motivate the development of distributed machine learning methods that push machine learning from the cloud to the edge. As one of the most popular distributed machine learning methods, federated edge learning (FEL) was proposed to improve the privacy and communication efficiency of training machine learning models over distributed data [5]. Specifically, in FEL, edge devices train a shared machine learning model collaboratively in an iterative manner under the coordination of the cloud server while keeping their data locally. During the training, edge devices use their local datasets to update a shared model downloaded from the server and then transmit the updated models to the server for aggregation. Since the model parameters transmitted between edge devices and the server contain much less sensitive information and are much smaller than the raw data, FEL greatly reduces the communication cost and privacy leakage. However, FEL faces several key challenges that hinder its wide adoption in real applications. One of the challenges is the limited and heterogeneous resources of edge devices, which will significantly impact the quality of the trained model. As edge devices in FEL are involved in the model training process, the resource capabilities of devices, such as network connectivity, computational speed, memory size, or battery level, will influence the qualities of local models collected by the server and hence hinder the convergence of global model. Therefore, it is essential to manage the resources of edge devices in FEL efficiently.

Related works on improving the resource efficiency of FEL have been proposed recently [6]–[8]. For example, a joint bandwidth allocation and scheduling policy approach was proposed in [6] to reduce the communication energy consumption of edge devices in FEL. In [7], an energy-efficient FEL scheme was developed by jointly optimizing the computation and communication energy consumption of edge devices. In [8], the transmission power and rate of edge devices and their CPU frequencies are jointly optimized to minimize the overall energy consumption of edge devices. Particularly, these works assumed that all devices have homogeneous and fixed resource capabilities when designing their resource-efficient mechanisms. However, in practice, the resource capabilities of edge devices vary a lot across devices and time.

In this paper, we take the heterogeneity and dynamicity of devices' resource capabilities into consideration to improve

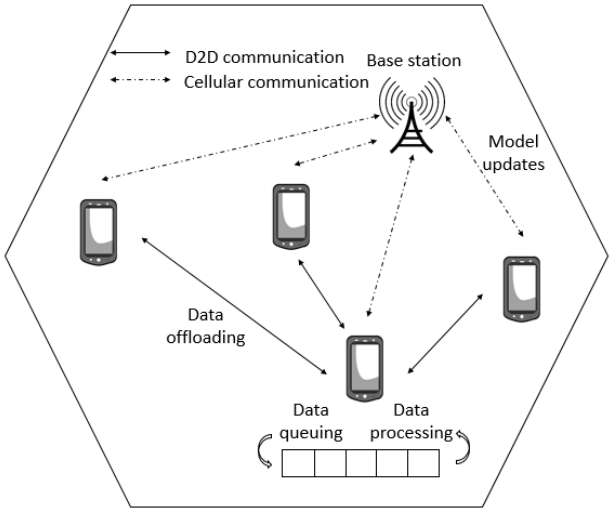

Figure 1: System architecture of our proposed FEL over a two-tier network.

the resource efficiency of FEL. Specifically, we focus on mitigating the influence of the heterogeneous energy levels across devices and the dynamic energy level of each device over time. We propose a novel FEL scheme over a two-tier network to reduce the overall energy cost of edge devices while preserving the learning performance. The basic idea is to use the device-to-device (D2D) network to balance the spatial energy heterogeneity and use the data queuing to mitigate the temporal energy dynamicity. Specifically, we design a data offloading and queuing mechanism to allow energy-hungry devices to either offload their training data to nearby energy-sufficient devices via fast D2D links with low communication cost or queue their data to be processed later until they have sufficient energy. Since the offloading and queuing strategies highly depend on the uncertain device resources, we formulate a stochastic optimization problem and design an online control algorithm based on Lyapunov optimization to determine the amount of data to offload, queue, and process on every device at each time slot during the training. Finally, we compare our scheme with two baselines using the real-world dataset and demonstrate the effectiveness of our scheme in improving energy efficiency.

The rest of this paper is organized as follows. In Section II, we propose our FEL scheme over a two-tier network. In Section III, we formulate the energy cost minimization as a stochastic optimization problem. In Section IV, we solve the formulated problem using Lyapunov optimization and propose an online control algorithm. Finally, we present and discuss our simulation results in Section V and conclude the paper in Section VI.

II. System modeling

A typical FEL system consists of I edge devices and a base station (BS). We consider a FEL system deployed over a two-tier network which consists of a macro-tier and a device-tier as shown in Figure 1. In the macro-tier, the BS communicates with devices through cellular communication. In the device-tier, a device communicates directly with another device via D2D communication without the intervention of a BS or relay device. As the BS serves as a cloud server, the goal of our FEL system is to allow these edge devices to train a global model collaboratively and iteratively under the coordination of the BS.

Due to the heterogeneity and dynamicity of edge devices, devices in this system have different and time-varying energy levels. Here, we assume that each device has a time-varying energy cost for completing the learning task due to its time-varying energy level. Specifically, the energy cost of a device for completing the training task will be high if it does not have sufficient energy, and the energy cost for a device with sufficient energy will be low. In the rest of this work, the difference in the energy costs of devices is defined as the spatial variability of energy cost, whereas the variation of energy cost from one time slot to another is defined as the temporal variability of energy cost.

To balance the spatial variability of energy cost, devices with high energy costs may offload (via D2D links) newly arrived data samples to other devices with lower energy costs. To tackle the temporal variability of energy cost, devices may delay the processing of newly arrived data samples at the current time slot to future time slots when energy cost is low. With local data samples, each device updates the global model downloaded from the BS by performing τ local gradient descent updates. Then, each device sends its local updated model to the BS, which will aggregate the local models to update the global model for the next round of training. This training process will repeat K rounds until the global model converges. Hence, each device will perform $T := K\tau$ local training iterations in total. In the following, we describe the three steps in each training round of our proposed FEL scheme.

A. Data Collection

Let $\mathcal{I} := \{1, \ldots, I\}$ denote the set of I edge devices. At time slot $t \in \{0, \ldots, T-1\}$, each device $i \in \mathcal{I}$ collects a new set of $D_i(t)$ data samples denoted by $\mathcal{D}_i(t)$. Each data sample $d := (\mathbf{x}_d, y_d)$ in $\mathcal{D}_i(t)$ consists of a feature vector $\mathbf{x}_d$ and its associated scalar label y_d. We assume $\mathcal{D}_i(t)$ to be independent identically distributed (i.i.d.) over time slots. Let $\mathcal{D}(t) := \cup_{i\in\mathcal{I}}\mathcal{D}_i(t)$ denote the set of data samples collected by all devices at time slot t.

B. Data Offloading and Queuing

Given the new dataset $\mathcal{D}_i(t)$ at time slot t, device i has two options to deal with each data sample $d \in \mathcal{D}_i(t)$ depending on its energy situation: it can either 1) offload the data sample d to another device $j \in \mathcal{I}, j \neq i$; 2) or queue the data sample d locally for the training on device i. Note that the data samples offloaded to device j will be queued to be processed at incoming time slots. Let $f_{ij}(t)$ denote the number of data samples offloaded from device i to device j and $f_{ii}(t)$ denote

the number of data samples kept locally on device i at time slot t, then the total number of data samples which are queued on device i for local training is $\sum_{j\in\mathcal{I}} f_{ji}(t)$. Each data sample in $\mathcal{D}_i(t)$ should be either offloaded to another device or queued locally, which can be constrained as follows:

$$D_i(t) = \sum_{j=1}^{I} f_{ij}(t), f_{ij}(t) \geq 0, \forall i \in \mathcal{I}. \tag{1}$$

Assume the transmission capacity of the D2D link between device i and device j is B_{ij}, which is the maximum number of data samples that can be transmitted at a time slot. Then, the number of data samples offloaded from device i to device j should satisfy that

$$f_{ij}(t) \leq B_{ij}, \forall i, j \in \mathcal{I}. \tag{2}$$

For the data queue maintained by device i, we define the queue backlog $Q_i(t)$ as the number of data samples waiting in the queue at time slot t, where $Q_i(0)$ is initialized to 0 at time slot 0. Let $\mathcal{G}_i(t)$ represent the set of data samples that are in the queue and will be used for local training at the current time slot t, then the queue dynamics can be expressed as

$$Q_i(t+1) = [Q_i(t) - G_i(t)]^+ + \sum_{j=1}^{I} f_{ji}(t), \forall i \in \mathcal{I}, \tag{3}$$

where $G_i(t) := |\mathcal{G}_i(t)|$ represents the size of $\mathcal{G}_i(t)$, and we can see that the departure and arrival rates of the queue are $G_i(t)$ and $\sum_{j=1}^{I} f_{ji}(t)$, respectively. To maintain the stability of the queue, we need to guarantee that the queues remain bounded as new data samples arrive and are queued, i.e., the average queue backlog over time should be bounded as follows:

$$\overline{Q} := \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i=1}^{I} \mathbb{E}\left[Q_i(t)\right] < \infty. \tag{4}$$

C. Data Processing

To train the model in an online manner, we minimize the following global loss over time slots:

$$\min_{\mathbf{w}} \; L(\mathbf{w}|\mathcal{D}) := \frac{\sum_{t=0}^{T-1} \sum_{i=1}^{I} \sum_{d\in\mathcal{G}_i(t)} l(\mathbf{w}, \mathbf{x}_d, y_d)}{D}, \tag{5}$$

where $\mathbf{w} \in \mathbb{R}^n$ represents an n-dimensional global model; $l(\mathbf{w}, \mathbf{x}_d, y_d)$ is the loss on data sample d; $\mathcal{D} := \cup_{t<T}\mathcal{D}(t)$ represents the set of data samples collected at all time slots; and $D := |\mathcal{D}|$ represents the size of $\mathcal{D}$. The global loss is the weighted average of local losses, i.e.,

$$L(\mathbf{w}|\mathcal{D}) = \frac{\sum_{i=1}^{I} G_i L_i(\mathbf{w}|\mathcal{G}_i)}{D}, \tag{6}$$

where

$$L_i(\mathbf{w}|\mathcal{G}_i) := \frac{\sum_{t=0}^{T-1} \sum_{d\in\mathcal{G}_i(t)} l(\mathbf{w}, \mathbf{x}_d, y_d)}{G_i}, \forall i \in \mathcal{I}. \tag{7}$$

Here, $L_i(\mathbf{w}|\mathcal{G}_i)$ represents the local loss function of device i, where $\mathcal{G}_i := \cup_{t<T}\mathcal{G}_i(t)$ represents the set of all data samples collected at device i, and $G_i := |\mathcal{G}_i|$ represents the size of $\mathcal{G}_i$.

We use the FedAvg algorithm proposed in [3] to solve our online learning problem in a distributed manner. Generally, at each time slot of FedAvg, each device downloads the global model maintained by the BS as its local model and updates the model using its local dataset, then the BS collects and aggregates all the updated local models to renew the global model for the next round of training. Specifically, let $\mathbf{w}(t)$ represent the global model at time slot t. In each training round $k \in [K]$, after receiving the global model from the BS, the device takes the global model as its initial local model and performs τ steps of gradient descent towards minimizing its local loss function to update its local model, i.e., device i performs the following update for τ time slots:

$$\mathbf{w}_i(t) = \mathbf{w}_i(t-1) - \eta \nabla L_i(\mathbf{w}_i(t-1)|\mathcal{G}_i(t)), \forall i \in \mathcal{I}, \tag{8}$$

where $\eta > 0$ is the step size, $\mathbf{w}_i(t)$ represents the local model of device i at time slot t. When $t \mod \tau = 0$, that is, at the beginning of each training round, $\mathbf{w}_i(t)$ is initialized as the global model $\mathbf{w}(t)$ downloaded from the BS. Here, $\nabla L_i(\mathbf{w}_i(t-1)|\mathcal{G}_i(t)) := \sum_{d\in\mathcal{G}_i(t)} \nabla l(\mathbf{w}, \mathbf{x}_d, y_d)/G_i(t)$ represents the gradient of local loss L_i at time slot t. We define the maximum computation capacity of device i as C_i, which is the maximum number of data samples that can be processed by device i for training at each time slot. Therefore, the number of data samples to be processed $G_i(t)$ should be bounded by computation capacity in any time slot, i.e.,

$$G_i(t) \leq C_i, \forall i \in \mathcal{I}. \tag{9}$$

After τ steps of gradient descent, each device sends its local model to the BS, and the BS updates the global model by aggregating the local models as follows:

$$\mathbf{w}(k\tau) = \frac{\sum_{i=1}^{I} H_i(k\tau)\mathbf{w}_i(k\tau)}{\sum_{i=1}^{I} H_i(k\tau)}, \tag{10}$$

where $k\tau$ indicates that the global aggregation happens every τ time slots, $H_i(k\tau) = \sum_{(k-1)\tau\leq t<k\tau} G_i(t)$ is the number of data samples processed at device i since last aggregation. Once the BS gets the new global model, it will broadcast the model to all devices for the next round of training.

III. Cost Minimization Problem Formulation

In this section, we define the cost function for our proposed FEL scheme to find the optimal data offloading, queuing, and processing mechanism that can minimize the energy cost and achieve a desirable learning performance at the same time. Our cost function consists of two components: the energy consumption of devices and the model training loss. Before a global aggregation, the energy consumed by device i consists of 1) the energy to offload data from device i to device j and 2) the energy to process data for local model training. During and after the global aggregation, additional energy is needed for device i to transmit the local model to the BS and then download the new global model from the BS. The transmission energy is proportional to the size of the local model, but it is not related to our design variables. Therefore, in our modeling,

we exclude the energy needed for model transmissions and focus on the energy cost for data offloading and processing. Specifically, we formulate the total energy cost of device i at time slot t as follows:

$$e_i(t) := U_i(t) \left(\sum_{j=1}^{I} c_{ij} f_{ij}(t) + c_i G_i(t) \right), i \in \mathcal{I}. \quad (11)$$

Here, $U_i(t)$ is a random variable that represents the per unit energy cost of device i at time slot t measured in \$/Joule. As we mentioned, the energy cost depends on the energy level, i.e., an energy-sufficient device will have a low per unit energy cost, and vice versa. The total energy consumed by device i at time slot t is $\sum_{j=1}^{I} c_{ij} f_{ij}(t) + c_i G_i(t)$ (in Joules), where c_{ij} is the amount of energy (in Joules) to offload one data sample (which depends on the data transmission rate and transmission power), and c_i is the amount of energy (in Joules) to process one data sample for local training (which depends on the CPU frequency and CPU cycles).

As we want to minimize the energy cost and training loss simultaneously, our overall cost function at time slot t can be formulated as follows:

$$Cost(t) := \sum_{i=1}^{I} \left[e_i(t) + p_i(t)(L(\mathbf{w}_i(t)) - L(\mathbf{w}^*(t))) \right], \quad (12)$$

where $p_i(t) > 0$ is a scaling factor that represents the importance of global training loss, and $L(\mathbf{w}^*(t))$ represents the minimal loss at the optimal point $\mathbf{w}^*(t)$. As the training loss is unknown beforehand, we use the convergence result of our FEL scheme (as given in Theorem III.1) to approximate the training loss gap in (12).

Theorem III.1. *(Convergence of the global loss with respect to the number of processed data) According to [9], the relationship between the global loss and the number of processed data can be expressed as:* $L(\mathbf{w}_i(t)) - L(\mathbf{w}^*(t)) \propto \sqrt{G_i^{-1}(t)}$.

Proof. Full proof is provided in the supplementary material [10]. ■

From Theorem III.1, we can observe that the global training loss gap $(L(\mathbf{w}_i(t)) - L(\mathbf{w}^*(t)))$ in (12) could be approximated by its upper bound $\sqrt{G_i^{-1}(t)}$. Since $\sqrt{G_i^{-1}(t)}$ is monotonically decreasing with $G_i(t)$, minimizing $\sqrt{G_i^{-1}(t)}$ is equivalent to minimizing $-G_i(t)$. In this case, we approximate our cost function in (12) as

$$Cost(t) := \sum_{i=1}^{I} \left[e_i(t) - p_i(t) G_i(t) \right]. \quad (13)$$

In summary, along with all the constraints mentioned before, our optimization problem becomes:

$$\min_{f_{ij}(t), G_i(t)} \overline{Cost} := \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\left[Cost(t) \right], \quad (14)$$

subject to (1), (2), (3), (4), (9).

Our goal is to find the optimal values for data offloading and queuing $f_{ij}(t)$ and data processing $G_i(t)$ that minimize the average cost over time $\overline{Cost}$. The expectation in (14) is with respect to random data arrival $\mathcal{D}_i(t)$ and random energy cost $U_i(t)$. Solving Problem (14) is challenging because optimization variables chosen at time slot t depend on future data arriving at time slots $t+1, t+2, \ldots, T-1$, which are unknown yet. In the following, we use Lyapunov optimization [11] to ensure the stability of our data queues and transform our problem (14) from a long-term problem to an online control problem, where decision variables are chosen based on information in past and present time slots, without requiring prior knowledge about future information.

IV. Our online control algorithm

In this section, we solve Problem (14) using Lyapunov optimization. We define the quadratic Lyapunov function and the one-step conditional Lyapunov drift, respectively, as follows:

$$Y(\mathbf{Q}(t)) \triangleq \frac{1}{2} \sum_{i=1}^{I} Q_i^2(t),$$
$$\Delta(\mathbf{Q}(t)) \triangleq \mathbb{E}\left[Y(\mathbf{Q}(t+1)) - Y(\mathbf{Q}(t)) | \mathbf{Q}(t) \right], \quad (15)$$

where $\mathbf{Q}(t) = [Q_1(t), \ldots, Q_I(t)]$ is a vector concatenating the queue backlogs of all devices at time slot t. The expectation is with respect to the random data arrival and per-unit energy cost. The quadratic Lyapunov function $Y(\mathbf{Q}(t))$ generates a scalar value that indicates the queues' stability. Our objective is to minimize the cost function in (14) while maintaining the stability of the queues $\{Q_i(t)\}_{i \in \mathcal{I}}$. To achieve this, we use the drift-plus-penalty minimization method [12], [13]. This method minimizes an upper bound of the drift-plus penalty at time slot t, i.e., $\Delta(\mathbf{Q}(t)) + V\mathbb{E}[Cost(t)|\mathbf{Q}(t)]$, where V is a scaling parameter used for weighting the drift and penalty.

Algorithm 1 Our Online Control Algorithm Using Lyapunov Optimization

1: Initialize $\mathbf{Q}(0) = \mathbf{0}$
2: **for** $t = 1$ to T **do**
3: Observe the system states $\mathcal{D}_i(t)$ and $U_i(t)$, $\forall i \in \mathcal{I}$, and the queues $\mathbf{Q}(t)$
4: Choose the design variables $f_{ij}^*(t)$, and $G_i^*(t)$ which are the optimal solution to the following problem:

$$\min_{f_{ij}(t), G_i(t)} \sum_{i=1}^{I} \left[Q_i(t) \left(\sum_{j=1}^{I} f_{ji}(t) - G_i(t) \right) \right] + VCost(t) \quad (16)$$

subject to (1), (2), and (9)
5: Update $\mathbf{Q}(t)$ according to the queue dynamics in (3)
6: **end for**

Lemma IV.1. *At time slot t, an upper bound for the drift-plus-penalty expression can be written as follows*

$$\Delta(\mathbf{Q}(t)) + V\mathbb{E}\left[Cost(t)|\mathbf{Q}(t)\right] \leq B_1 + V\mathbb{E}\left[Cost(t)|\mathbf{Q}(t)\right] \sum_{i=1}^{I} Q_i(t)\mathbb{E}\left[\sum_{j=1}^{I} f_{ji}(t) - G_i(t)|\mathbf{Q}(t)\right], \quad (17)$$

where B_1 is a constant.

Proof. Full proof is provided in the supplementary material [10]. ■

Lemma IV.2. *The optimal solution of Problem* (16) *minimizes the upper bound of the drift-plus-penalty in* (17)*.*

Proof. Full proof is provided in the supplementary material [10]. ■

In Lemma IV.1, we give the upper bound of the drift-plus-penalty. We propose an online Lyapunov control algorithm in Algorithm 1 to solve this upper bound. In this algorithm, the queue vector is initialized as $\mathbf{0}$ at first. Then, at each time slot t, we observe the system states: data arrival $\mathcal{D}_i(t)$ and energy cost per unit energy $U_i(t)$ for all $i \in \mathcal{I}$. Finally, we calculate the optimal design variables $f_{ij}^*(t)$ and $G_i^*(t)$ at time slot t by solving Problem (16), which minimizes the right hand side of (17) (as stated in Lemma IV.2). As Problem (16) is linear, it can be solved using simplex methods. In Theorem IV.3, we prove the optimality of our algorithm in bounding the average queue backlog and minimizing the average total loss.

Theorem IV.3. *(Optimality of our online control algorithm) Assuming both $U_i(t)$ and $\mathcal{D}_i(t)$ is i.i.d. over time slots for all $i \in \mathcal{I}$, our online Lyapunov control algorithm ensures that the time-average queue backlog is bounded as: $\overline{Q} \leq (B_1 + VB_2)/\zeta$, where B_1, B_2, and ζ are constants. Moreover, the time-average total cost is within B_1/V of the optimal cost $\overline{g}^*$: $\overline{Cost} \leq B_1/V + \overline{g}^*$.*

Proof. Full proof is provided in the supplementary material [10]. ■

V. Simulation Results

In this section, we present the experimental setup and discuss the simulation results.

A. Experimental Setup

1) Dataset and Baselines: We evaluate our approach for the image recognition task, which uses the modified national institute of standards and technology (MNIST) dataset. The MNIST dataset consists of $70,000$ samples of 28×28 images. We use $60,000$ samples for training and the rest for testing in our experiments. We set the number of edge devices as $I = 10$ and the total number of time slots as $T = 1000$. As the total number of data samples collected at all devices $D = 60,000$, we simulate data arrival on device i at time slot t as a Poisson arrival process with the rate $\lambda = D/IT$. The rate λ represents the average number of data samples arrived on a device at each time slot. We compare our proposed scheme that considers both spatial and temporal variability of energy cost with two baseline schemes: 1) one ignoring spatial variability (i.e., without data offloading); and 2) another one ignoring both spatial and temporal variability (i.e., without data offloading and queuing).

2) Communication and Computation Capability: Here, we assume all communication links between edge devices are active, i.e., a fully connected D2D network. The link capacity B_{ij} between device i and device j, $\forall i, j \in \mathcal{I}, i \neq j$, is set as 5λ, and the computation capacity C_i at device $i, \forall i \in \mathcal{I}$, is set as 5λ.

3) Energy Consumption and Cost: We set the size of one data sample (image) $n_b = 28 \times 28$ (bits), the frequency of the CPU clock $\vartheta = 10^9$ (Hz), the number of CPU cycles required for computing one bit of data $\omega = 40$ (cycles/bit), and the energy consumption coefficient $\psi = 10^{-28}$. Therefore, the energy consumed at device i for processing one data sample $c_i = n_b\vartheta^2\omega\psi = 31.36$ (Joules/sample), $\forall i \in \mathcal{I}$. Furthermore, let each device transmit data with power $P = 0.5$ (W). We model the average data rate for device i to communicate with device j as a Uniform random variable $R_i^j \sim \mathcal{U}(50, 150)$ (Mbps) by default, $\forall i, j \in \mathcal{I}, i \neq j$. Hence, the energy consumed for offloading one data sample from device i to device j is $c_{ij} = Pn_b/R_i^j$ (Joules/sample). Besides, we simulate the per unit energy cost $U_i(t)$ as a Uniform random variable. Specifically, the per unit energy cost of device $i \in \{1, 2, 3, 4, 5\}$ satisfies $U_i(t) \sim \mathcal{U}(2, 8)$ (\$/Joule), and the per unit energy cost of device $i \in \{5, 6, 7, 8, 9, 10\}$ satisfies $U_i(t) \sim \mathcal{U}(6, 12)$ (\$/Joule).

B. Experimental Results

We first study the energy efficiency of our approach compared with the baselines. In Figure 2a, we report the average total cost with respect to the number of time slots. For our approach, we consider three different D2D channel rates, i.e., $R_i^j \sim \mathcal{U}(40, 60)$ (Mbps) with mean 50 (Mbps), $R_i^j \sim \mathcal{U}(50, 150)$ (Mbps) with mean 100 (Mbps), and $R_i^j \sim \mathcal{U}(200, 300)$ (Mbps) with mean 250 (Mbps). We set the control parameter $V = 1$ and the training loss scaling factor $p_i(t) = 100$. We can observe that the baseline scheme that ignores spatial and temporal variability achieves the highest cost, followed by the baseline scheme that ignores only spatial variability, while our scheme achieves the lowest cost. This matches our theoretical observation that when spatial variability is ignored, energy-hungry devices that have high energy costs do not offload data samples to other devices where data can be processed at a lower cost. In this case, the newly arrived data are queued and processed locally regardless of the energy cost, resulting in low efficiency. Furthermore, if temporal variability is ignored, devices do not queue data to wait for being processed. Instead, all data samples will be processed at the time slot they arrived in the device, even though the energy cost at that time is very high. Besides, with different channel data rate between devices, the average total cost of our approach varies. We can observe that a higher average channel data rate implies a lower total cost since less

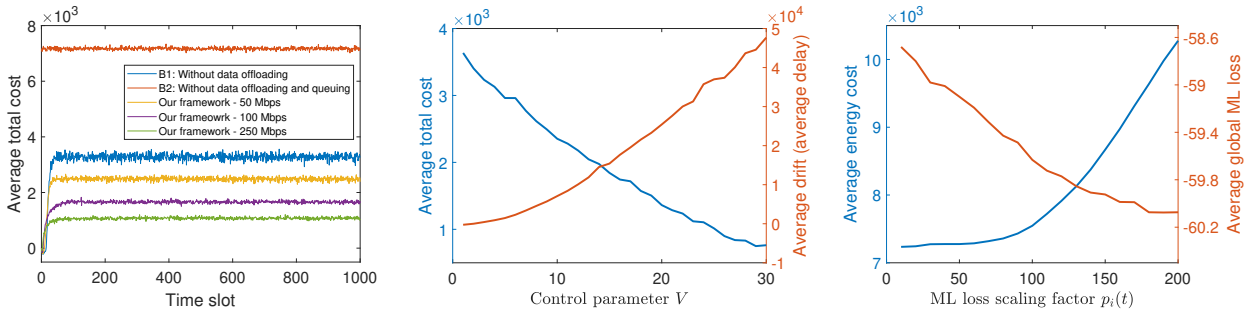

Figure 2: (a) Average total cost for our scheme compared with other baselines. (b) Trade-off between total cost and queue drift. (c) Trade-off between energy cost and training loss.

energy is needed for transmitting data samples via D2D links, which encourages energy-hungry devices to offload their data to other devices with sufficient energy. However, when the average channel data rate is low, the energy cost of the data transmission between devices is high. In this case, if devices cannot benefit much from offloading their data due to the high communication cost, they may queue the newly-arrived data locally to wait for being processed.

Then, we observe the trade-off between total cost (i.e., penalty) and queue drift and the trade-off between energy cost and training loss in our approach. Figure 2b shows the average total cost and average drift with respect to different Lyapunov control parameters. Here, we set the training loss scaling factor $p_i(t) = 100$. We can see the trade-off between the cost and drift controlled by the Lyapunov control parameter V. As V increases, the queue drift increases. In this case, fewer data samples will be processed for training at each time slot, and data samples that are not processed in the current time slot are queued to be processed in the following time slots at a lower cost, and hence, the total cost decreases. Fig 2c shows the average energy cost and training loss with respect to different training loss scaling factors $p_i(t)$. Here, we set the Lyapunov control parameter $V = 1$. We can observe the trade-off between energy cost and training loss controlled by the training loss scaling factor $p_i(t)$. With a higher scaling factor, devices tend to process more data samples to reduce the training loss, which will result in a higher cost in data processing and hence increase the total energy cost.

VI. Conclusion

In this paper, we have proposed a two-tier FEL scheme to improve the energy efficiency of edge devices. We have considered the practical situation that edge devices have heterogeneous and time-varying energy capabilities, which incur a high energy cost in FEL. By optimally designing the data offloading and queuing mechanism, the limited and heterogeneous energy of edge devices can be better utilized while maintaining the learning performance. Simulation results on the real-world dataset show that our scheme achieves the best overall performance compared with baselines without the offloading or queuing mechanism.

Acknowledgment

The work of R. Hu and Y. Gong was supported in part by the U.S. National Science Foundation under grants CNS-1850523, CNS-2047761, and CNS-2106761. The work of Y. Guo was supported in part by the US NSF under grants CNS-2029685 and CNS-2106761.

References

[1] [Online]. Available: https://blogs.idc.com/2021/01/06/future-of-industry-ecosystems-shared-data-and-insights/

[2] Y. Guo and Y. Gong, "Practical collaborative learning for crowdsensing in the internet of things with differential privacy," in *IEEE Conference on Communications and Network Security (CNS)*, 2018, pp. 1–9.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics (PMLR)*, 2017, pp. 1273–1282.

[4] R. Hu, Y. Guo, H. Li, Q. Pei, and Y. Gong, "Personalized federated learning with differential privacy," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9530–9539, 2020.

[5] A. Tak and S. Cherkaoui, "Federated edge learning: Design issues and challenges," *IEEE Network*, vol. 35, no. 2, pp. 252–258, 2020.

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

[7] ——, "Energy-efficient resource management for federated edge learning with CPU-GPU heterogeneous computing," *IEEE Transactions on Wireless Communications*, vol. 20, no. 12, pp. 7947–7962, 2021.

[8] X. Mo and J. Xu, "Energy-efficient federated edge learning with joint communication and computation design," *Journal of Communications and Information Networks*, vol. 6, no. 2, pp. 110–124, 2021.

[9] S. Wang, Y. Ruan, Y. Tu, S. Wagle, C. G. Brinton, and C. Joe-Wong, "Network-aware optimization of distributed learning for fog computing," *IEEE/ACM Transactions on Networking*, vol. 29, no. 5, pp. 2019–2032, 2021.

[10] R. Hu, Y. Guo, and Y. Gong, "Supplementary." [Online]. Available: https://drive.google.com/file/d/1JPyhi159UFlNE28McuJeKO8mAvOd7R47/view?usp=sharing

[11] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.

[12] E. Fountoulakis, N. Pappas, Q. Liao, A. Ephremides, and V. Angelakis, "Dynamic power control for packets with deadlines," in *IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.

[13] Z. Zhou, S. Yang, L. J. Pu, and S. Yu, "CEFL: Online admission control, data scheduling and accuracy tuning for cost-efficient federated learning across edge nodes," *IEEE Internet of Things Journal*, 2020.