Towards Resource-Efficient Edge AI: From Federated Learning to Semi-Supervised Model Personalization

Zhaofeng Zhang, Sheng Yue, and Junshan Zhang, *Fellow, IEEE*

***Abstract*—A central question in edge intelligence is "how can an edge device learn its local model with limited data and constrained computing capacity?" In this study, we explore the approach where a global model initialization is first obtained by running federated learning (FL) across multiple edge devices, based on which a semi-supervised algorithm is devised for a single edge device to carry out quick adaptation with its local data. Specifically, to account for device heterogeneity and resource constraints, a global model is first trained via FL, where each device conducts multiple local updates only for its customized subnet. A subset of devices can be selected to upload updates for aggregation during each training round. Further, device scheduling is optimized to minimize the training loss of FL, subject to resource constraints, based on the carefully crafted reward function defined as the one-round progress of FL each device can provide. We examine the convergence behavior of FL for the general non-convex case. For semi-supervised model personalization, we use the FL-based model initialization as a teacher network to impute soft labels on unlabeled data, thereby addressing the insufficiency of labeled data. Experiments are conducted to evaluate the performance of the proposed algorithms.**

***Index Terms*—Device heterogeneity, edge intelligence, federated learning, semi-supervised learning.**

I. Introduction

With the proliferation of mobile computing and Artificial Intelligence of Things (AIoT), billions of IoT devices are deployed at the Internet edge, generating zillions of bytes of data. That is to say, Big Data have recently gone through a radical shift of data sources from the megascale cloud data centers to the increasingly widespread end devices, e.g., mobile devices and Internet-of-Things (IoT) devices. Historically, Big Data, comprising data streams such as online shopping records, social media content, and business informatics, primarily resided within mega-scale data centers. However, the landscape is witnessing a shift in this trend owing to the widespread adoption of mobile computing and IoT technologies. Indeed, the new mobile computing ecosystem will present many novel application scenarios for AI and fuel the continuous booming of AI. Pushing the AI frontier to the edge mobile computing ecosystem at the last mile of the Internet, however, is still highly nontrivial due to the concerns on performance, cost, and privacy. Toward this goal, the conventional wisdom is to transport the data bulks from the IoT devices to the cloud data centers for analytics. Nevertheless, such data transfer across the wide area network (WAN) can lead to exorbitant monetary expenses and transmission delays, posing significant hindrances. Moreover, a critical concern exists about potential privacy breaches in this data transportation process. On-device analytics has been proposed as an alternative approach, wherein AI applications are executed directly on the IoT device, facilitating localized processing of the IoT-generated data [1], [2], [3]. However, this alternative approach also faces limitations, primarily revolving around inadequate performance and energy efficiency issues. This stems from numerous AI applications demanding substantial computational capabilities, far surpassing the capacity of resource-constrained and energy-limited IoT devices. Moreover, in many AIoT applications, a single-edge device has limited data samples only, part of which could be unlabeled, making the learning process more challenging. For instance, the newly captured images for face identification could be unlabeled or generated from a different data distribution on a mobile phone. In a nutshell, it is challenging for a single resource-constrained edge device to accomplish model training with limited data. Thus motivated, this paper seeks to answer the following important question: *"How can an edge mobile device under resource constraints carry out edge learning with limited (labeled and unlabeled) data?"*

Inspired by the tremendous success of the warm-start model training method [4], [5], we believe that the first essential step to answering the above question is to obtain a global model initialization, which can then be used for fine-tuning by using the limited data at the edge device. To better illustrate this motivation, consider the scenario wherein self-driving cars need to be aware of their surroundings and traffic to cruise through the traffic safely and arrive at their destinations. To this end, a self-driving car must learn about traffic, make predictions, and take actions in a real-time manner, which would be infeasible to use conventional methods such as cloud computing due to

Manuscript received 2 March 2023; revised 12 September 2023; accepted 12 September 2023. Date of publication 18 September 2023; date of current version 4 April 2024. This work was supported in part by the NSF under Grants CNS-2203239, CNS-2203412, RINGS-2148253, and CCSS-2203238, in part by the NSFC under Grant 62302260, and in part by the CPSF under Grant 2023M731956. Recommended for acceptance by L. Guo. *(Corresponding author: Sheng Yue.)*

Zhaofeng Zhang is with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ 85281 USA (e-mail: zzhan199@asu.edu).

Sheng Yue is with the Department of Computer Science and Technology, Tsinghua University, Beijing 100190, China, and also with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ 85281 USA (e-mail: shaun.yue@hotmail.com).

Junshan Zhang is with the College of Engineering, University of California, Davis, CA 95616 USA, and also with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ 85281 USA (e-mail: jazh@ucdavis.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TMC.2023.3316189, provided by the authors.

Digital Object Identifier 10.1109/TMC.2023.3316189

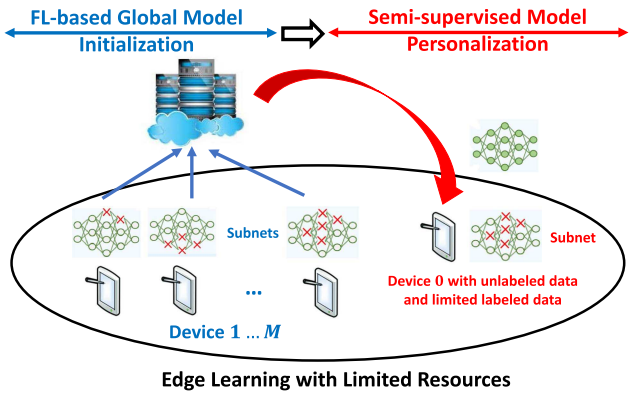


Fig. 1. Illustration of Resource-Efficient Edge Learning.

large amounts of delays in communication between the car and the cloud. Moreover, it is impractical for this self-driving car to train an AI model since general AI applications require much more computational resources and labeled training samples than a single self-driving car. Fortunately, many edge-learning tasks share similarities. For instance, in the above scenario, many self-driving cars connected via V2V and V2X communications can perform similar coordination behaviors according to environmental changes. Based on this observation, we advocate learning a model initialization by running federated learning (FL) [6] over the data samples across many edge devices to extract global structural information for knowledge transfer. Since edge devices often have limited labeled samples, inspired by knowledge distillation (KD), we propose to use the FL-based model initialization as a teacher neural network to assign soft labels on unlabeled data for the single edge device so that we can explore both limited labeled data and unlabeled data while leveraging the transferred knowledge from the global model initialization. This is akin to learning a meta-model via meta-learning [7], with the following advantage: the main objective of FL is to train a global model and is hence less computationally demanding than meta-learning which is designed to learn the model for each task.

We caution that to obtain the model initialization, running FL hinges heavily upon collaborative learning across edge devices with heterogeneous resource constraints in terms of computing capability, memory, power, limited communication bandwidth, and dataset sizes. Notably, FL under device heterogeneity and resource constraints are not well understood [8], [9], [10], [11], and this is one key challenge we will tackle in this study. In particular, some devices could experience severe resource constraints, e.g., they may overheat and stop working temporarily or experience poor wireless transmissions, which may significantly delay the parameter aggregation at the server. As a result, it is of great interest to devise a 'resource-efficient FL' that can consider resource constraints at heterogeneous devices with different computing and communication capabilities. Building on the model initialization trained via resource-efficient FL, we propose a semi-supervised learning algorithm to train a personalized model at a single edge device with limited data.

As illustrated in Fig. 1, a global backbone model initialization is first trained across many devices via FL, which is then adapted to learn a personalized model at the device with limited data. At the outset, each device is assigned a device-customized subnet, which is generated to meet the local computing constraints at individual devices via a single-shot fast neural network pruning method. We note that a 'subnet' here is a compressed model of the backbone one. During each training round, based on the latest aggregated model, each device carries out multiple local updates only for its assigned subnet, thereby improving the computation efficiency. Moreover, due to the bandwidth constraints, only a subset of devices can be selected in each round to upload their locally updated subnets and pruning profiles for aggregation at the server. Based on each device's carefully crafted reward function, we formulate the device scheduling problem to minimize the training loss, subject to resource constraints. Given the model initialization learned from FL, a semi-supervised learning algorithm is devised to train a personalized model for a new edge learning task with limited labeled/unlabeled data.

Inspired by knowledge distillation [12], [13] in deep learning, we use the global model learned from FL as a teacher network to impute soft labels on the unlabeled data so that the transferred knowledge and the intrinsic structure of unlabeled data can be leveraged simultaneously.

The main contributions of this work are summarized as follows.

- We study semi-supervised edge learning, facilitated by the model initialization via resource-efficient FL, where a global backbone model is first trained across many edge devices via FL, which is then adapted to learn a personalized model for an edge device with limited data. Notably, for model personalization, the learned model initialization from FL is used as a teacher network to generate soft labels on unlabeled data so that the transferred knowledge and the intrinsic structure of unlabeled data can be leveraged simultaneously.
- To account for device heterogeneity and resource constraints, we focus on resource-efficient FL, for which we define the reward function of each device as the resulting descent of the global objective function that the device can provide in terms of the one-round progress of FL. Inspired by the Upper Confidence Bound (UCB) algorithm for the Multi-armed Bandit (MAB) problem, we develop an adaptive design of rewards, in the sense that the server makes a more aggressive selection of devices in the initial phase of the training process, by taking a more optimistic view of the resulting descent in the training loss; and then selects devices more conservatively to mitigate the impact of uncertainty (variance) at later rounds.
- Aiming to minimize the training loss of resource-efficient FL, we formulate a device scheduling problem subject to resource constraints and develop a fixed-priority preemptive scheduling algorithm accordingly. Moreover, we characterize the performance of FL by examining its convergence behavior for the general non-convex case.
- We evaluate the performance of the proposed edge learning algorithm on various datasets and deep neural network (DNN) architectures. The experimental results clearly illustrate the improvement of the proposed algorithm over the existing baselines in terms of accuracy and efficiency,

corroborating that the proposed edge learning algorithms can successfully address the insufficiency of labeled data and device heterogeneity.

In the remainder of the paper, we introduce the problem formulation of edge learning and the designing details of the resource-efficient FL algorithm in Section II. In Section III, we design the reward function of each device and provide the convergence analysis of the proposed FL algorithm. The semi-supervised model personalization algorithm via knowledge distillation is discussed in Section IV. Extensive experimental results are presented in Section V. We provide a brief review of related work in Section VI. Finally, the conclusions and future work are discussed in Section VII.

II. Resource-Efficient Edge Learning

A. Semi-Supervised Edge Learning

We consider a semi-supervised learning setting where edge device 0 has a small labeled dataset $\mathcal{D}_l = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{|\mathcal{D}_l|}$ with total $|\mathcal{D}_l|$ samples and an unlabeled dataset $\mathcal{D}_u = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}_u|}$ with $|\mathcal{D}_u|$ samples, and $|\mathcal{D}_l| \ll |\mathcal{D}_u|$. For a model parameter $\boldsymbol{\theta} \in \mathbb{R}^N$, the empirical loss on $\mathcal{D}_l$ is defined as

$$L(\boldsymbol{\theta}, \mathcal{D}_l) \triangleq \frac{1}{|\mathcal{D}_l|} \sum_{i=1}^{|\mathcal{D}_l|} l(\boldsymbol{\theta}, (\mathbf{x}_i, \mathbf{y}_i)), \tag{1}$$

where $l : \mathbb{R}^N \to \mathbb{R}$ is the loss function for a single data sample. We introduce an additional term $R(\boldsymbol{\theta}, \boldsymbol{w}, \mathcal{D}_u)$ for simultaneously exploring the unlabeled dataset and extracting the valuable knowledge from other edge devices, aggregated in a global model $\boldsymbol{w}$ by FL, and strike a balance therein. We have the semi-supervised edge learning problem as follows:

$$\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}, \mathcal{D}_l) + \lambda R(\boldsymbol{\theta}, \boldsymbol{w}, \mathcal{D}_u), \tag{2}$$

where λ is a penalty parameter striking the trade-off between $R(\boldsymbol{\theta}, \boldsymbol{w}, \mathcal{D}_u)$ and the loss $L(\boldsymbol{\theta}, \mathcal{D}_l)$ on labeled data. The ultimate goal is to learn a personalized model for device 0 by fully using limited (labeled and unlabeled) data and the knowledge transferred from other edge devices.

B. Learning Global Model Initialization Via Resource-Efficient FL

For training the model initialization $\boldsymbol{w}$ across edge devices within the set $\mathcal{M}$, we consider the following standard FL problem:

$$\min_{\boldsymbol{w}} f(\boldsymbol{w}) \triangleq \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} f_m(\boldsymbol{w}), \tag{3}$$

where $f_m(\boldsymbol{w})$ is the expected loss of device m, defined as

$$f_m(\boldsymbol{w}) \triangleq \mathbb{E}_{\xi_m \sim \mathcal{P}_m} l(\boldsymbol{w}, \xi_m), \tag{4}$$

with $\boldsymbol{w} \in \mathbb{R}^N$ being the model parameter, ξ_m being one labeled data sample, and $\mathcal{P}_m$ being the underlying data distribution of device m. The training data samples are assumed to be non-IID across edge devices. Before the training process, a portion of model parameters, i.e., a device-specific subnet, is determined from the backbone model by each device via a single-shot neural network pruning method. At the beginning of each round, the server broadcasts the latest parameters of the backbone network to all devices. After receiving the parameters, each device only updates the weights of its corresponding subnet using its local data and then transmits the local updates to the server. Moreover, due to the limited communication bandwidth, only a subset of devices can be selected to upload their local updates and pruning profiles in each training round for updating the global model based on each device's carefully crafted reward function. Once the devices obtain the scheduling information, the local updates and pruning profiles can be uploaded from selected devices to the server, where the local updates are aggregated to obtain a new backbone network model. This finishes one round. The next round starts when the server broadcasts the new model to all devices. In the following, we detail the local updating rule at individual devices, the communication model, the device scheduling problem formulation, and the global backbone model updating rule for resource-efficient FL.

1) Computation-Efficient Local Updates of Subnets At Individual Devices: The computing capability of individual devices, which is in terms of its floating point operations per second (FLOPS), impacts the size of its device-specific subnet [14]. Specifically, essential connections are discovered based on their influence on the loss function based on a small batch of training samples. Given the desired sparsity level, redundant connections are pruned once before training (i.e., single-shot), and then the sparse pruned network is trained in the standard way. During the computing process at each device, only weights belonging to its subnet need to be updated [15]. Thus, the device with higher computing capability will employ a denser subnet.

We apply SNIP [16], a single-shot pruning method based on connection sensitivity without pre-training, to quickly obtain a sparse subnet for each device. Sizes of these device-specific subnets $\{N_m\}_{m \in \mathcal{M}}$ are set to be propositional to the devices' computational abilities so that they can be compatible with local computing hardware of individual devices. For convenience, let a $N \times |\mathcal{M}|$ matrix $\boldsymbol{I}$ denote the pruning profile over all devices, where the entry $I_{nm} = 1$ indicates that the m-th device's subnet contains the n-th weight, and $I_{nm} = 0$ otherwise. Then the network pruning problem at device m can be formulated as

$$\min_{\mathbf{I}_m, \boldsymbol{w}_m} f_m(\mathbf{I}_m \odot \boldsymbol{w}_m), \tag{5}$$

$$\text{s.t.} \quad \boldsymbol{w}_m \in \mathbb{R}^N, \mathbf{I}_m \in \{0,1\}^N, \quad \|\mathbf{I}_m\|_0 \leq N_m, \tag{6}$$

where $\odot$ denotes the Hadamard product. Instead of directly optimizing the above pruning problem, which is difficult, we follow the idea from [16] to determine the importance of each connection by measuring its effect on the loss function f_m. By relaxing the binary constraint on the indicator variables $\mathbf{I}_m$, the effect can be approximated based on a small batch of samples $\mathcal{D}_b$ by the derivation of f_m with respect to I_{nm}, which is denoted as $g_n(\boldsymbol{w}_m)$ and can be written as

$$g_n(\boldsymbol{w}_m) = \left. \frac{\partial f_m(\mathbf{I}_m \odot \boldsymbol{w}_m; \mathcal{D}_b)}{\partial I_{nm}} \right|_{\mathbf{I}_m = 1}. \tag{7}$$

This can be computed efficiently in one forward-backward pass using automatic differentiation for all n simultaneously. We then define connection sensitivity as the normalized magnitude of the

derivatives:

$$\alpha_{nm} = \frac{|g_n(\boldsymbol{w}_m)|}{\sum_{k=1}^{N} |g_k(\boldsymbol{w}_m)|}. \tag{8}$$

Once the sensitivity is computed, only the top-N_m connections are retained. Formally, the indicator variables $\mathbf{I}_m$ are set as

$$I_{nm} = \mathbb{1}\left[\alpha_{nm} - \tilde{\alpha}_m \geq 0\right], \quad \forall n \in \{1 \dots N\}, \tag{9}$$

where $\tilde{\alpha}_m$ is the N_m-th largest element in the vector $\boldsymbol{\alpha}_m$ and $\mathbb{1}$ is the indicator function. Compared to other existing pruning methods, the above one is more computationally efficient for resource-constrained edge devices since it only uses a small batch of training samples in one forward-backward.

Given the pruning profile, the m-th device will compute its local stochastic gradient of the n-th weight with $I_{nm} = 1$ and update it τ times in each training round. On the contrary, it will skip calculating the stochastic gradient of the n'-th weight if $I_{n'm} = 0$. The coordinate-wise local model updating rule is given by

$$\boldsymbol{w}_{n,m}^{t+1} = \begin{cases} \boldsymbol{w}_n^t - \eta \sum_{\alpha=0}^{\tau-1} g_n(\boldsymbol{w}_m^{t,\alpha}; \mathcal{D}_m) & \text{if } \mathrm{I}_{\mathrm{nm}} = 1, \\ \boldsymbol{w}_n^t & \text{if } \mathrm{I}_{\mathrm{nm}} = 0, \end{cases} \tag{10}$$

where $\eta > 0$ is learning rate and $g_n(\boldsymbol{w}_m^{t,\alpha}; \mathcal{D}_m)$ is the m-th device's local stochastic gradient of the n-th weight at step α within the t-th training round with $\boldsymbol{w}_m^{t,0} = \boldsymbol{w}^t$.

2) Communication Model: As alluded to earlier, due to the communication constraints, the server needs to decide the scheduler $\boldsymbol{s}^t = \{s_m^t\}_{m=1}^{|\mathcal{M}|}$ to select a subset of devices, where $s_m^t \in \{0, 1\}$. $s_m^t = 1$ indicates that the m-th device is selected to upload its model parameters and pruning profiles at round t and $s_m^t = 0$ otherwise. We assume all devices share the same wireless link to the aggregation server. Due to the interference among these wireless devices, they must share the communication bandwidth B in time or frequency to avoid mutual interference. We next investigate the performance of two fundamental communication scheduling schemes, namely time-sharing versus bandwidth-sharing-based communication scheduling, in terms of reducing communication delay. We note that "bandwidth-sharing" is the scheme where each device is allocated a fixed proportion of the total bandwidth for communication. The "time-sharing" scheme is where the total bandwidth is always fully allocated to one single device when scheduled. The following result indicates that it suffices to focus on time-sharing-based communication scheduling.

Proposition 1 (Communication scheduling scheme): For any bandwidth-sharing-based scheduler, there exists a time-sharing-based scheduler such that the number of transmission-completed devices is no less than that of bandwidth-sharing-based schedulers within the training time budget.

Proof: Due to the limited space, we outline only a few main steps for the proof. We prove Proposition 1 by induction. Given any bandwidth-sharing-based communication schedule, we first show that for any pair of two devices in the set $\mathcal{M}$, we can construct a new feasible time-sharing schedule for this pair of two devices such that the required total communication time would not increase. Next, we can use the above reasoning iteratively to find a new time-sharing schedule for all the devices in set $\mathcal{M}$. The full-version proof is provided in the Appendix, available online. □

Proposition 1 reveals that, in general, the scheduler based on time-sharing is more efficient than the one based on bandwidth-sharing because the time-sharing-based approach reduces the communication time of each device by using all the bandwidth. Thus, when the m-th device is scheduled and the up-link signal-to-noise ratio (SNR) of the device m is γ_m, the uploading latency of device m is $T_{cm,m} = \frac{N_m}{B \log_2(1+\gamma_m)}$.

3) Adaptive Device Scheduling for Model Aggregation.: Aiming to minimize the training loss, we formulate a device scheduling problem to maximize rewards across devices via selecting devices within each training round. Let r_m^t denote the reward function of the device m in round t. Given the budget of training time T_{ddl} within one round, the reward maximization problem of the t-th round is given as

$$\max_{\mathbf{s}^t} \sum_{m \in \mathcal{M}} s_m^t r_m^t, \tag{11}$$

$$\text{s.t. } s_m^t \in \{0, 1\}, \forall m \in \mathcal{M}, \tag{12}$$

$$V\left(\mathbf{s}^t\right) \leq T_{ddl}. \tag{13}$$

where $V(\mathbf{s}^t)$ is the training time of the t-th round. (12) is the feasibility condition on the device selection. (13) requires that the training time of one round does not exceed T_{ddl}. We note that obtaining an analytic formulation of $V(\mathbf{s}^t)$ is challenging since it not only depends on the selection of devices but also hinges on the scheduling order and devices' computing time. To determine the optimal scheduling policy, a naive approach is to use an exhaustive search by calculating the total rewards (that can be completed at the server based on the most recent rewards from the past training rounds before the communication starts) for all possible scheduling policies and then find the optimal one. However, the computational complexity of the exhaustive search is $O(|\mathcal{M}|!)$, which is prohibitively high [17]. Therefore, it necessitates a computationally efficient approximation scheduling algorithm for solving the problem (11). Inspired by the fixed-priority preemptive scheduling commonly used in real-time systems [18], we propose sub-optimal scheduling to achieve efficient dynamic device selection. More specifically, we assign the priorities of devices based on the descending order of the corresponding devices' reward earning rates, which are defined by $v_m^t = \frac{r_m^t}{T_{cm,m}}$. Once finishing computing, device m will preempt the ongoing communication process of another device m' if and only if $v_m^t > v_{m'}^t$. Device m' can continue its communication if $v_m^t \leq v_{m'}^t$. The pursuit here is that the device with a larger reward earning rate is always scheduled first, which is optimal for the linear relaxation of problem (11). The proposed scheduling algorithm is outlined as Algorithm 1. We note that the scheduled devices upload their rewards to the server after updating their subnets. (In other words, the scheduled devices do not need to upload their rewards simultaneously.) Hence, there is no interference in uploading reward values in Algorithm 1. Moreover, the value of the reward function is a scalar so that its transmission latency can be ignored.

Algorithm 1: Priority-Based Preemptive Scheduling Algorithm for Resource-Efficient FL.

Inputs: $\{r_m^t\}_{m=1}^{|\mathcal{M}|}$ and $\{T_{cm,m}\}_{m=1}^{|\mathcal{M}|}$
Outputs: The scheduling policy π^t
for $m = 1, \ldots, |\mathcal{M}|$ **do**
  Device m uploads its reward to the server after finishing updating its local model;
  if $v_m^t > v_{m'}^t$ holds **then**
    All the bandwidth is reallocated to device m;
  else
    Device m waits;
  end if
end for
return π^t

Algorithm 2: Learning Model Initialization via Resource-Efficient FL.

Each device generates the subnet of size N_m based on the single-shot network pruning method;
for $t = 0, 1, \ldots, T$ **do**
  The server broadcasts $\boldsymbol{w}^t$ to all the devices;
  for $m = 1, \ldots, |\mathcal{M}|$ **do**
    Device m updates its local model τ times using

$$\boldsymbol{w}_{n,m}^{t+1} = \begin{cases} \boldsymbol{w}_n^t - \eta \sum_{\alpha=0}^{\tau-1} g_n(\boldsymbol{w}_m^{t,\alpha}; \mathcal{D}_m) & \text{if } \mathrm{I_{nm}} = 1, \\ \boldsymbol{w}_n^t & \text{if } \mathrm{I_{nm}} = 0; \end{cases}$$

    Device m uploads $\boldsymbol{w}_m^{t+1}$ and $\mathbf{I}_m$ if it is scheduled by the server according to π^t;
  end for
  The server calculates the global model using

$$\boldsymbol{w}_n^{t+1} = \begin{cases} \frac{\sum_{m\in\mathcal{M}} I_{nm} s_m^t \boldsymbol{w}_{n,m}^{t+1}}{\sum_{m\in\mathcal{M}} I_{nm} s_m^t} & \text{if } \sum_{\mathrm{m}\in\mathcal{M}} \mathrm{I_{nm}} \mathrm{s_m^t} \geq 1, \\ \boldsymbol{w}_n^t & \text{if } \sum_{\mathrm{m}\in\mathcal{M}} \mathrm{I_{nm}} \mathrm{s_m^t} = 0; \end{cases}$$

end for

4) Global Updates of Backbone Network: After receiving all local updates from the selected devices, the server calculates the n-th weight (i.e., coordinate) of the global model as

$$\boldsymbol{w}_n^{t+1} = \begin{cases} \frac{\sum_{m\in\mathcal{M}} I_{nm} s_m^t \boldsymbol{w}_{n,m}^{t+1}}{\sum_{m\in\mathcal{M}} I_{nm} s_m^t} & \text{if } \sum_{\mathrm{m}\in\mathcal{M}} \mathrm{I_{nm}} \mathrm{s_m^t} \geq 1, \\ \boldsymbol{w}_n^t & \text{if } \sum_{\mathrm{m}\in\mathcal{M}} \mathrm{I_{nm}} \mathrm{s_m^t} = 0, \end{cases} \quad (14)$$

where $\sum_{m\in\mathcal{M}} I_{nm} s_m^t$ is the number of scheduled devices whose subnets contain the n-th coordinate in the t-th round. The full algorithm of learning model initialization via resource-efficient FL is summarized in Algorithm 2.

III. Reward Function Design and Convergence Analysis

In this section, we first characterize the one-round progress of FL to present the descent of the global objective function each device can provide and design the reward function of each device accordingly. We then provide the convergence analysis of the FL algorithm for general non-convex objectives. We assume that N coordinates of the global backbone model are grouped into $|\mathcal{M}|$ blocks $b_1 \cup \ldots \cup b_{|\mathcal{M}|}$, where these blocks can be overlapping in terms of coordinates. For the m-th block, it contains N_m coordinates, i.e., $|b_m| = N_m$. Before the discussion, We make the following standard assumptions.

Assumption 1 (Coordinate-wise Lipschitz gradient continuity): The global objective function f and the local ones $\{f_m\}_{m=1}^M$ are coordinate-wise L_n-smooth, i.e., for each coordinate n, it has

$$\|\nabla_n f(\boldsymbol{x}) - \nabla_n f(\boldsymbol{y})\| \leq L_n \|\boldsymbol{x}_n - \boldsymbol{y}_n\|,$$
$$\|\nabla_n f_m(\boldsymbol{x}) - \nabla_n f_m(\boldsymbol{y})\| \leq L_n \|\boldsymbol{x}_n - \boldsymbol{y}_n\|, \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d,$$

where $\forall m \in \mathcal{M}$ and $L_n \leq L_{\max}$ for $n = 0, 1, \ldots, N$.

Assumption 2 (Bounded gradient): For any coordinate $n \in \{1, \ldots, N\}$, the gradient is bounded by a non-negative constant A, i.e.,

$$\left\|g_n\left(\boldsymbol{w}_m^{t,\alpha}; \mathcal{D}_m\right)\right\| \leq A, \left\|g_n\left(\boldsymbol{w}_m^{t,\alpha}; \xi_m\right)\right\| \leq A, \left\|\nabla_n f\left(\boldsymbol{w}^t\right)\right\| \leq A,$$

where $g_n(\boldsymbol{w}_m^{t,\alpha}; \xi_m)$ is the n-th coordinate's gradient based on one training sample ξ_m.

Assumption 3 (Bounded variance): For any coordinate $n \in \{1, \ldots, N\}$, the variance of its gradient is bounded above, i.e.,

$$\mathbb{E}\left[\left(g_n(\boldsymbol{w}_m^{t,\alpha}; \xi_m) - \mathbb{E}\left[g_n(\boldsymbol{w}_m^{t,\alpha}; \xi_m)\right]\right)^2\right] \leq \sigma^2,$$
$$\text{for n} = 1, \ldots, \text{N}, \forall \text{m}.$$

Assumption 4 (Polyak-Lojasiewicz inequality [19]): The global objective function f satisfies the Polyak-Lojasiewicz (PL) inequality, i.e., for all $\boldsymbol{x}$ we have for some $\mu > 0$ that

$$\frac{1}{2}(||\nabla f(\boldsymbol{x})||_*)^2 \geq \mu[f(\boldsymbol{x}) - f^*],$$

where $||\cdot||_*$ can be any norm and f^* is the optimal function value.

Assumption 5 (Device task similarity [20], [21], [22]): There exists a positive constant $\epsilon_m > 0$ such that for any $m \in \mathcal{M}$, the following holds:

$$\|\nabla_n f(\boldsymbol{w}) - \nabla_n f_m(\boldsymbol{w})\| \leq \epsilon_m, \text{ for } n = 1, \ldots, N.$$

Assumption 1 is critical for analyzing the one-step progress in coordinate descent studies [23], [24]. Assumptions 2–3 provide upper bounds on the norm and variance of the gradient, which are standard and 'match' the gradient clipping method in practical implementations [25]. Assumption 4 is a sufficient condition for gradient descent to achieve a linear convergence rate, which is weaker than the strongly convex condition [26]. Assumption 5 indicates that the variation of the gradients between the global function $f(\boldsymbol{w})$ and the local one $f_m(\boldsymbol{w})$ is bounded by some constant, which captures the similarity of the tasks corresponding to non-IID data across devices [20], [22], [27]. Based on Assumption 1, we have the following lemma, which is used for characterizing the one-round progress of the proposed

FL algorithm. The proof of Lemma 1 is omitted due to space limitations.

Lemma 1: If the global objective function f is coordinate-wise L_n-smooth, we have that

$$f\left(\boldsymbol{w}^{t+1}\right) \leq f\left(\boldsymbol{w}^{t}\right)+\nabla f\left(\boldsymbol{w}^{t}\right)^{T}\left(\boldsymbol{w}^{t+1}-\boldsymbol{w}^{t}\right) + \frac{1}{2}\sum_{n=1}^{N} L_n|\boldsymbol{w}_n^{t+1}-\boldsymbol{w}_n^{t}|^2. \tag{15}$$

A. Reward Function Design

We have the following result on the one-round progress of the proposed resource-efficient FL algorithm.

Proposition 2 (One-round progress of resource-efficient FL): Suppose that Assumption 1, 2, and 5 hold. Then, for $\eta \leq \frac{1}{L_{\max}}$, we have that

$$\begin{aligned} & f\left(\boldsymbol{w}^{t}\right)-f\left(\boldsymbol{w}^{t+1}\right) \\ & \geq \frac{\eta}{2}\left\{\sum_{m\in\mathcal{M}} s_m^t\left[2\left\langle\nabla_{b_m} f_m(\boldsymbol{w}^t), \sum_{\alpha=0}^{\tau-1} g_{b_m}(\boldsymbol{w}_m^{t,\alpha};\mathcal{D}_m)\right\rangle\right.\right. \\ & \quad \left.\left.-\left\|\sum_{\alpha=0}^{\tau-1} g_{b_m}(\boldsymbol{w}_m^{t,\alpha};\mathcal{D}_m)\right\|^2\right] \right. \\ & \quad \left. -2\sum_{\alpha=0}^{\tau-1}\sum_{n=1}^{N}\left[\sum_{m\in\mathcal{M}} I_{nm}s_m^t\left(1+\frac{\epsilon_m}{A}\right)-1\right]A^2\right\}. \end{aligned} \tag{16}$$

Proof: Due to the limited space, we outline a few main steps for the proof. We first define $\hat{g}_n^t = \frac{\sum_{m\in\mathcal{M}} I_{nm}s_m^t \sum_{\alpha=0}^{\tau-1} g_n(\boldsymbol{w}_m^{t,\alpha};\mathcal{D}_m)}{\sum_{m\in\mathcal{M}} I_{nm}s_m^t}$. Recall that $\boldsymbol{w}_n^{t+1}-\boldsymbol{w}_n^t = \hat{g}_n^t$ when $\sum_{m\in\mathcal{M}} I_{nm}s_m^t \geq 0$. According to Lemma 1, it can be easily seen that

$$\begin{aligned} f\left(\boldsymbol{w}^{t+1}\right) \leq f\left(\boldsymbol{w}^{t}\right) & -\frac{\eta}{2}\sum_{n=1}^{N} \\ & \times\left[2\nabla_n f\left(\boldsymbol{w}^t\right)\hat{g}_n^t-\eta L_{\max}\left\|\hat{g}_n^t\right\|^2\right]. \end{aligned}$$

For the n-th coordinate, we define its contribution to the global objective descent as $(DE)_n = \frac{\eta}{2}[2\nabla_n f(\boldsymbol{w}^t)\hat{g}_n^t - \eta L_{\max}\|\hat{g}_n^t\|^2]$. Using the coordinate updating rule in the proposed algorithm and the Cauchy-Schwarz inequality, we can establish a lower bound on $(DE)_n$ as

$$\begin{aligned} (DE)_n \geq \frac{\eta}{2}\Bigg[& 2\nabla_n f\left(\boldsymbol{w}^t\right)\sum_{m\in\mathcal{M}} I_{nm}s_m^t\sum_{\alpha=0}^{\tau-1} g_n\left(\boldsymbol{w}_m^{t,\alpha};\mathcal{D}_m\right) \\ & -\eta L_{\max}\sum_{m\in\mathcal{M}} I_{nm}s_m^t\left(\sum_{\alpha=0}^{\tau-1} g_n\left(\boldsymbol{w}_m^{t,\alpha};\mathcal{D}_m\right)\right)^2 \\ & -2\tau\left(\sum_{m\in\mathcal{M}} I_{nm}s_m^t-1\right)A^2\Bigg]. \end{aligned} \tag{17}$$

Then, (16) can be obtained after some further algebraic manipulation. The full-version proof is provided in the Appendix, available online. □

Based on the above result, it is clear that the one-round progress is stochastic. We note that the last term at the right-hand side of (16) does not influence the one-round descent of the global objective function if the number of overlapping coordinates over blocks is small, which is often the case in practice. Thus, we focus on maximizing the first term at the right-hand side of (16) to design the reward function for each device. For convenience, define

$$\begin{aligned} LB = 2 & \left\langle\nabla_{b_m} f(\boldsymbol{w}^t), \sum_{\alpha=0}^{\tau-1} g_{b_m}(\boldsymbol{w}_m^{t,\alpha};\mathcal{D}_m)\right\rangle \\ & -\left\|\sum_{\alpha=0}^{\tau-1} g_{b_m}(\boldsymbol{w}_m^{t,\alpha};\mathcal{D}_m)\right\|^2, \end{aligned} \tag{18}$$

and we need to quantify LB while $\nabla_{b_m} f(\boldsymbol{w}^t)$ is never accessible. One naive approach is directly estimating $\nabla_{b_m} f(\boldsymbol{w}^t)$ as the stochastic gradient $g_{b_m}^t(\boldsymbol{w}^t;\mathcal{D}_m)$, which would not work well only based on limited data. Here, we derive a confidence interval of LB to provide insights into each device's adaptive reward function design. We define

$$\begin{aligned} U_m = 2 & \left\langle g_{b_m}^t(\xi_m), \sum_{\alpha=0}^{\tau-1} g_{b_m}\left(\boldsymbol{w}_m^{t,\alpha};\mathcal{D}_m\right)\right\rangle \\ & -\left\|\sum_{\alpha=0}^{\tau-1} g_{b_m}\left(\boldsymbol{w}_m^{t,\alpha};\mathcal{D}_m\right)\right\|^2. \end{aligned} \tag{19}$$

That $LB = \mathbb{E}(U_m)$ is easy to obtain. We further define

$$\begin{aligned} \overline{U_m} = 2 & \left\langle g_{b_m}(\boldsymbol{w}_m^t;\mathcal{D}_m), \sum_{\alpha=0}^{\tau-1} g_{b_m}\left(\boldsymbol{w}_m^{t,\alpha};\mathcal{D}_m\right)\right\rangle \\ & -\left\|\sum_{\alpha=0}^{\tau-1} g_{b_m}\left(\boldsymbol{w}_m^{t,\alpha};\mathcal{D}_m\right)\right\|^2 \end{aligned} \tag{20}$$

According to Assumption 2, we have $-(2\tau+\tau^2)N_m A^2 \leq U_m \leq 2\tau N_m A^2$. Given the confidence level $1-q$, we can obtain the following inequality by directly utilizing Hoeffding's inequality:

$$\begin{aligned} q &= P\left(|\overline{U_m}-LB| \geq \kappa\right) \\ &\leq 2\exp\left(-\frac{2|\mathcal{D}_m|^2\kappa^2}{|\mathcal{D}_m|\left[(\tau^2+4\tau)N_m A^2\right]^2}\right). \end{aligned} \tag{21}$$

Solving the above inequality for LB gives us the following confidence interval of LB:

$$\begin{aligned} & \overline{U}_m - N_m A^2\sqrt{\frac{-(\tau^2+4\tau)\log(\frac{q}{2})}{2|\mathcal{D}_m|}} \leq LB \\ & \leq \overline{U}_m + N_m A^2\sqrt{\frac{-(\tau^2+4\tau)\log(\frac{q}{2})}{2|\mathcal{D}_m|}}, \end{aligned} \tag{22}$$

Moreover, we have the following results on the training error.

Proposition 3 (Characterizing training error of resource-efficient FL): Suppose that Assumption 1 and 3–5 hold, and

$C = \mu(\tau - 1 + \delta)$. Then, for

$$\frac{L_{\max}^2\eta^2(\tau+1)(\tau-2)}{2} + L_{\max}\eta\tau \leq 1 \text{ and } \mathrm{L}_{\max}^2\eta^2 \leq 1-\delta$$

with some constant $0 < \delta < 1$, we have that

$$\begin{aligned}\mathbb{E}_t\left[f\left(\boldsymbol{w}^{t+1}\right)\right] - f^* \leq (1-C\eta)^t\left[f\left(\boldsymbol{w}^0\right) - f^*\right] \\ + \frac{\eta\tau}{2}\sum_{t'=0}^{t}\Bigg\{(1-C\eta)^{t-t'}\sum_{m\in\mathcal{M}} s_m^t\Big[N_m\epsilon_m^2 \\ + \frac{N_m\sigma^2}{|\mathcal{D}_m|}\left(\tau + \frac{(2\tau-1)(\tau-1)}{6}\right)\Big]\Bigg\}. \end{aligned} \tag{23}$$

Proof: We outline a few main steps for the proof due to the limited space. Here, we slightly abuse the notation '$\mathbb{E}$'. In the following, '$\mathbb{E}$' always means taking the overall expectation. By taking the summation (we ignore the effects of the overlapping coordinates over blocks) and then taking the overall expectation of (17), we have

$$\begin{aligned}&\mathbb{E}f\left(\boldsymbol{w}^{t+1}\right) - f\left(\boldsymbol{w}^t\right) \\ &\leq \sum_{m\in\mathcal{M}} s_m^t\left[-\eta\sum_{\alpha=0}^{\tau-1}\mathbb{E}\left\langle\nabla_{b_m}f(\boldsymbol{w}^t), \nabla_{b_m}f_m(\boldsymbol{w}_m^{t,\alpha})\right\rangle\right. \\ &\quad\left.+\frac{L_{\max}\eta^2}{2}\mathbb{E}\left\|\sum_{\alpha=0}^{\tau-1} g_{b_m}(\boldsymbol{w}_m^{t,\alpha};\mathcal{D}_m)\right\|^2\right]. \end{aligned} \tag{24}$$

Using the properties of the iterates in the proposed algorithm and the smoothness of f and $\{f_m\}_{m=1}^M$, we can further obtain an upper bound on $\mathbb{E}f(\boldsymbol{w}^{t+1}) - f(\boldsymbol{w}^t)$ in terms of f^*, σ^2, and ϵ_m^2 as

$$\begin{aligned}&\mathbb{E}f\left(\boldsymbol{w}^{t+1}\right) - f\left(\boldsymbol{w}^t\right) \\ &\leq -\frac{(\tau-1+\delta)\eta}{2}\sum_{m\in\mathcal{M}} s_m^t\left\|\nabla_{b_m}f(\boldsymbol{w}^t)\right\|^2 \\ &\quad+\frac{\eta\tau}{2}\sum_{m\in\mathcal{M}} s_m^t\left[\frac{N_m\sigma^2}{|\mathcal{D}_m|}\left(\tau+\frac{(2\tau-1)(\tau-1)}{6}\right)+N_m\epsilon_m^2\right] \\ &\leq -(\tau-1+\delta)\eta\mu\left[f\left(\boldsymbol{w}^t\right)-f^*\right] \\ &\quad+\frac{\eta\tau}{2}\sum_{m\in\mathcal{M}} s_m^t\left[\frac{N_m\sigma^2}{|\mathcal{D}_m|}\left(\tau+\frac{(2\tau-1)(\tau-1)}{6}\right)+N_m\epsilon_m^2\right]. \end{aligned} \tag{25}$$

Subtracting f^* on both sides of the above inequality and using the properties of the iterates in the proposed algorithm completes the proof. The full-version proof is provided in the Appendix, available online. □

Remark 1: The second term at the right-hand side of (23) captures the impact of variance $\frac{N_m}{|\mathcal{D}_m|}\sigma^2$ on the training performance. We note that the weight $(1-C\eta)^{t-t'}$ increases with t' increasing as $1 - C\eta < 1$. In other words, the variance in a later round has a larger impact on the error (the second term at the right-hand side of (23)) than in an earlier round. The observation has an important implication: the devices with larger subnet size should be selected since the objective function needs to descend quicker in the initial phase of the training process; the devices with more data should be selected to mitigate the effects of the variance at later rounds.

Inspired by the remark above, we propose the following adaptive reward function design.

$$r_m^t = \overline{U}_m + N_m A^2\sqrt{\frac{-(\tau^2+4\tau)\log(\frac{q}{2})}{2|\mathcal{D}_m|}}\left(1-\frac{2t}{t_{\max}}\right) \tag{26}$$

where $t_{\max}$ is the final training round index. It is obvious that the reward function value is close to the upper bound of LB when t is small so that the server makes a more aggressive selection of devices, using a more optimistic descent of the global objective function at the beginning of the training; the reward function value is closed to the lower bound of LB when t is large so that the server has a more conservative selection of devices to reduce the variance as much as possible.

B. Convergence Analysis of FL

Based on Assumption 1 and 3–5, we can have the following result about the convergence of the resource-efficient FL algorithm for the general non-convex case.

Theorem 1 (Convergence of resource-efficient FL): Suppose that Assumption 1 and 3–5 hold, and the learning rate η satisfying

$$\frac{L_{\max}^2\eta^2(\tau+1)(\tau-2)}{2} + L_{\max}\eta\tau \leq 1 \text{ and } \mathrm{L}_{\max}^2\eta^2 \leq 1-\delta$$

with some constant $0 < \delta < 1$. Then, the expected average squared gradient norms of scheduled devices satisfy the following bound:

$$\begin{aligned}&\frac{1}{T+1}\mathbb{E}\left[\sum_{t=0}^{T}\sum_{m\in\mathcal{M}} s_m^t\left\|\nabla_{b_m}f\left(\boldsymbol{w}^t\right)\right\|^2\right] \\ &\leq \frac{2\left(f\left(\boldsymbol{w}^0\right)-f^*\right)}{(T+1)(\tau-1+\delta)\eta} \\ &\quad+\frac{\tau}{(T+1)(\tau-1+\delta)}\sum_{t=0}^{T}\sum_{m\in\mathcal{M}} s_m^t N_m\left[\frac{\sigma^2}{|\mathcal{D}_m|}(\tau\right. \\ &\quad\left.+\frac{(2\tau-1)(\tau-1)}{6}\right)+\epsilon_m^2\Bigg]. \end{aligned}$$

Proof: By taking the summation of (25) and noting that $f^* - f(\boldsymbol{w}^0) \leq f(\boldsymbol{w}^T) - f(\boldsymbol{w}^0)$, we obtain

$$\begin{aligned}&\mathbb{E}\sum_{t=0}^{T}\sum_{m\in\mathcal{M}} s_m^t\left\|\nabla_{b_m}f\left(\boldsymbol{w}^t\right)\right\|^2 \\ &\leq \frac{2\left(f\left(\boldsymbol{w}^0\right)-f^*\right)}{(\tau-1+\delta)\eta} + \frac{\tau}{\tau-1+\delta}\sum_{t=0}^{T}\sum_{m\in\mathcal{M}} s_m^t N_m \\ &\quad\times\left[\frac{\sigma^2}{|\mathcal{D}_m|}\left(\tau+\frac{(2\tau-1)(\tau-1)}{6}\right)+\epsilon_m^2\right]. \end{aligned}$$

Dividing the above inequality by $T+1$ completes the proof. □

Remark 2: Theorem 1 shows that scheduled gradients' expected average squared norms converge to a nonzero constant

as $T \to \infty$. The first term $\frac{2(f(\boldsymbol{w}^0)-f^*)}{(T+1)(\tau-1+\delta)\eta}$ represents the gap from the initial model to the optimal solution and eventually goes to zero as the number of rounds goes to infinity. The second term is scaled by the dataset sizes and local computing steps, which indicates that larger dataset sizes and more frequent communication lead to better performance.

IV. Semi-Supervised Model Personalization Via Knowledge Distillation

Given the model initialization $\boldsymbol{w}$ obtained via FL, device 0 aims to learn a new model $\boldsymbol{\theta}$ based on its scarce labeled dataset $\mathcal{D}_l$ and unlabeled dataset $\mathcal{D}_u$. Inspired by the recent successes of knowledge distillation (KD), we propose a computing-efficient semi-supervised learning algorithm to explore both limited labeled data and unlabeled data while leveraging the transferred knowledge from the global model initialization. Similar to FL, a sparse subnet with size $N_{\boldsymbol{\theta}}$ is first quickly obtained by the single-shot network pruning method so that the subnet can be compatible with the computing hardware of device 0. We then use the global model $\boldsymbol{w}$ as a teacher network to assign soft labels on unlabeled data. The original semi-supervised learning problem (2) can be recast as the following minimization one:

$$\min_{\boldsymbol{\theta}} -\frac{1}{|\mathcal{D}_l|}\sum_{(\boldsymbol{x}_i,y_i)\in\mathcal{D}_l}\log p(y_i|\boldsymbol{x}_i;\boldsymbol{\theta},1)) - \frac{\lambda}{|\mathcal{D}_u|}\sum_{\boldsymbol{x}_i\in\mathcal{D}_u}\sum_{y} p(y|\boldsymbol{x}_i;\boldsymbol{w},T_{KD})\log p(y|\boldsymbol{x}_i;\boldsymbol{\theta},T_{KD}) \tag{27}$$

$$\text{s.t. } \|\boldsymbol{\theta}\|_0 = N_{\boldsymbol{\theta}}, \tag{28}$$

where $p(y|\boldsymbol{x}_i;\boldsymbol{w},T_{KD}) = \frac{e^{z(\boldsymbol{x}_i,y,\boldsymbol{w})/T_{KD}}}{\sum_{y'} e^{z(\boldsymbol{x}_i,y',\boldsymbol{w})/T_{KD}}}$, $z(\cdot)$ is the logit value, and T_{KD} is the distillation temperature. Notably, in (28), the new device only needs to update a subnet of the local model, hence significantly reducing the computational cost. Though we focus on using labeled and unlabeled data in this work, one can only use the second term (the distillation loss) in (27) for training θ when the number of labeled samples is extremely small. We also note that the proposed algorithm can be implemented on the device either with the same model architecture as $\boldsymbol{w}$ or a more compact one. Algorithm 3 outlines the main steps of learning θ via semi-supervised model personalization. As the theoretical foundation for KD remains open in deep learning, providing theoretical performance guarantees for Algorithm 3 is highly nontrivial. However, the results using real-world benchmarks demonstrate that it can indeed improve over the existing baselines in terms of accuracy and efficiency.

V. Experiments

In this section, we study the image classification problem and evaluate the performance of the proposed resource-efficient edge learning framework.

Algorithm 3: Learning θ via Semi-Supervised Model Personalization.

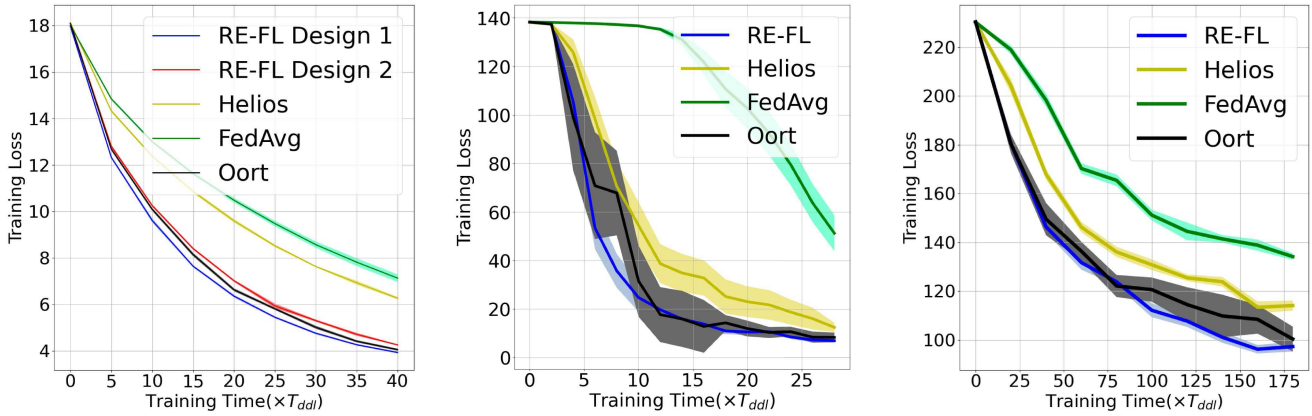
Inputs: $\mathcal{D}_l$, $\mathcal{D}_u$, λ, T_{KD}, N_θ and $\boldsymbol{w}$
Outputs: The personalized model θ
1: Sample a mini-batch $\mathcal{D}_b$ from $\mathcal{D}_l$;
2: Use $\mathcal{D}^b$ to generate the subnet of size N_θ based on the single-shot network pruning method;
3: Use $\boldsymbol{w}$ to generate soft labels on $\mathcal{D}_u$;
4: Train the subnet based on the loss $\mathcal{L}$;
5: **return** θ

A. Datasets, Models and Evaluation

We examine the performance of the proposed edge learning algorithms on various datasets, including Caltech 101 [28], MNIST [29], and CIFAR-10 [30], where the data for training $\boldsymbol{w}$ via FL are distributed among 50 devices. We use two CNN architectures, including Lenet and VGG-16, the choice of which varies among datasets due to the differences in image sizes and feature diversities. We first compare the performance of the proposed algorithms training FL-based initialization with three baselines, namely FedAvg [6] algorithm, Helios [10], and Oort [31]. Helios is the FL baseline considering devices' computational heterogeneity but not communication constraints. Oort is an FL scheduling framework that considers device heterogeneity but without analyzing convergence. We then compare the proposed semi-supervised model personalization algorithm with the following approaches: Two global models are trained respectively by FedAvg and Helios and used as the teacher networks for obtaining personalized models; Three personalized FL approaches KT-pFL [32], pFedHN [33], and FedPCL [34] that realize personalized model training in a federated manner. In all experiments, we set bandwidth $B = 0.5$ MHz. Each device has samples from only two random classes. We use the cross-entropy loss function to calculate the training loss. All experiments are repeated ten times to obtain the average performance, and the comparison is shown with 95% confidence intervals. The experiments are implemented using Python 3.6 and PyTorch 1.8.1 with CUDA 11 on a server with one Intel Core i9-10900K CPU and one Nvidia GeForce RTX 3090 24 G GPU. Details on the settings of datasets, wireless channel conditions, computation capabilities, and CNN architectures are relegated to the Appendix, available online.

B. FL-Based Model Initialization

We first examine the performance of our proposed algorithm in the limited data regime, where less than 15% of training data of the Caltech 101 dataset are distributed to devices. We assume the server first broadcasts a VGG-16 model, which contains three fully connected and 13 convolution layers and is trained on the Imagenet dataset [35]. Once all devices receive the model, they only prune and retrain the last two layers instead of the entire VGG-16 model. For convenience, we refer to the reward design based on (26) as 'RE-FL design 1' and the reward design based on directly estimating $\nabla_{b_m} f(\boldsymbol{w}^t)$ as the stochastic gradient $g^t_{b_m}(\boldsymbol{w}^t;\mathcal{D}_m)$ as 'RE-FL design 2'. From Fig. 2(a) and



(a) Training Loss over Caltech 101: Limited Data Regime.

(b) Training Loss over MNIST: Abundant Data Regime.

(c) Training Loss over CIFAR-10: Abundant Data Regime.

Fig. 2. Training Performance of FL-based Model Initialization.

TABLE I
TESTING ACCURACY COMPARISON OF FL-BASED MODEL INITIALIZATION

Dataset	RE-FL	Helios	FedAvg	Oort
Caltech 101	**86.33%±0.48% (design 1)** **85.48%±0.62% (design 2)**	76.28% ±0.43%	69.18% ±3.34%	86.08% ±0.53%
MNIST	**96.26% ±0.67%**	93.55% ±1.85%	72.82% ±2.21%	95.83% ±0.45%
CIFAR-10	**61.14% ±0.94%**	58.24% ±0.99%	51.84% ±2.52%	60.03% ±0.95%

TABLE II
TESTING ACCURACY COMPARISON OF SEMI-SUPERVISED MODEL PERSONALIZATION

Dataset	RE-SSMP	Helios-based EL	FedAvg-based EL	PKT-based EL	PFL-based EL	CL-based EL
Caltech 101	**72.78%±2.07%**	67.09%±1.15%	65.74%±2.24%	68.14%±1.25%	65.86%±2.07%	70.10%±1.89%
MNIST	**88.38%±1.57%**	87.74%±2.87%	79.58%±3.13%	86.13%±0.60%	85.26%±2.34%	88.28%±1.11%
CIFAR-10	**63.67%±1.93%**	57.21%±1.07%	55.13%±5.04%	57.92%±2.45%	58.58%±1.74%	59.40%±2.01%

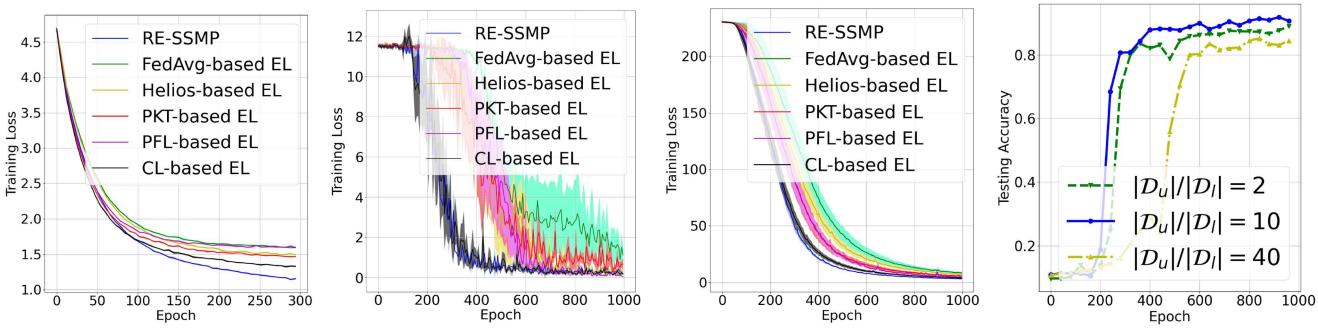
Table I, we can observe that our proposed algorithm outperforms the baselines regarding the accuracy and the convergence speed for different device settings. Note that the design 1 of reward function consistently outperforms the design 2 in the limited data regime thanks to its adaptive characteristics. We then evaluate the performance of our proposed algorithm on training a CNN model in the abundant data regime. Specifically, we explore two different datasets, MNIST and CIFAR-10, to train a LeNet model, which contains three fully connected layers and two convolution layers; the whole training dataset is distributed to devices so that each device owns abundant training samples. As illustrated in Fig. 2(b), (c) and Table I, our proposed FL algorithm outperforms the counterparts in terms of convergence speed and testing accuracy for different settings.

C. Semi-Supervised Model Personalization

We evaluate the proposed semi-supervised edge learning algorithm with limited labeled data and the FL-based model initialization. Specifically, we conduct this experiment on three different datasets, where $|\mathcal{D}_l| = 100$, $|\mathcal{D}_u| = 900$ for Caltech 101, $|\mathcal{D}_l| = 50$, $|\mathcal{D}_u| = 500$ for MNIST and $|\mathcal{D}_l| = 100$, $|\mathcal{D}_u| = 800$ for CIFAR-10. For the sake of fairness, we compare it with the following approaches: Two global models are trained respectively by FedAvg and Helios and used as the teacher networks for obtaining personalized models. We refer to the proposed semi-supervised model personalization algorithm as 'RE-SSMP' for convenience. It can be seen from Table II and Fig. 3(a)–(c) that the proposed algorithm clearly outperforms the counterparts' convergence speed and testing accuracy. It corroborates that the proposed algorithm can successfully address the insufficiency of labeled data and deliver model personalization at the network edge. To quantify the impact of knowledge transfer on the performance, we train the model for device 0 with the same global initialization but different $\frac{|\mathcal{D}_u|}{|\mathcal{D}_l|}$, in the sense that device 0 relies more on the transferred knowledge when $\frac{|\mathcal{D}_u|}{|\mathcal{D}_l|}$ is larger. As shown in Fig. 3(d), the performance improves as $\frac{|\mathcal{D}_u|}{|\mathcal{D}_l|}$ increases from 2 to 10. The reason is that more information is provided by the model initialization as more unlabeled samples are assigned with soft labels. The performance drops as $\frac{|\mathcal{D}_u|}{|\mathcal{D}_l|}$ increases further. In other words, the transferred knowledge will in turn hurt the learning performance due to the sample distribution shift between device 0 and devices in $\mathcal{M}$.

VI. RELATED WORK

Resource-Efficient FL: [10] recently proposed to dynamically mask different sets of neurons for different devices depending on their computing resources profiles, and these neurons are



(a) Training Loss over Caltech 101. (b) Training Loss over MNIST. (c) Training Loss over CIFAR-10. (d) Impact of $\frac{|\mathcal{D}_u|}{|\mathcal{D}_l|}$.

Fig. 3. Performance of Semi-supervised Model Personalization.

recovered and updated during the aggregation. In the work proposed by [11], each device is assigned a DNN with a topology that fits its computing capability, where part of the DNNs is shared and jointly learned. We note that the above works and their related algorithms [36], [37], [38] propose FL algorithms where the workloads fit the computing capabilities at individual devices. Still, it is not guaranteed that each device can work synchronously due to the unpredictable behaviors of stragglers in practice. Moreover, they do not consider the communication overhead, which can become a significant bottleneck, especially for training a large-scale DNN. Along a different line, some recent works develop device scheduling algorithms under limited bandwidth but do not consider devices' computing abilities. Specifically, [39] aims to optimize the learning performance, which depends on how devices are selected in each round from a long-term perspective. [40] develop a probabilistic scheduling framework based on the gradient divergence at the devices. A recent FL work proposed by [31] prioritizes the use of those devices that have both data with greater utility in improving model accuracy and the capability to run training quickly, and [31] did not provide convergence analysis. Personalized FL focuses on realizing personalized model training. Notably, [32] formulates the aggregation procedure in the original personalized FL into a personalized group knowledge transfer training algorithm, which enables each client to maintain a personalized soft prediction on the server side to guide the others' local training. In [33], a central hypernetwork model is trained to generate a set of models for each client, which provides effective parameter sharing across clients while maintaining the capacity to generate unique and diverse personal models. [34] proposes a personalized FL approach that shares knowledge across clients through their class prototypes and builds client-specific representations in a prototype-wise contrastive manner. Nevertheless, the above approaches do not simultaneously consider computing capabilities and communication constraints among heterogeneous edge devices. To our knowledge, our work is the first attempt to develop a resource-efficient edge learning framework that accounts for device heterogeneity and communication constraints, where a semi-supervised algorithm enables edge learning with limited labeled data based on a global model initialization by running FL across multiple edge devices.

Semi-supervised Learning for DNNs: Semi-supervised learning [41] is a powerful tool for exploring unlabeled data with DNNs. Recent works indicate that semi-supervised learning is theoretically [42] and empirically [43] effective on unlabeled data. These methods either require fitting the unlabeled data on its outputs generated by a self-trained model with limited labeled data [13], [44] or stability of DNN outputs under various data augmentations [43], [45] (also known as consistency regularization). However, without a deeply pre-trained model to provide an unbiased regularization, semi-supervised learning from scratch could be easily misled by inaccurate pseudo-labels, especially when labeled data is extremely insufficient [46].

Neural Network Pruning: Recently, the neural network pruning method has emerged as a powerful tool to significantly increase the computing efficiency for large-scale neural network training. The rationale behind the network pruning is that neural networks are usually overparameterized, and comparable performance can be obtained by a much smaller one obtained with careful pruning. Notably, [47] is one of the pioneering works on network pruning, and sparsity enforcing penalty terms [48] and saliency criteria [49] (e.g., weight sensitivity) are widely used in recent studies on network pruning. Recent work SNIP [16] proposes a single-shot pruning method based on connection sensitivity without pretraining to quickly obtain the optimized sparse subnets.

VII. Conclusion and Future Work

In this study, we advocate first obtaining a global model initialization by running FL across multiple edge devices, based on which a semi-supervised algorithm is developed to enable edge learning with limited data. Specifically, to account for device heterogeneity and resource constraints, a global backbone model is first trained via FL, where each device carries out multiple local updates only for its customized subnet and only a fraction of devices can be selected to upload locally updated subnets and pruning profiles for aggregation during each training round. Moreover, device scheduling is optimized to minimize the training loss of FL, subject to resource constraints, based on the carefully crafted reward function. We provide a rigorous convergence analysis of FL for the general non-convex

case. For semi-supervised model personalization, we use the FL-based initial model as a teacher network to assign soft labels on unlabeled data, aiming to address the insufficiency of labeled data and to learn a personalized model efficiently. Experiments are conducted to evaluate the performance of the proposed algorithm.

There are several interesting questions and directions for future work. First, it is interesting to adopt the proposed method to a more practical setting where the target edge device only contains unlabeled data samples (i.e., the unsupervised case). Second, although FL can be directly applied to reinforcement learning with policy gradient [50], it may lead to poor sample efficiency. It remains largely open to developing resource-efficient FL-based reinforcement learning over edge devices. Another possible extension is to consider multi-task learning over a fully decentralized edge network based on the same rationale provided by the proposed algorithm, and it is expected that a resource-efficient fully decentralized learning algorithm will significantly decrease the computing and communication cost over the edge network.

REFERENCES

[1] Z. Zhang, Y. Chen, and J. Zhang, "Distributionally robust learning based on Dirichlet process prior in edge networks," *J. Commun. Inf. Netw.*, vol. 5, no. 1, pp. 26–39, 2020.

[2] Z. Zhang, S. Lin, M. Dedeoglu, K. Ding, and J. Zhang, "Data-driven distributionally robust optimization for edge intelligence," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 2619–2628.

[3] Z. Zhang, Y. Chen, and J. Zhang, "Distributionally robust edge learning with Dirichlet process prior," in *Proc. 40th Int. Conf. Distrib. Comput. Syst.*, 2020, pp. 798–808.

[4] H. Wang, S. Lin, and J. Zhang, "Warm-start actor-critic: From approximation error to sub-optimality gap," 2023, *arXiv:2306.11271*.

[5] A. Silva and M. C. Gombolay, "Encoding human domain knowledge to warm start reinforcement learning," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 5042–5050.

[6] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," *Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[7] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2017, pp. 1126–1135.

[8] C. Thapa, M. A. P. Chamikara, and S. Camtepe, "SplitFed: When federated learning meets split learning," 2020, *arXiv: 2004.12088*.

[9] Y. Jiang et al., "Model pruning enables efficient federated learning on edge devices," 2019, *arXiv: 1909.12326*.

[10] Z. Xu, F. Yu, J. Xiong, and X. Chen, "Helios: Heterogeneity-aware federated learning with dynamically balanced collaboration," 2019, *arXiv: 1912.01684*.

[11] M. Rapp, R. Khalili, and J. Henkel, "Distributed learning on heterogeneous resource-constrained devices," 2020, *arXiv: 2006.05403*.

[12] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[13] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton, "Big self-supervised models are strong semi-supervised learners," 2020, *arXiv: 2006.10029*.

[14] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz, "Importance estimation for neural network pruning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 11264–11272.

[15] Z. Xu, F. Yu, Z. Qin, C. Liu, and X. Chen, "DiReCtX: Dynamic resource-aware CNN reconfiguration framework for real-time mobile applications," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 40, no. 2, pp. 246–259, Feb. 2021.

[16] N. Lee, T. Ajanthan, and P. H. Torr, "SNIP: Single-shot network pruning based on connection sensitivity," in *Proc. Int. Conf. Learn. Representations*, 2019.

[17] X. Gong, "Delay-optimal distributed edge computing in wireless edge networks," in *Proc. IEEE Conf. on Comput. Commun.*, 2020, pp. 2629–2638.

[18] Y. Wang and M. Saksena, "Scheduling fixed-priority tasks with preemption threshold," in *Proc. 6th Int. Conf. Real-Time Comput. Syst. Appl.*, 1999, pp. 328–335.

[19] B. T. Polyak, "Gradient methods for minimizing functionals," *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, vol. 3, no. 4, pp. 643–653, 1963.

[20] S. Yue, J. Ren, J. Xin, S. Lin, and J. Zhang, "Inexact-ADMM based federated meta-learning for fast and continual edge learning," in *Proc. ACM 22nd Int. Symp. Theory, Algorithmic Found., Protoc. Des. Mobile Netw. Mobile Comput.*, 2021, pp. 91–100.

[21] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning: A meta-learning approach," 2020, *arXiv: 2002.07948*.

[22] S. Lin, G. Yang, and J. Zhang, "A collaborative learning framework via federated meta-learning," in *Proc. IEEE 40th Int. Conf. Distrib. Comput. Syst.*, 2020, pp. 289–299.

[23] J. Nutini, I. Laradji, and M. Schmidt, "Let's make block coordinate descent go fast: Faster greedy rules, message-passing, active-set complexity, and superlinear convergence," 2017, *arXiv: 1712.08859*.

[24] J. Nutini, M. Schmidt, I. Laradji, M. Friedlander, and H. Koepke, "Coordinate descent converges faster with the gauss-southwell rule than random selection," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1632–1641.

[25] J. Zhang, T. He, S. Sra, and A. Jadbabaie, "Why gradient clipping accelerates training: A theoretical justification for adaptivity," 2019, *arXiv: 1905.11881*.

[26] H. Karimi, J. Nutini, and M. Schmidt, "Linear convergence of gradient and proximal-gradient methods under the Polyak-Łojasiewicz condition," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discov. Databases*, Springer, 2016, pp. 795–811.

[27] S. Yue, J. Ren, J. Xin, D. Zhang, Y. Zhang, and W. Zhuang, "Efficient federated meta-learning over multi-access wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 5, pp. 1556–1570, May 2022.

[28] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories," in *Proc. Conf. Comput. Vis. Pattern Recognit. Workshop*, 2004, pp. 178–178.

[29] Y. LeCun, C. Corinna, and C. Burges, "MNIST handwritten digit database," 1998. Accessed: Sep. 25, 2023. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[30] A. Krizhevsky et al., "Learning multiple layers of features from tiny images," 2009.

[31] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Oort: Efficient federated learning via guided participant selection," in *Proc. 15th USENIX Symp. Operating Syst. Des. Implementation*, 2021, pp. 19–35.

[32] J. Zhang, S. Guo, X. Ma, H. Wang, W. Xu, and F. Wu, "Parameterized knowledge transfer for personalized federated learning," *Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 10092–10104, 2021.

[33] A. Shamsian, A. Navon, E. Fetaya, and G. Chechik, "Personalized federated learning using hypernetworks," in *Proc. Int. Conf. on Mach. Learn.*, PMLR, 2021, pp. 9489–9502.

[34] Y. Tan, G. Long, J. Ma, L. Liu, T. Zhou, and J. Jiang, "Federated learning from pre-trained models: A contrastive learning approach," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 19332–19344, 2022.

[35] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[36] S. Horvath, S. Laskaridis, M. Almeida, I. Leontiadis, S. Venieris, and N. Lane, "FjORD: Fair and accurate federated learning under heterogeneous targets with ordered dropout," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 12876–12889.

[37] B. Yuan, C. R. Wolfe, C. Dun, Y. Tang, A. Kyrillidis, and C. M. Jermaine, "Distributed learning of deep neural networks using independent subnet training," 2019, *arXiv: 1910.02120*.

[38] C. Dun, C. R. Wolfe, C. M. Jermaine, and A. Kyrillidis, "ResIST: Layer-wise decomposition of resnets for distributed training," 2021, *arXiv:2107.00961*.

[39] J. Xu and H. Wang, "Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective," 2020, *arXiv: 2004.04314*.

[40] J. Ren, Y. He, D. Wen, G. Yu, K. Huang, and D. Guo, "Scheduling in cellular federated edge learning with importance and channel awareness," 2020, *arXiv: 2004.00490*.

[41] Y. Grandvalet and B. Yoshua, "Semi-supervised learning by entropy minimization," in *Proc. NeurIPS*, 2004.
[42] C. Wei, K. Shen, Y. Chen, and T. Ma, "Theoretical analysis of self-training with deep networks on unlabeled data," 2020, *arXiv: 2010.03622*.
[43] K. Sohn et al., "FixMatch: Simplifying semi-supervised learning with consistency and confidence," 2020, *arXiv: 2001.07685*.
[44] D.-H. Lee et al., "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Proc. Workshop Challenges Representation Learn.*, 2013, Art. no. 896.
[45] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, "Unsupervised data augmentation for consistency training," 2019, *arXiv: 1904.12848*.
[46] X. Wang, J. Gao, M. Long, and J. Wang, "Self-tuning for data-efficient deep learning," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2021, pp. 10738–10748.
[47] R. Reed, "Pruning algorithms-a survey," *IEEE Trans. Neural Netw.*, vol. 4, no. 5, pp. 740–747, Sep. 1993.
[48] M. Ishikawa, "Structural learning with forgetting," *Neural Netw.*, vol. 9, no. 3, pp. 509–521, 1996.
[49] E. D. Karnin, "A simple procedure for pruning back-propagation trained neural networks," *IEEE Trans. Neural Netw.*, vol. 1, no. 2, pp. 239–242, Jun. 1990.
[50] X. Wang, C. Wang, X. Li, V. C. Leung, and T. Taleb, "Federated deep reinforcement learning for Internet of Things with decentralized cooperative edge caching," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9441–9455, Oct. 2020.

Zhaofeng Zhang received the BEng degree in electrical engineering from Huazhong University of Science and Technology, Wuhan, China, in 2015, and the MS degree in electrical engineering from Arizona State University, Tempe, AZ, USA, in 2017. Currently, he is working toward the PhD degree with the School of Electrical, Computer, and Energy Engineering in Arizona State University, Tempe, AZ, USA. His research interests include edge computing, statistical machine learning, deep learning, and optimization.

Sheng Yue received the BSc degree in mathematics and the PhD degree in computer science from Central South University, China, in 2017 and 2022, respectively. Currently, he is a postdoc with the Department of Computer Science and Technology, Tsinghua University, China. His research interests include edge computing, statistical machine learning, deep learning, and optimization.

Junshan Zhang (Fellow, IEEE) received the PhD degre from the School of ECE, Purdue University, in 2000, and was on the faculty of the School of ECEE at Arizona State University from 2000 to 2021. He is a professor with ECE Department, University of California Davis. His research interests fall in the general field of information networks and data science, including edge intelligence, reinforcement learning, continual learning, network optimization and control, and game theory, with applications in connected and automated vehicles, 5 G and beyond, wireless networks, IoT data privacy/security, and smart grid. He is a recipient of the ONR Young Investigator Award in 2005 and the NSF CAREER award in 2003. He received the IEEE Wireless Communication Technical Committee Recognition Award in 2016. His papers have won a few awards, including the Best Student Paper award at WiOPT 2018, the Kenneth C. Sevcik Outstanding Student Paper Award of ACM SIGMETRICS/IFIP Performance 2016, the Best Paper Runner-up Award of IEEE INFOCOM 2009 and IEEE INFOCOM 2014, and the Best Paper Award at IEEE ICC 2008 and ICC 2017. Building on his research findings, he co-founded Smartiply Inc., a Fog Computing startup company delivering boosted network connectivity and embedded artificial intelligence. He served as editor-in-chief for *IEEE Transactions on Wireless Communications* from 2019 to 2022 and is a senior editor for *IEEE/ACM Transactions on Networking*. He was TPC co-chair for several major conferences in communication networks, including IEEE INFOCOM 2012 and ACM MOBIHOC 2015. He was the general chair for ACM/IEEE SEC 2017 and WiOPT 2016. He was a distinguished lecturer of the IEEE Communications Society.