Adaptive Federated Pruning in Hierarchical Wireless Networks

Xiaonan Liu, *Member, IEEE,* Shiqiang Wang, *Senior Member, IEEE,* Yansha Deng, *Senior Member, IEEE,* Arumugam Nallanathan, *Fellow, IEEE*

***Abstract*—Federated Learning (FL) is a promising privacy-preserving distributed learning framework where a server aggregates models updated by multiple devices without accessing their private datasets. Hierarchical FL (HFL), as a device-edge-cloud aggregation hierarchy, can enjoy both the cloud server's access to more datasets and the edge servers' efficient communications with devices. However, the learning latency increases with the HFL network scale due to the increasing number of edge servers and devices with limited local computation capability and communication bandwidth. To address this issue, in this paper, we introduce model pruning for HFL in wireless networks to reduce the neural network scale. We present the convergence analysis of an upper on the l_2-norm of gradients for HFL with model pruning, analyze the computation and communication latency of the proposed model pruning scheme, and formulate an optimization problem to maximize the convergence rate under a given latency threshold by jointly optimizing the pruning ratio and wireless resource allocation. By decoupling the optimization problem and using Karush–Kuhn–Tucker (KKT) conditions, closed-form solutions of pruning ratio and wireless resource allocation are derived. Simulation results show that our proposed HFL with model pruning achieves similar learning accuracy compared with the HFL without model pruning and reduces about 50% communication cost.**

***Index Terms*—Hierarchical Wireless network, federated pruning, machine learning, communication and computatioin latency.**

I. Introduction

In recent years, with the availability of enormous mobile data and growing privacy concerns, stringent privacy protection laws, such as the European Commission's General Data Protection Regulation (GDPR) [1] and the Consumer Privacy Bill of Rights in the U.S. [2], have been proposed. These potentially impede the development of Artificial Intelligence (AI)-based frameworks which are mainly cloud/edge-centric, where data is delivered to a cloud/edge server for data analysis by machine learning (ML) algorithms [3], [4].

In response, Federated Learning (FL) has emerged as a powerful privacy-preserving distributed ML architecture [5]. The standard steps of FL are: 1) each device uses its dataset to train a local model; 2) devices send their local models to the server for model aggregation; 3) the server transmits the updated global model back to devices. These steps are repeated across multiple iterations until convergence. Therefore, in FL, only the updated local models, rather than the raw data, are transmitted to the server for model aggregation, which enables privacy preservation towards the development of AI-empowered applications.

However, FL usually suffers from a bottleneck of communication overhead before achieving convergence because of long transmission latency between the cloud server and devices [6]. Meanwhile, the wireless channel between the cloud server and devices can be unreliable due to wireless fading, which further affects model sharing and degrades learning performance under latency constraints. In addition, since some ML models have large size, directly communicating with the cloud server over the wireless channel by a massive number of devices could lead to congestion in the backbone network.

To mitigate this issue, hierarchical federated learning (HFL) framework is proposed, where small-cell base stations are equipped with edge servers to perform edge aggregation of local models from local devices [7]. When edge servers achieve a certain learning accuracy, updated edge models are transmitted to the cloud server for global aggregation. Therefore, by leveraging edge servers as intermediaries to perform partial model aggregation in proximity, the communication overhead reduces significantly. Also, more efficient communication and computation resource allocation, such as energy and bandwidth allocation, can be achieved by the coordination of edge servers [8], [9].

Unfortunately, there are still challenges for HFL in wireless networks. First, with the increasing number of edge servers and devices, the limited wireless resources cannot provide efficient services, which leads to high communication latency for model uploading. Second, the computation capabilities of devices are limited, which results in high computation latency, especially for large-scale learning models. To address these issues, federated pruning is introduced in [10], [11], where the model size is adapted during FL to reduce both communication and computation overhead and minimize the overall training time, while maintaining a similar learning accuracy as the original model. However, authors in [10], [11] only considered one edge server and multiple devices so that the number of devices access to the edge server is limited, leading to inevitable training performance loss [12]. Therefore, we see a necessity for leveraging a cloud server to access the massive training samples, while edge servers enjoy quick model updates from their local devices with model pruning.

Motivated by the above, in this work, we propose a joint

X. Liu and A. Nallanathan are with the School of Electronic Engineering and Computer Science, Queen Mary University of London (QMUL), U.K. (e-mail: {x.l.liu, a.nallanathan}@qmul.ac.uk).

Y. Deng is with the Department of Engineering, King's College London, London, WC2R 2LS, U.K. (e-mail: yansha.deng@kcl.ac.uk). (Corresponding author: Yansha Deng).

S. Wang is with IBM T. J. Watson Research Center, NY, USA. (e-mail: wangshiq@us.ibm.com).

model pruning and wireless resource allocation for HFL in wireless networks. First, the variation of computation and communication latency caused by the model pruning are mathematically analyzed. Second, the convergence analysis of HFL with model pruning is proposed. Then, the pruning ratio and wireless resource allocation under latency and bandwidth constraints are jointly optimized to improve learning performance. The main contributions are summarized as follows.

- To adapt to dynamical wireless environments, we propose HFL with adaptive model pruning for device-edge-cloud wireless networks. To the best of our knowledge, this is the first paper considering adaptive model pruning in HFL.
- We model the number of model weights based on the pruning ratio. Also, we model the computation and communication latency of the HFL framework under a given pruning ratio. Furthermore, we analyze the convergence of an upper bound on the l_2-norm of gradients for HFL with adaptive model pruning. Then, the pruning ratio and wireless resource allocation are jointly optimized to minimize the upper bound under latency and bandwidth constraints.
- To obtain the optimal closed-form solutions of pruning ratio and wireless resource allocation in each communcation round, we decouple the optimization problem into two sub-problems and deploy Karush–Kuhn–Tucker (KKT) conditions.
- Simulation results show that our proposed HFL with adaptive model pruning achieves similar learning accuracy compared to the HFL without model pruning and decreases about 50% communication cost. Also, the learning accuracy of our proposed HFL with adaptive model pruning is larger than that of non-hierarchy model pruning.

The rest of this paper is organized as follows. Section II presents the related works. The system model is detailed in Section III. The convergence analysis and problem formulation are presented in Section IV. The optimal pruning ratio and wireless resource allocation are described in Section V. The simulation results and conclusions are detailed in Section VI and Section VII, respectively.

II. Related Works

In this section, related works on neural network pruning, efficient FL, and resource allocation and device selection in HFL are briefly introduced in the following three subsections.

A. Neural Network Pruning

Early works considered model pruning are second-order Taylor expansion pruning [13], magnitude based pruning [14], iterative synaptic slow pruning (SynFlow) [15], and single-shot network pruning (SNIP) [16]. However, the computation of Hessian matrix in the second-order Taylor expansion pruning has high complexity which is infeasible for modern DNNs. The learning models in magnitude-based pruning need to be trained until convergence before the next pruning step, which leads to a high computation latency. SynFlow prunes the model at model initialization (before training), and SNIP prunes the model using first training round's gradient information, which cannot guarantee the learning accuracy as the original model. To reduce the complexity of neural networks and guarantee learning accuracy, importance-based pruning has become popular in recent years [17], where weights with smaller importance are removed from the network. It is observed that directly training the pruned network can reach a similar accuracy as pruning a pre-trained original network. In addition to the importance-based pruning that trains the learning model until convergence before the next pruning step, there are iterative pruning methods where the model is pruned after every few steps of training [18]. Furthermore, a dynamic pruning approach that allows the neural network to grow and shrink during training was proposed in [19]. However, the pruning techniques in [17]–[19] are mainly considered in centralized learning with full access to training data, which is fundamentally different from our adaptive HFL pruning that works with distributed datasets at local devices and preserves device privacy.

B. Efficient Federated Learning

To improve the computation and communication efficiency of FL, federated dropout (FedDrop) was studied [20]–[23]. A FedDrop scheme in [20] was proposed building on the classic dropout scheme for random model pruning. Specifically, in each iteration of the FL algorithm, several subnets were independently generated from the global model at the server using heterogeneous dropout rates, each of which was adapted to the state of an assigned channel. An adaptive FedDrop technique was proposed in [21] to optimize both server-device communication and computation costs by allowing devices to train locally on a selected subset of the global model. In [22], the authors argued that the metrics used to measure the performance of FedDrop and its variants were misleading, and they proposed and performed new experiments which suggested that FedDrop was actually detrimental to scaling efforts. In [23], ordered FedDrop was introduced, where a mechanism that achieved an ordered, nested representation of knowledge in neural networks and enabled the extraction of lower footprint submodels without the need for retraining. FedDrop is a simple way to prevent the learning model from overfitting through randomly dropping neurons and is only used during the training phase, which decreases communication and computation latencies and slightly improves learning accuracy. However, during the testing phase, the whole learning model is transmitted between the server and devices, which cannot guarantee efficient FL.

To address the issue in FedDrop, federated pruning was proposed [10], [11]. In [10], authors considered federated pruning of the whole learning model, the model gradient was transmitted between the edge server and devices, and the model was updated by gradient averaging. However, pruning the convolutional layer of convolutional neural network (CNN) decreases the robust capability of CNN and delivering the whole model gradient cannot guarantee a low communication latency. In [11], a novel FL approach with adaptive and

distributed parameter pruning, called PruneFL, was proposed, which adapted the model size during FL to minimize the overall training time, while maintaining a similar accuracy as the original model. PruneFL included initial pruning at a selected device and further pruning as part of the FL process. The model size was adapted during this process, which included maximizing the approximate empirical risk reduction divided by the time of one FL round. However, initial pruning at only a selected device in [11] may lead to extra training time if the trained model cannot be generalized to other devices and the proposed PruneFL cannot be adapt to dynamic wireless environments. In addition, authors in [10], [11] only considered one edge server and multiple devices, the performance of federated model pruning in hierarchical wireless networks was still unclear.

Except from FedDrop and federated pruning for efficient FL in training or testing phases, some other methods such as device-to-device (D2D)-assisted efficient FL protocol, were proposed to guarantee communication-efficient FL in wireless networks [24]–[26]. In [24], a D2D-assisted FL scheme, called (D2D-FedAvg), over mobile edge computing (MEC) networks was proposed to minimize the communication cost. D2D-FedAvg created a two-tier learning model where D2D learning groups communicated their results as a single entity to the MEC for traffic reduction. Also, D2D grouping, master device selection, and D2D exit were proposed to form a complete D2D-assisted FedAvg. In [25], sign-SGD was considered in a D2D-assisted FL scheme to minimize communication costs. In [26], an efficient FL protocol called local-area network (LAN) FL was proposed, which involved a hierarchical aggregation mechanism in LAN due to its abundant bandwidth and almost negligible monetary cost than wide-area network (WAN). LAN FL could accelerate the learning process and reduce the monetary cost with frequent local aggregation in the same LAN and infrequent global aggregation on a cloud across WAN. However, the computation efficiency of FL in [24]–[26] was not considered, especially for devices with limited computation capability.

C. Resource Allocation and Device Selection

In [27]–[32], computation and communication resource allocation and edge association of HFL were investigated. Specifically, in [27], the sum of system and learning costs was minimized by optimizing bandwidth, computing frequency, power allocation, and sub-carrier assignment by successive convex approximation and Hungarian algorithms. In [28], a conflict graph-based solution was proposed to minimize the overall energy consumption of training local models subject to HFL latency constraints. In [29], a hierarchical game framework was proposed to study the dynamics of edge association and resource allocation in self-organizing HFL networks, and a Stackelberg differential game was used to model the optimal bandwidth and reward allocation strategies of the edge servers and devices. In [30], a fog-enabled FL framework was proposed to facilitate distributed learning for delay-sensitive applications in resource-constrained internet of things environments, where a greedy heuristic approach was formulated to select an optimal fog node for model aggregation. In [31], a multi-layer FL protocol, called HybridFL, was designed for a MEC architecture, which could mitigate stragglers and end device drop-out. In [32], an in-network aggregation process (INA) was designed to enable decentralizing the model aggregation process at the server, thereby minimizing the training latency for the whole FL network. However, the whole FL learning model still needs to be transmitted between servers and devices in [27]–[32], which cannot guarantee computation and communication efficient HFL.

III. System Model

In a HFL network, we assume a set of edge servers $\mathcal{K} = \{k = 1, 2, ..., K\}$, a set of mobile devices $\mathcal{N} = \{n = 1, 2, ..., N\}$, and a cloud server S. The kth edge server provides wireless connections for $\mathcal{N}_k \in \mathcal{N}$ mobile devices and is connected to the cloud server S through a fiber link. Each edge server is equipped with M antennas and each mobile device is equipped with a single antenna. In addition, the nth device has a local dataset $\mathcal{D}_n = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{D_n}$, where $\boldsymbol{x}_i$ is the ith input data sample, y_i is the corresponding labeled output of $\boldsymbol{x}_i$, and D_n is the number of data samples. The main notations in this paper are listed in Table I.

A. Model Pruning

In the HFL framework, the scale of neural networks can be very large with increasing requirements of learning performance, such as high learning accuracy. Consequently, model updating in local devices and transmission between edge servers and devices could cause high computation and communication latency. To solve these problems, model pruning is deployed to decrease the model size.

Pruning unimportant neurons or weights effectively decreases the model size and only causes a small performance loss. The learning accuracy only decreases dramatically with a high pruning ratio. According to [17], the importance of weight is quantified by the error induced by removing it, and the induced error is measured as a squared difference of prediction errors with and without the jth weight $w_{n,j}$ of the nth device, which is denoted as

$$\mathcal{I}_{n,j} = \left(F_n(\boldsymbol{w}_n) - F_n(\boldsymbol{w}_n | w_{n,j} = 0)\right)^2, \quad (1)$$

where $F_n(\boldsymbol{w}_n)$ and $\boldsymbol{w}_n$ are the local loss function and local model of the nth device, respectively. The larger the error is, the more important the weight will be. However, calculating $\mathcal{I}_{n,j}$ for each weight of the nth device in (1) is computationally expensive, especially when the nth device has a large number of model weights. To decrease the computational complexity of the importance calculation, we calculate the difference between the jth local model weight and the updated jth local model weight as

$$\hat{\mathcal{I}}_{n,j} = |w_{n,j} - \hat{w}_{n,j}|. \quad (2)$$

The importance calculation in (2) is easily computed since the updated local model weight $\hat{w}_{n,j}$ is already available from backpropagation.

TABLE I
MAIN NOTATIONS

$\mathcal{K}$	set of edge servers	$\mathcal{N}$	set of mobile devices
$\mathcal{D}_n$	local dataset	$F_n(\boldsymbol{w}_n)$	local loss function
$\boldsymbol{w}_n$	local model	$\mathcal{I}_{n,j}$	importance of weight
ρ_n	pruning ratio	$\boldsymbol{m}_{k,n}^{q,e}$	pruning mask
$W_{n,\text{conv}}$	number of weights in convolutional layers	$W_{n,\text{fully}}$	number of weights in fully-connected layers
$\nabla F_n(\boldsymbol{w}_{k,n}^{q,e,t}, \xi_{k,n}^{q,e,t})$	local gradient	η	learning rate
C_n	number of CPU cycles	f_n	CPU frequency
T	number of local iterations	$T_{n,e}^{\text{cmp}}$	local computation latency
$R_{n,k,e}^{\text{up}}$	uplink transmission rate	$b_{n,e}$	bandwidth fraction
B	total bandwidth	$\hat{q}$	quantization bit
$T_{n,k,e}^{\text{up}}$	uplink transmission latency	$\boldsymbol{w}_G$	cloud model

The primary objective of model pruning is to alleviate the high computational demands of the training and inference phases. When the lth layer of the learning model is pruned through importance-based model pruning given pruning ratio $\rho_{n,l}$, there is no need to perform forward and backward passes or gradient updates on the pruned units. As a result, model pruning offers gains both in terms of floating point operation (FLOP) count and model size [33]. Specifically, for the lth fully-connected layer, the number of weights is calculated as

$$W_{n,l} = \lceil \rho_{n,l} W_{n,l,\text{in}} \rceil \lceil \rho_{n,l} W_{n,l,\text{out}} \rceil, \tag{3}$$

where $\rho_{n,l}$ is the pruning ratio of the lth layer of the nth device, and $W_{n,l,\text{in}}$ and $W_{n,l,\text{out}}$ correspond to the number of input and output weights, respectively, and the number of weights is decreased by $\frac{W_{n,l,\text{in}} W_{n,l,\text{out}}}{\lceil \rho_{n,l} W_{n,l,\text{in}} \rceil \lceil \rho_{n,l} W_{n,l,\text{out}} \rceil} \sim \frac{1}{\rho_{n,l}}$. Furthermore, the bias terms are reduced by a factor of $\frac{W_{n,l,\text{out}}}{\lceil \rho_{n,l} W_{n,l,\text{out}} \rceil} \sim \frac{1}{\rho_{n,l}}$. For simplicity, we directly deploy ρ_n to denote the pruning ratio of the nth device in the following sections. When considering the pruning strategy, the pruned weight in the matrix is set to be zero, and the matrix can be a sparse matrix. In the sparse matrix, only the non-zero weights and their corresponding position information are transmitted, namely, the index-value pairs of the sparse matrix are transmitted. Based on the received index-value pairs of the sparse matrix, the edge server is able to obtain the position information of the unpruned weights and proceed model aggregation.

B. Learning Process in HFL

The proposed HFL architecture is shown in Fig. 1, where learning models are aggregated in the edge and cloud servers. As a result, learning models updated by mobile devices in a global iteration include edge aggregation and cloud aggregation. To quantify training overhead in the HFL framework, we formulate latency overhead in edge and cloud aggregation within one global iteration. The learning process is introduced as follows:

1) Edge Aggregation: This stage has five steps, including cloud model broadcasting, edge model broadcasting, local model updating, transmission, and aggregation. That is, the cloud server broadcasts the cloud model to edge servers, each edge server broadcasts the edge model to its associated mobile devices, and mobile devices update local models with their datasets and transmit them to their associated edge servers for model aggregation, which are introduced as following steps.

a) Step 1. Cloud Model Broadcast: In the qth global communication round, the cloud model $\boldsymbol{w}_G^q$ is broadcast to all edge servers. Considering a convolutional neural network (CNN) and the number of weights W in CNN is calculated as

$$W = W_{\text{conv}} + W_{\text{fully}} = W_{\text{conv}} + \sum_{l=1}^{L-1} N_l N_{l+1}, \tag{4}$$

where W_{conv} is the number of weights of convolutional layers, L is the number of fully-connected layers, and $\sum_{l=1}^{L-1} N_l N_{l+1}$ is the total number of weights of fully-connected layers. Since the fiber links between the cloud and edge servers have a high transmission rate, the transmission latency between them is ignored.

b) Step 2. Edge Model Broadcasting: The kth edge server transmits the received cloud model $\boldsymbol{w}_G^q$ to its associated mobile devices through downlink transmission. In actual scenarios, the transmission latency of downlink weights broadcast is very small due to sufficient downlink broadcast channel bandwidth. Therefore, downlink transmission latency is ignored in the study of this paper.

c) Step 3. Local Model Updating: When the nth device receives the model from the kth edge server in the beginning of the eth edge communication round, it deploys a pruning mask $\boldsymbol{m}_{k,n}^{q,e}$ to prune the received edge model $\boldsymbol{w}_{k,n}^{q,e}$, which is calculated as

$$\boldsymbol{w}_{k,n}^{q,e,0} = \boldsymbol{w}_{k,n}^{q,e} \odot \boldsymbol{m}_{k,n}^{q,e}. \tag{5}$$

In the pruning mask $\boldsymbol{m}_{k,n}^{q,e}$, if $m_{k,n}^{q,e,j} = 1$, $\boldsymbol{w}_{k,n}^{q,e,0}$ contains the jth model weight, otherwise, $m_{k,n}^{q,e,j} = 0$, and $m_{k,n}^{q,e,j}$ is determined by (2). In addition, t means the tth iteration in local model updating. Then, by gradient descent, the nth device updates the pruned local model $\boldsymbol{w}_{k,n}^{q,e,0}$. Given a pruning ratio $\rho_{n,e}$ of the nth mobile device, the number of weights after pruning is calculated as

$$W_{\rho_{n,e}} = W_{n,\text{conv}} + (1 - \rho_{n,e}) W_{n,\text{fully}}. \tag{6}$$

In (6), we mainly consider weight pruning in the fully-connected layer rather than the convolutional layer. It is

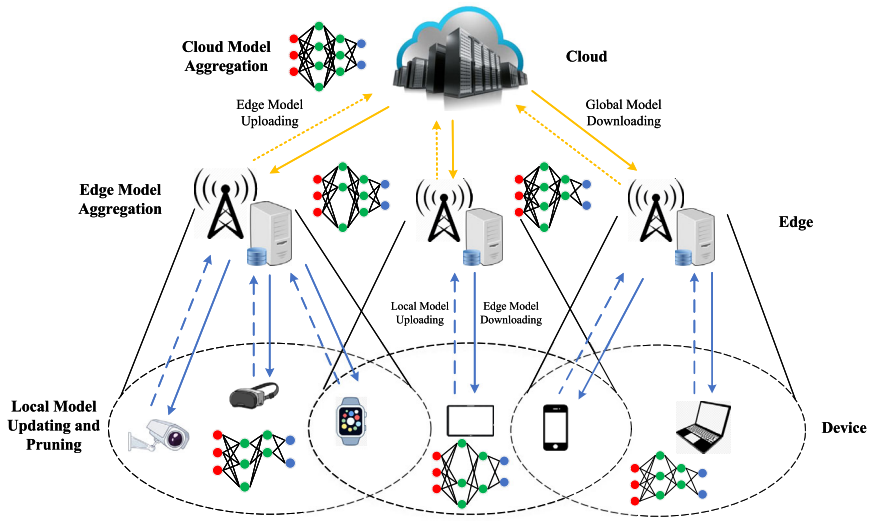


Fig. 1. Hierarchical Federated Learning (HFL) framework.

because pruning in the convolutional layer decreases the robust capability of CNN.

The local loss $F_n(\boldsymbol{w}_{k,n}^{q,e,t})$ of the nth device in the tth iteration is defined on its local dataset $\mathcal{D}_n$ and is denoted as

$$F_n(\boldsymbol{w}_{k,n}^{q,e,t}) = \frac{1}{D_n}\sum_{i=1}^{D_n} f_n(\boldsymbol{x}_i, y_i, \boldsymbol{w}_{k,n}^{q,e,t}), \tag{7}$$

where $f_n(\boldsymbol{x}_i, y_i, \boldsymbol{w}_{k,n}^{q,e,t})$ is the loss function (e.g., cross-entropy and mean square error (MSE)) that denotes the difference between the model output and the desired output based on the local model $\boldsymbol{w}_{k,n}^{q,e,t}$. The fact that calculating the loss over the whole dataset is time-consuming, and in some cases it is not feasible because of the limited memory capacity of the mobile device, we employ minibatch stochastic gradient descent (SGD) in which the nth mobile device deploys a sub-dataset of its dataset to calculate the loss. The local model updating in the tth iteration is calculated as

$$\boldsymbol{w}_{k,n}^{q,e,t+1} = \boldsymbol{w}_{k,n}^{q,e,t} - \eta \nabla F_n(\boldsymbol{w}_{k,n}^{q,e,t}, \xi_{k,n}^{q,e,t}) \odot \boldsymbol{m}_{k,n}^{q,e}, \tag{8}$$

where $\nabla F_n(\boldsymbol{w}_{k,n}^{q,e,t}, \xi_{k,n}^{q,e,t})$ is the gradient in the tth iteration, η is the learning rate, $\xi_{k,n}^{q,e,t} \subseteq \mathcal{D}_n$ is the mini-batch randomly selected from the data samples $\mathcal{D}_n$ of the nth mobile device.

Then, we can calculate the computation latency incurred by the nth device. We assume that the number of CPU cycles for the nth device to update one model weight is C_n, thus, the total number of CPU cycles to run one local iteration is $C_n W_{\rho_{n,e}}$. We denote that the allocated CPU frequency of the nth device for computation is f_n with $f_n \in [f_n^{\min}, f_n^{\max}]$. Therefore, the total latency of local iterations is calculated as

$$T_{n,e}^{\text{cmp}} = \frac{TC_n W_{\rho_{n,e}}}{f_n} = \frac{TC_n[W_{n,\text{conv}} + (1-\rho_n)W_{n,\text{fully}}]}{f_n}. \tag{9}$$

where T is the number of iterations, $\frac{TC_n W_{n,\text{conv}}}{f_n}$ is the computation latency of the convolutional layers, and $\frac{TC_n(1-\rho_n)W_{n,\text{fully}}}{f_n}$ is the computation latency of the fully-connected layers.

d) Step 4. Local Model Uplink Transmission: After finishing local model updating, the nth device transmits its updated local model $\boldsymbol{w}_{k,n}^{q,e,T}$ to the kth edge server, which results in wireless transmission latency. We assume that the set of mobile devices associated with the kth server is $\mathcal{S}_k$ with $\mathcal{S}_k \subseteq \mathcal{N}$.

According to [7] and [34], we consider an orthogonal frequency-division multiple access (OFDMA) protocol for devices associated with the edge server and there is no interference among devices. The achievable transmission rate between the nth mobile device and the kth edge server in the eth edge communication round is denoted as

$$R_{n,k,e}^{\text{up}} = b_{n,e} B \log_2\left(1 + \frac{g_{n,k}^e p_n}{\sigma^2}\right), \tag{10}$$

where $b_{n,e}$ is the bandwidth fraction allocated to the nth mobile device in the eth edge communication round, B is the total bandwidth allocated to each edge server, $g_{n,k}^e$ is the channel gain between the nth mobile device and the kth edge server, p_n is the transmission power of the nth mobile device, and σ^2 is the noise power. Then, the uplink transmission latency from the nth mobile device to the kth edge server is calculated as

$$T_{n,k,e}^{\text{up}} = \frac{\hat{q} W_{\rho_{n,e}}}{R_{n,k,e}^{\text{up}}} = \frac{\hat{q}[W_{n,\text{conv}} + (1-\rho_n)W_{n,\text{fully}}]}{R_{n,k}^{\text{up}}}, \tag{11}$$

where $\hat{q}$ is the quantization bit, $\frac{\hat{q} W_{n,\text{conv}}}{R_{n,k}^{\text{up}}}$ is the uplink transmission latency of the convolutional layers, and $\frac{\hat{q}(1-\rho_n)W_{n,\text{fully}}}{R_{n,k}^{\text{up}}}$ is the uplink transmission latency of the fully-connected layers. Quantization bit determines the precision of the model weights in wireless transmission, and the learning accuracy decreases with a lower quantization bit [35]. Furthermore, a higher pruning ratio and a higher quantization bit achieves a lower learning accuracy. It is because more important weights are pruned with a higher pruning ratio, even if the precision of the weights is high.

e) Step 5. Edge Model Aggregation: Because of model pruning, some model weights are not contained in the received local models. Let $\mathcal{N}_{k,q,e}^{j}$ be the set of mobile devices associated with the kth edge server and containing the jth model weight in the eth edge communication round. Then, edge model update of the jth model weight is performed by aggregating local models with the jth model weight available, which is calculated as

$$\hat{w}_k^{q,e,T,j} = \frac{1}{|\mathcal{N}_{k,q,e}^{j}|} \sum_{n \in \mathcal{N}_{k,q,e}^{j}} w_{k,n}^{q,e,T,j}, \tag{12}$$

where $|\mathcal{N}_{k,q,e}^{j}|$ is the number of local models containing the jth model weight.

Then, the kth edge server delivers $\boldsymbol{w}_k^{q,e+1}$ to its associated mobile devices in $\mathcal{S}_k$ for the next round of local model updating in step 1. Actually, steps 1 to 5 of edge aggregation continue iterating until the kth edge server reaches a certain level of accuracy. Each edge server does not access the local dataset of each mobile device, which preserves personal data privacy. Since each edge server typically has high computation capability, the computation latency of edge model aggregation is neglected.

2) Cloud Aggregation: This stage has two steps, including edge model uploading and cloud model aggregation. First, the kth edge server transmits updated $\boldsymbol{w}_k^{q,E}$ to the cloud for global aggregation after E edge communication rounds. Then, the cloud server aggregates edge models from all edge servers as

$$\boldsymbol{w}_G^{q+1} = \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} \boldsymbol{w}_k^{q,E}, \tag{13}$$

where $|\mathcal{K}|$ is the number of edge servers. The detailed HFL with model pruning is presented in **Algorithm 1**.

The edge devices first update the received edge model $\boldsymbol{w}_{k,n}^{q,e}$ by $\boldsymbol{w}_{k,n}^{q,e} = \boldsymbol{w}_{k,n}^{q,e} - \eta \nabla F_n(\boldsymbol{w}_{k,n}^{q,e}, \xi_{k,n}^{q,e})$ for several iterations. Then, the importance of each weight in the local model is calculated by (2), and the local model weights are sorted in a descending order. Given the pruning ratio ρ_n of the nth device and the number of weights of the fully-connected layers $W_{n,\text{fully}}$, the weights whose importance ranked last $\rho_n W_{n,\text{fully}}$ are pruned and their corresponding coefficients in $m_{k,n}^{q,e,j}$ are set to be 0, otherwise, the corresponding coefficients in $m_{k,n}^{q,e,j}$ are set to be 1. Then, the device is able to update the pruned model by (8), which further reduces the computation latency during the training phase.

Algorithm 1 HFL with model pruning

Local dataset $\mathcal{D}_n$ on N_k local devices associated with the kth edge server, learning rate η, pruning policy $\mathbb{P}$, number of local epochs T, number of edge communication rounds E, edge model parameterized by $\boldsymbol{w}_k^{q,e}$, number of global communication rounds Q, global model parameterized by $\boldsymbol{w}_G^{q}$.
for global communication round q = 1,...,Q **do**
  Generate $\boldsymbol{w}_k^{q,0} = \boldsymbol{w}_G^{q}$.
  for edge communication round e = 1,...,E **do**
    for local device n = 1,...,N_k **do**
      Local updating $\boldsymbol{w}_{k,n}^{q,e} = \boldsymbol{w}_{k,n}^{q,e} - \eta \nabla F_n(\boldsymbol{w}_{k,n}^{q,e}, \xi_{k,n}^{q,e})$ for several iterations and generate mask $\boldsymbol{m}_{k,n}^{q,e}$ by (2).
      Generate $\boldsymbol{w}_{k,n}^{q,e,0} = \boldsymbol{w}_{k,n}^{q,e} \odot \boldsymbol{m}_{k,n}^{q,e}$.
      for iteration t = 1,2,...,T **do**
        Update $\boldsymbol{w}_{k,n}^{q,e,t+1}$ as (8).
      end for
    end for
    for parameter j in local models **do**
      Find $\mathcal{N}_{k,q,e}^{j} = \{n : m_{k,n}^{q,e,j} = 1\}$.
      Update $\hat{w}_{k,q}^{e,T,j}$ as (12).
    end for
  end for
  Update $\boldsymbol{w}_G^{q+1}$ as (13).
end for

C. Computation and Communication Latency

Synchronous training is deployed in HFL and we mainly consider computation and uplink transmission latency. Note that the computational complexity of importance calculation is very low as compared with the local forward and back propagation during model training. Therefore, the computational complexity of importance calculation is ignored, and the latency for local computation and uplink transmission is written as

$$T_{n,k,e} = T_{n,e}^{\text{cmp}} + T_{n,k,e}^{\text{up}} \tag{14}$$

$$= \frac{TC_n W_{\rho_{n,e}}}{f_n} + \frac{\hat{q} W_{\rho_{n,e}}}{R_{n,k,e}^{\text{up}}}. \tag{15}$$

As a result, the latency of the kth edge server in the eth edge communication round is expressed as

$$T_{k,e} = \max_{n \in \mathcal{S}_k} \{T_{n,k,e}\}. \tag{16}$$

From (16), we observe that the bottleneck of the computation and communication latency is affected by the last device that finishes all local iterations and uplink transmission after local model updating.

IV. Convergence Analysis and Problem Formulation

In this section, the convergence analysis of model pruning in HFL is first analyzed. Then, an optimization problem is formulated to minimize the upper bound of the convergence analysis.

A. Convergence Analysis

Since the neural network is non-convex in general, the average l_2-norm of gradients is deployed to evaluate the convergence performance [36], [37]. The following assumptions are employed in hierarchical federated pruning convergence analysis.

Assumption 1. (Smoothness) Cost functions $F_1, ..., F_N$ are all $L-$ smooth:

$$\|\nabla F_n(\boldsymbol{w}_1) - \nabla F_n(\boldsymbol{w}_2)\| \leq L\|\boldsymbol{w}_1 - \boldsymbol{w}_2\|, \tag{17}$$

where L is a positive constant.

Assumption 2. (Pruning-induced Noise) Different from the other convergence analysis of HFL in [38]–[41], we consider the effect of pruning-induced noise. According to [42], the model error of the nth device under the pruning ratio $\rho_{n,e}$ is bounded by

$$\mathbb{E}\|\boldsymbol{w}_{k,n}^{q,e} - \boldsymbol{w}_{k,n}^{q,e} \odot \boldsymbol{m}_{k,n}^{q,e}\|^2 \leq \rho_{n,e} D^2, \tag{18}$$

where D is a positive constant.

Assumption 3. (Bounded Gradient) The second moments of stochastic gradients is bounded [43], [44], which is denoted as

$$\mathbb{E}\|\nabla F_n(\boldsymbol{w}_{k,n}^{q,e,t}, \xi_{k,n}^{q,e,t})\|^2 \leq \phi^2. \tag{19}$$

In (19), ϕ is a positive constant, and $\xi_{k,n}^{q,e,t}$ are mini-batch data samples for any k, n, q, e, t.

Assumption 4. (Gradient Noise for IID data) Under IID data distribution, we assume that

$$\mathbb{E}[\nabla F_n(\boldsymbol{w}_{k,n}^{q,e,t}, \xi_{k,n}^{q,e,t})] = \nabla F_n(\boldsymbol{w}_{k,n}^{q,e,t}), \tag{20}$$

and

$$\mathbb{E}\|\nabla F_n(\boldsymbol{w}_{k,n}^{q,e,t}, \xi_{k,n}^{q,e,t}) - \nabla F_n(\boldsymbol{w}_{k,n}^{q,e,t})\|^2 \leq \hat{\sigma}^2. \tag{21}$$

In (21), $\hat{\sigma} > 0$ is a constant. In addition, to extend the analysis to the case with non-IID data distribution, we can change the assumption of gradient noise for IID data to be gradient noise for non-IID data and revise the **Lemma 4** in Appendix A - Proof of Theorem 1. Then, the convergence analysis can be used for non-IID data distribution.

Theorem 1: With the above assumptions, HFL with pruning converges to a small neighborhood of a stationary point of standard FL as follows:

$$\frac{1}{QW}\sum_{q=1}^{Q}\sum_{j=1}^{W}\mathbb{E}\|\nabla F^j(\boldsymbol{w}_q)\|^2 \leq \frac{2\mathbb{E}[F(\boldsymbol{w}^0) - F(\boldsymbol{w}^*)]}{QW\eta ET} + H_1 + H_2\sum_{e=1}^{E}\sum_{n=1}^{N}\rho_{n,e}, \tag{22}$$

where H_1 and H_2 are expressed as

$$H_1 = 3L\eta TEW\phi^2 + \frac{\phi^2 N\eta^2 T^2 L^3 + 3LW\eta ETN\hat{\sigma}^2 + 3WEL^3T^3\eta^3\phi^2 N}{\Gamma^*}, \tag{23}$$

and

$$H_2 = \frac{2EL^2 + 6W\eta L^3 D^2 T}{\Gamma^*}. \tag{24}$$

In (22), (23), and (24), Q is the number of global communication rounds, E is the number of edge communication rounds, T is the number of iterations in each device, W is the total number of model weights, and Γ^* is the minimum occurrence of the parameter in local models of all rounds.

Proof : Please refer to Appendix A.

B. Problem Formulation

Based on the aforementioned system model and convergence analysis, we consider an optimization problem with respect to minimizing the upper bound in (22). The optimization problem is formulated as follows:

$$\min_{b_{n,e},\rho_{n,e}} \quad H_2\sum_{e=1}^{E}\sum_{n=1}^{N}\rho_{n,e}, \tag{25}$$

$$s.t. \quad T_{n,k,e} \leq T_{\text{th}}, \tag{26}$$

$$\sum_{n=1}^{N} b_{n,e} \leq 1, \tag{27}$$

$$0 \leq b_{n,e} \leq 1, \tag{28}$$

$$\rho_{n,e} \in [0, 1]. \tag{29}$$

where T_{th} in (26) represents the computation and communication latency constraint, (27) and (28) represent the wireless resource constraints, namely, the bandwidth fraction $b_{n,e}$ allocated to the nth device in the eth edge communication round cannot larger than the total bandwidth B, and (29) represents the pruning ratio constraint, which should be carefully selected to avoid the learning accuracy decreasing sharply.

Minimizing the global loss function requires an explicit form about how to select the pruning ratio based on the latency and wireless resource constraints. Since it is almost impossible to know the training performance exactly before the model has been trained, we turn to find an upper bound of l_2-norm of gradients and minimize it for the global loss minimization [45]. Obviously, the optimization problem in (25) is a mixed integer non-linear programming (MINLP) problem, which is non-convex and impractical to directly obtain optimal solutions. Therefore, we decompose the original problem into several sub-problems to obtain sub-optimal solutions.

V. Optimal Pruning Ratio and Wireless Resource Allocation

In this section, we decompose the optimization in (25) into two sub-problems with the aim to derive the optimal solutions of pruning ratio and wireless resource allocation.

A. Optimal Pruning Ratio

Based on (26), the transmission and computation latency of the nth mobile device should satisfy the latency threshold, which is denoted as

$$\frac{TC_nW_{\rho_{n,e}}}{f_n} + \frac{\hat{q}W_{\rho_{n,e}}}{R_{n,k,e}^{\text{up}}} \leq T_{\text{th}}. \tag{30}$$

Theorem 2: The pruning ratio of the nth device associated with the kth edge server should satisfy

$$\rho_{n,e}^{\star} \geq \left(1 - \frac{T_{\text{th}} - T_{n,e}^{\text{cmp-conv}} - T_{n,k,e}^{\text{com-conv}}}{T_{n,e}^{\text{cmp-fully}} + T_{n,k,e}^{\text{com-fully}}}\right)^{+}, \tag{31}$$

where $T_{n,e}^{\text{cmp-conv}}$ and $T_{n,k,e}^{\text{com-conv}}$ are computation and uplink transmission latency of the convolutional layer, respectively. In (31), $T_{n,e}^{\text{cmp-fully}}$ and $T_{n,k,e}^{\text{com-fully}}$ are computation and uplink

TABLE II
SIMULATION PARAMETERS OF HFL WITH ADAPTIVE MODEL PRUNING AND WIRELESS RESOURCE ALLOCATION

Transmission power of device	28 dBm	Bandwidth	20MHz
CPU frequency of device	3 GHz	Learning rate	0.001
AWGN noise power	-110 dBm	Batchsize	128
Quantization bit	64	Number of global communication rounds	10
Number of edge servers	5	Number of edge communication rounds	5
Number of devices per edge server	5	Number of local iterations	10

transmission latency of the fully-connected layer, respectively. Also, in (31), $(z)^+ = \max(z, 0)^+$.

Proof : Please refer to Appendix B.

Remark 1: From Theorem 2 and (59) in the Appendix B, the optimal pruning ratio is jointly determined by the computation capability and uplink transmission rate of the local device. When both the computation capability and uplink transmission rate are high for a local device, a small pruning ratio can be adopted.

B. Optimal Wireless Resource Allocation

According to the optimal pruning ratio in (31), the optimization problem in (25) is rewritten as

$$\min \quad H_2 \sum_{e=1}^{E} \sum_{n=1}^{N} \left(1 - \frac{T_{\text{th}} - T_{n,e}^{\text{cmp-conv}} - T_{n,k,e}^{\text{com-conv}}}{T_{n,e}^{\text{cmp-fully}} + T_{n,k,e}^{\text{com-fully}}} \right), \quad (32)$$

subject to (27) and (28). When $T_{n,k,e}^{\text{com-conv}} = W_{n,\text{conv}} \hat{q} / R_{n,k,e}^{\text{up}}$ and $T_{n,k,e}^{\text{com-fully}} = W_{n,\text{com-fully}} \hat{q} / R_{n,k,e}^{\text{up}}$, (32) is further derived as

$$\min \quad H_2 \sum_{e=1}^{E} \sum_{n=1}^{N} \left(1 - \frac{R_{n,k,e}^{\text{up}} (T_{\text{th}} - T_{n,e}^{\text{cmp-conv}}) - \hat{q} W_{n,\text{conv}}}{R_{n,k,e}^{\text{up}} T_{n,e}^{\text{cmp-fully}} + \hat{q} W_{n,\text{fully}}} \right). \quad (33)$$

The optimal solution of wireless resource allocation is obtained by solving the optimization problem in (33). First, according to the following Lemma 1, we prove that the optimization problem in (33) is convex.

Lemma 1: The optimization problem in (33) is convex.

Proof : Please refer to Appendix C.

According to the Lagrange multiplier method, the optimal wireless resource allocation is obtained in the following theorem.

Theorem 3: To obtain the optimal learning performance, the optimal bandwidth allocated to the nth deivce should satisfy

$$b_{n,e}^{\star} = \frac{\sqrt{\frac{(V_1 V_4 + V_2 V_3) B \log_2 \left(1 + \frac{g_{n,k}^e p_n}{\sigma^2}\right)}{\lambda^{\star}}} - V_4}{B V_3 \log_2 \left(1 + \frac{g_{n,k}^e p_n}{\sigma^2}\right)}, \quad (34)$$

where $V_1 = T_{\text{th}} - T_{n,e}^{\text{cmp-conv}}$, $V_2 = \hat{q} W_{n,\text{conv}}$, $V_3 = T_{n,e}^{\text{cmp-fully}}$, $V_4 = \hat{q} W_{n,\text{fully}}$, and $\lambda^{\star}$ is the optimal Lagrange multiplier.

Proof : Please refer to Appendix D.

Based on Theorem 2 and 3, the optimal pruning ratio is written as

$$\rho_{n,e}^{\star} = 1 - \frac{b_{n,e}^{\star} (T_{\text{th}} - T_{n,e}^{\text{cmp-conv}}) B \log_2 \left(1 + \frac{g_{n,k}^e p_n}{\sigma^2}\right) - \hat{q} W_{n,\text{conv}}}{b_{n,e}^{\star} T_{n,e}^{\text{cmp-fully}} B \log_2 \left(1 + \frac{g_{n,k}^e p_n}{\sigma^2}\right) + \hat{q} W_{n,\text{fully}}}. \quad (35)$$

Remark 2: From Theorem 3, the wireless resource allocation decreases with better channel condition. More wireless resource is allocated to the local devices with bad channel condition to guarantee transmission latency. In addition, more wireless resource is allocated to the local devices with high computation capability, which can decrease the communication latency and improve the convergence rate.

VI. SIMULATION RESULTS

In this section, we examine the effectiveness of our proposed HFL with model pruning via simulation. In the simulation, we consider a scenario with five edge servers and each edge server has five devices participating in model training. We deploy a common CNN model for image classification over the datasets MNIST, Fashion MNIST, and CIFAR10, which contain 50000 training samples and 10000 testing samples, respectively. The training data are shuffled to guarantee the local data are IID. The input size of CNN is $1 \times 28 \times 28$, and the sizes of the first and second convolutional layers are $32 \times 28 \times 28$ and $64 \times 14 \times 14$, respectively. The sizes of the first and second max-pooling layers are $32 \times 14 \times 14$ and $64 \times 7 \times 7$, respectively. The sizes of the first and second fully-connected layers are 3136 and 8, respectively. The size of the output layer is 10. The devices exchange learning models with edge servers over the wireless channel. The main simulation parameters are listed in Table I. In addition, each global communication round updates learning models by 1250 iterations, including 5 edge servers, 5 edge devices per edge server, 5 edge communication rounds, and 10 local iterations.

A. HFL with Importance-based Pruning

Fig. 2 (a) and Fig. 2 (b) plot the testing loss of importance-based model pruning of HFL with different pruning ratios on datasets MNIST and Fashion MNIST, respectively. Fig. 2 (c) plots the testing accuracy of importance-based model pruning of HFL under different pruning ratios on datasets MNIST, Fashion MNIST, and CIFAR10. It is observed that the testing loss increases, and convergence rate and testing accuracy decrease with increasing value of the pruning ratio. It is because a larger pruning ratio indicates that more weights may be pruned, which results in a high model aggregation error, and more iterations are required to train learning models.

B. HFL with Adaptive Model Pruning in Wireless Networks

In this section, the effect of latency constraint in adaptive HFL model pruning and joint design of adaptive model pruning and wireless resource allocation over the dataset Fashion MNIST are simulated.

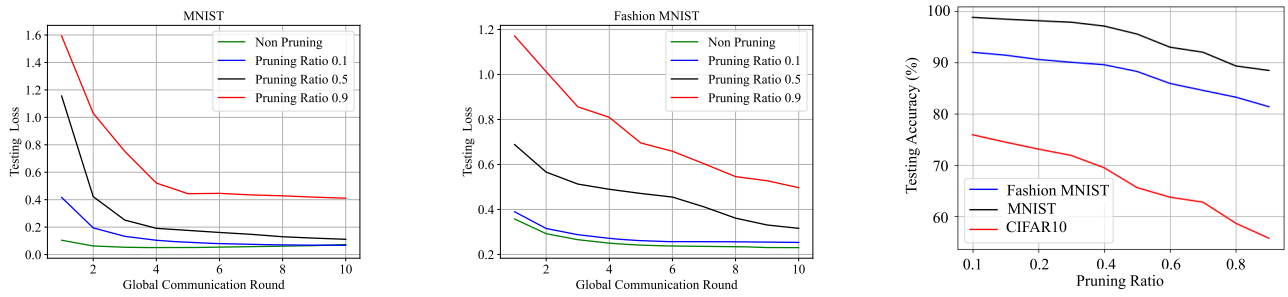


Fig. 2. (a) Testing Loss of importance-based model pruning of HFL with different pruning ratios on MNIST. (b) Testing Loss of importance-based model pruning of HFL with different pruning ratios on Fashion MNIST. (c) Testing accuracy of importance-based model pruning of HFL with different pruning ratios on MNIST, Fashion MNIST, and CIFAR10.

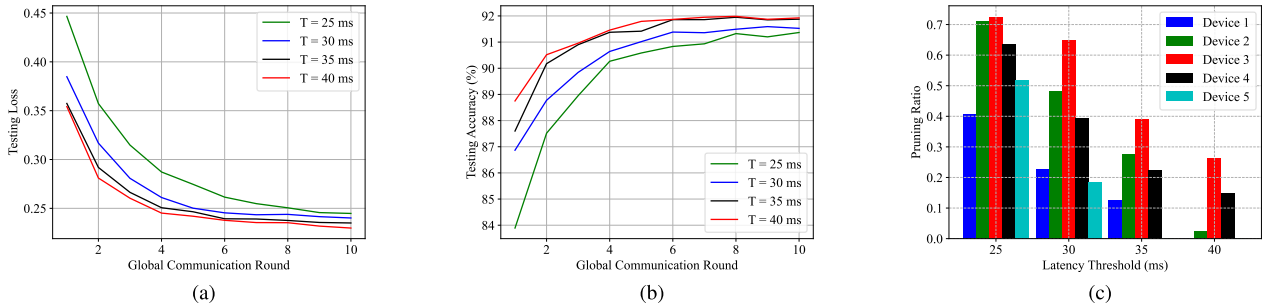


Fig. 3. (a) Testing loss of adaptive HFL model pruning with different latency thresholds on Fashion MNIST. (b) Testing accuracy of adaptive HFL model pruning with different latency thresholds on Fashion MNIST. (c) Pruning ratio required to achieve a given latency threshold on Fashion MNIST.

1) Effect of Latency Constraint: Fig. 3 (a) and Fig. 3 (b) present testing loss and accuracy of adaptive HFL model pruning with different latency constraints on Fashion MNIST, respectively. Fig. 3 (c) plots pruning ratio required for each device in one of edge servers to achieve a given latency threshold on Fashion MNIST. We consider four latency constraints, which are $25ms$, $30ms$, $35ms$, and $40ms$. From the figure, we observe that when the latency constraint increases, the testing loss decreases and the testing accuracy increases. Also, a small number of iterations is required to achieve convergence with a high latency constraint. In addition, the pruning ratio decreases with high latency thresholds. It is because with a large latency constraint, a small pruning ratio is selected by devices. On the contrary, for the device with a small latency constraint, a large pruning ratio is selected to satisfy the latency requirement while sacrificing the learning performance and more iterations are required to achieve convergence. In the following simulation, we assume that the latency constraint is $30ms$.

2) Adaptive Model Pruning and Wireless Resource Allocation: To demonstrate the joint design of adaptive model pruning and wireless resource allocation, we compare the proposed adaptive model pruning with other two baseline schemes. These three schemes are described as follows.

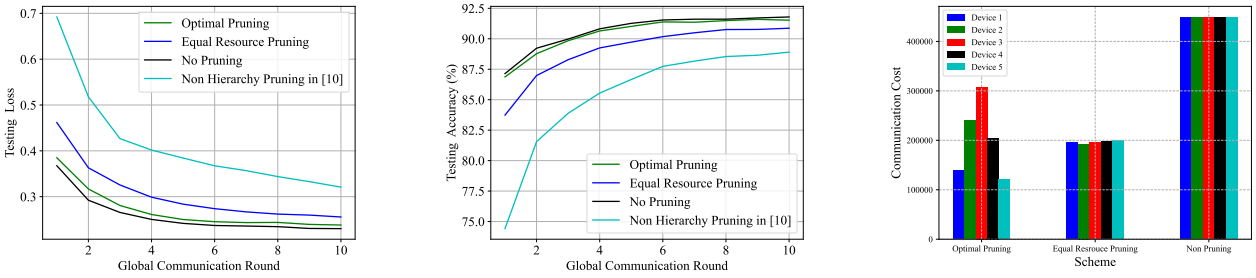
- **Optimal Pruning**: Both the pruning ratio and wireless resource allocation are optimized according to Section V.
- **Equal Resource Pruning**: The pruning ratio is optimized. However, the bandwidth is equally allocated to all devices.
- **No Pruning**: The bandwidth is equally allocated to all devices and model pruning is not deployed.

TABLE III
PER-ROUND COMPUTATION AND COMMUNICATION LATENCY (MS)
$T = T^{\mathrm{CMP}} + T^{\mathrm{UP}}$

Scheme	Optimal Pruning	Equal Resource Pruning	No Pruning
HFL	30	48 ± 0.21	52 ± 0.33
Non HFL	350	590 ± 12.34	750 ± 15.44

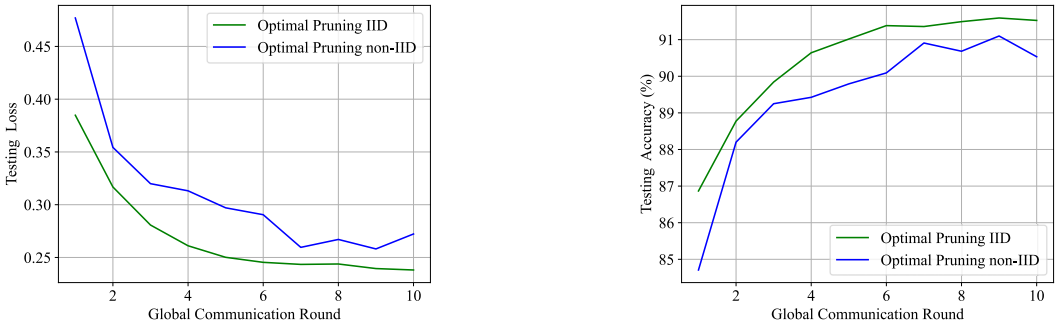
Table II shows the per-round computation and communication latency of HFL and non-HFL schemes. It is observed that the computation and communication latency of Non-HFL is more than 10 times that of HFL. It is because in non-HFL networks, devices need to communication with the cloud server which is far away from them, which results in high transmission latency. Also, we can obtain that the computation and communication latency of optimal pruning is smaller than that of equal resource pruning and no pruning. This is because the optimal pruning ratio is selected based on the given wireless resource, which further decreases the computation and communication latency.

Fig. 4 (a) and Fig. 4 (b) plot the testing loss and accuracy of joint design of adaptive model pruning and wireless resource allocation on Fashion MNIST, respectively. Fig. 4 (c) plots



(a) (b) (c)

Fig. 4. (a) Testing loss of joint design of adaptive model pruning and wireless resource allocation on Fashion MNIST. (b) Testing accuracy of joint design of adaptive model pruning and wireless resource allocation on Fashion MNIST. (c) Communication costs on different schemes.



(a) (b)

Fig. 5. (a) Testing loss of joint design of adaptive model pruning and wireless resource allocation on IID or non-IID Fashion MNIST. (b) Testing accuracy of joint design of adaptive model pruning and wireless resource allocation on IID or non-IID Fashion MNIST.

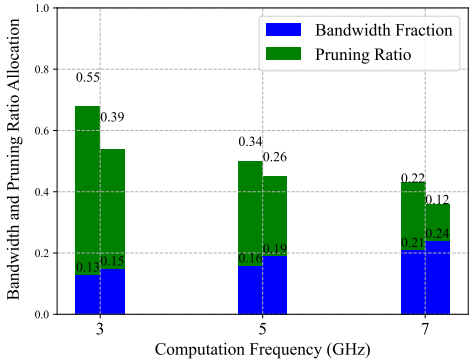


Fig. 6. Relationship among pruning ratio, wireless resource allocation, and computation capability in the proposed optimal pruning scheme.

communication costs on different schemes. The communication cost means the number of model weights needs to be uploaded. It is observed that optimal pruning has the ability to adapt to the wireless resource and the communication cost is much smaller than that of the no pruning scheme. These figures show that the performance of testing loss and accuracy of the proposed adaptive model pruning and wireless resource allocation is close to the no pruning scheme. However, the latency of the proposed algorithm is about 50% less than that of the no pruning scheme. It is because the proposed adaptive pruning scheme has the ability to dynamically remove the unimportant weights according to the wireless channel, which further reduces the latency for both local model updating and uplink transmission, especially when the model size is large. In addition, from Fig. 4 (a) and Fig. 4 (b), we can obtain that the learning accuracy of HFL with the optimal model pruning is much better than that of model pruning in non-hierarchy networks [10]. This is because, in HFL, the participating devices are able to provide massive datasets for model updating, which further improves the learning performance.

Fig. 5 (a) and Fig. 5 (b) plot the testing loss and accuracy of joint design of adaptive model pruning and wireless resource allocation on IID or non-IID Fashion MNIST. It is observed that the testing accuracy and loss of IID data are better than that of non-IID data. It is because in non-IID data, the data class distribution at each device is skewed, which means some data classes are too scarce or even missing. Also, servers need more communication rounds to converge.

Fig. 6 plots the relationship among pruning ratio, wireless resource allocation, and computation capability in the proposed optimal pruning scheme. It is observed that under the same computation capability, when more bandwidth is allocated to the local device, a smaller pruning ratio is adopted to guarantee a high convergence rate. Also, we can obtain that for the local device with a higher computation capability, more wireless resource is allocated to the local device, and a

smaller pruning ratio is selected to guarantee the computation and communication latency and improve the convergence rate.

VII. Conclusions

In this paper, an adaptive model pruning for HFL in wireless networks was developed to reduce the learning network scale. Specifically, the convergence analysis of an upper bound on the l_2-norm of gradients for HFL with model pruning was derived. Then, the pruning ratio and wireless resource allocation were jointly optimized under latency and bandwidth constraints by KKT conditions. Simulation results have shown that our proposed HFL with model pruning achieved similar learning accuracy compared to HFL without pruning and reduced about 50% computation and communication latency. In addition, due to the new challenges brought to dynamic edge-device association, such as high computation complexity and extra communication latency, we would like to address this problem in our future works. To address the data heterogeneity of edge devices, a novel FL framework with partial model pruning and personalization will be considered in our future work. This framework splits the learning model into a global part with model pruning shared with all devices to learn data representations and a personalized part to be fine-tuned for a specific device, which is able to increase the learning accuracy for the device with non-IID data.

Appendix

A. Appendix A - Proof of Theorem 1

We now analyze the convergence of HFL with respect to the pruning ratio ρ and pruning mask $\boldsymbol{m}$. Throughout the proof, we use the following inequalities frequently.

From Jensen's inequality, for any $\boldsymbol{z}_m \in \mathbb{R}^d, m \in \{1,2,...,M\}$, we have

$$\left\| \frac{1}{M} \sum_{m=1}^{M} \boldsymbol{z}_m \right\|^2 \leq \frac{1}{M} \sum_{m=1}^{M} \|\boldsymbol{z}_m\|^2, \tag{36}$$

which directly gives

$$\left\| \sum_{m=1}^{M} \boldsymbol{z}_m \right\|^2 \leq M \sum_{m=1}^{M} \|\boldsymbol{z}_m\|^2. \tag{37}$$

Peter-Paul inequality (also known as Young's inequality) gives

$$\langle \boldsymbol{z}_1, \boldsymbol{z}_2 \rangle \leq \frac{1}{2}\|\boldsymbol{z}_1\|^2 + \frac{1}{2}\|\boldsymbol{z}_2\|^2, \tag{38}$$

and for any constant $s > 0$ and $\boldsymbol{z}_1, \boldsymbol{z}_2 \in \mathbb{R}^d$, we have

$$\|\boldsymbol{z}_1 + \boldsymbol{z}_2\|^2 \leq (1+s)\|\boldsymbol{z}_1\|^2 + \left(1 + \frac{1}{s}\right)\|\boldsymbol{z}_2\|^2. \tag{39}$$

Lemma 2: Under Assumption 2 and 3, for any global and edge communication rounds q and e, we obtain that

$$\sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N} \mathbb{E}\|\boldsymbol{w}_{k,n}^{q,e,t-1} - \boldsymbol{w}_{k,n}^{q,e}\|^2 \leq \eta^2\phi^2 NET^3 + 2TD^2 \sum_{e=1}^{E}\sum_{n=1}^{N} \rho_{n,e}. \tag{40}$$

Proof : In (40), $\boldsymbol{w}_{k,n}^{q,e}$ is the received edge model of the kth edge server from the cloud server at the beginning of the eth edge communication round, and difference $(\boldsymbol{w}_{k,n}^{q,e,t-1} - \boldsymbol{w}_{k,n}^{q,e})$ consists of two parts, namely, variation because of local model training $(\boldsymbol{w}_{k,n}^{q,e,t-1} - \boldsymbol{w}_{k,n}^{q,e,0})$ and variation because of pruning $(\boldsymbol{w}_{k,n}^{q,e,0} - \boldsymbol{w}_{k,n}^{q,e})$. Therefore, (40) is rewritten as

$$\begin{aligned}
&\sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N} \mathbb{E}\|\boldsymbol{w}_{k,n}^{q,e,t-1} - \boldsymbol{w}_{k,n}^{q,e}\|^2 \\
&= \sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N} \mathbb{E}\|(\boldsymbol{w}_{k,n}^{q,e,t-1} - \boldsymbol{w}_{k,n}^{q,e,0}) + (\boldsymbol{w}_{k,n}^{q,e,0} - \boldsymbol{w}_{k,n}^{q,e})\|^2 \\
&\leq \sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N} 2\mathbb{E}\|\boldsymbol{w}_{k,n}^{q,e,t-1} - \boldsymbol{w}_{k,n}^{q,e,0}\|^2 \\
&\quad + \sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N} 2\mathbb{E}\|\boldsymbol{w}_{k,n}^{q,e,0} - \boldsymbol{w}_{k,n}^{q,e}\|^2.
\end{aligned} \tag{41}$$

In (41), $\boldsymbol{w}_{k,n}^{q,e,t-1}$ is updated from $\boldsymbol{w}_{k,n}^{q,e,0}$ by $t-1$ iterations on the nth device. Through the local gradient updating, we obtain that

$$\begin{aligned}
&\sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N} 2\mathbb{E}\|\boldsymbol{w}_{k,n}^{q,e,t-1} - \boldsymbol{w}_{k,n}^{q,e,0}\|^2 \\
&= \sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N} 2\mathbb{E}\left\| \sum_{i=0}^{t-2} -\eta \nabla F_n(\boldsymbol{w}_{k,n}^{q,e,i}, \xi_{k,n}^{q,e,i}) \odot \boldsymbol{m}_{k,n}^{q,e} \right\|^2 \\
&\leq 2\eta^2 \sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N} (t-1) \sum_{i=0}^{t-2} \mathbb{E}\|\nabla F_n(\boldsymbol{w}_{k,n}^{q,e,i}, \xi_{k,n}^{q,e,i}) \odot \boldsymbol{m}_{k,n}^{q,e}\|^2 \\
&\leq 2\eta^2\phi^2 NE \sum_{t=1}^{T} (t-1)^2 = \eta^2\phi^2 NE \frac{2T^3 - 3T^2 + T}{3} \\
&\leq \eta^2\phi^2 NET^3,
\end{aligned} \tag{42}$$

where the third step in (42) is obtained from the bounded gradient in Assumption 3.

Then, $\boldsymbol{w}_{k,n}^{q,e,0} - \boldsymbol{w}_{k,n}^{q,e}$ in (41) is calculated as

$$\begin{aligned}
&\sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N} 2\mathbb{E}\|\boldsymbol{w}_{k,n}^{q,e,0} - \boldsymbol{w}_{k,n}^{q,e}\|^2 \\
&= \sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N} 2\mathbb{E}\|\boldsymbol{w}_{k,n}^{q,e} \odot \boldsymbol{m}_{k,n}^{q,e} - \boldsymbol{w}_{k,n}^{q,e}\|^2 \\
&\leq 2\sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N} \rho_{n,e} D^2 = 2TD^2 \sum_{e=1}^{E}\sum_{n=1}^{N} \rho_{n,e},
\end{aligned} \tag{43}$$

where the second step is obtained from pruning-induced noise in Assumption 2. By plugging (42) and (43) into (41), we obtain the desired result, which ends the proof of Lemma 2.

Lemma 3: Under Assumptions 1-3, for any global and edge communication rounds q and e, we obtain that

$$\mathbb{E}\left\|\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{t=1}^{T}\sum_{n\in\mathcal{N}_k^{q,e,j}}[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})-\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e})]\right\|^2 \leq \frac{\phi^2 NE^2\eta^2L^2T^4+2ET^2L^2D^2\sum_{e=1}^{E}\sum_{n=1}^{N}\rho_{n,e}}{\Gamma^*}, \quad (44)$$

where $\Gamma_k^{q,e,j}=|\mathcal{N}_k^{q,e,j}|$ is the number of local models containing parameters j in the eth edge communication round and $\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})$ is the gradient of the jth weight.

Proof :

$$\begin{aligned}
&\mathbb{E}\left\|\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{t=1}^{T}\sum_{n\in\mathcal{N}_k^{q,e,j}}[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})-\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e})]\right\|^2\\
&\leq ET\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{t=1}^{T}\sum_{n\in\mathcal{N}_k^{q,e,j}}\mathbb{E}\|\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})-\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e})\|^2\\
&\leq \frac{ET}{\Gamma^*}\sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N}\mathbb{E}\|\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})-\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e})\|^2\\
&\leq \frac{ET}{\Gamma^*}\sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N}\mathbb{E}\|\nabla F_n(\boldsymbol{w}_{k,n}^{q,e,t-1})-\nabla F_n(\boldsymbol{w}_{k,n}^{q,e})\|^2\\
&\leq \frac{ET}{\Gamma^*}\sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N}L^2\mathbb{E}\|\boldsymbol{w}_{k,n}^{q,e,t-1}-\boldsymbol{w}_{k,n}^{q,e}\|^2, \quad (45)
\end{aligned}$$

where we relax the inequality by selecting the smallest $\Gamma^*=\min\Gamma_k^{q,e,j}$ and changing the summation over n to all devices in the second step. Then, in the third step, we consider that l_2-gradient norm of a vector is no larger than the sum of norm of all sub-vectors, which allows us to consider ∇F_n rather than its sub-vectors. The last step in (45) is obtained from L-smoothness in Assumption 1, which ends the proof of Lemma 3.

Lemma 4: For IID data distribution under Assumption 4, for any global and communication rounds q and e, we obtain that

$$\mathbb{E}\left\|\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{t=1}^{T}\sum_{n\in\mathcal{N}_k^{q,e,j}}[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1},\xi_{k,n}^{q,e,t-1})-\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})]\right\|^2 \leq \frac{E^2T^2N\hat{\sigma}^2}{\Gamma^*}. \quad (46)$$

Proof :

$$\begin{aligned}
&\mathbb{E}\left\|\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{t=1}^{T}\sum_{n\in\mathcal{N}_k^{q,e,j}}[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1},\xi_{k,n}^{q,e,t-1})-\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})]\right\|^2\\
&\leq \frac{ET}{\Gamma^*}\sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N}\mathbb{E}\|\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1},\xi_{k,n}^{q,e,t-1})-\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})\|^2\\
&\leq \frac{ET}{\Gamma^*}\sum_{e=1}^{E}\sum_{t=1}^{T}\sum_{n=1}^{N}\mathbb{E}\|\nabla F_n(\boldsymbol{w}_{k,n}^{q,e,t-1},\xi_{k,n}^{q,e,t-1})-\nabla F_n(\boldsymbol{w}_{k,n}^{q,e,t-1})\|^2\leq\frac{E^2T^2N\hat{\sigma}^2}{\Gamma^*}. \quad (47)
\end{aligned}$$

In the second step, we consider that l_2-gradient norm of a vector is no larger than the sum of norm of all sub-vectors, which allows us to consider ∇F_n rather than its sub-vectors. The last step in (47) is obtained from gradient noise for IID data in Assumption 4, which ends the proof of Lemma 4.

Lemma 5: The upperbound of $\mathbb{E}\|\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_G^q\|^2$ is denoted as

$$\begin{aligned}
\mathbb{E}\|\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_G^q\|^2 &\leq 3\eta^2W^2E^2T^2\phi^2\\
&+\frac{3W^2\eta^2E^2T^2N\hat{\sigma}^2+3W^2E^2L^2T^4\eta^4N\phi^2}{\Gamma^*}\\
&+\frac{6W^2\eta^2L^2D^2T^2E\sum_{e=1}^{E}\sum_{n=1}^{N}\rho_{n,e}}{\Gamma^*}, \quad (48)
\end{aligned}$$

where W is the number of model weights.

Proof :

$$\begin{aligned}
&\mathbb{E}\|\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_G^q\|^2\\
&=\mathbb{E}\left\|\frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}}\sum_{e=1}^{E}\sum_{j=1}^{W}\frac{1}{\Gamma_k^{q,e,j}}\sum_{n\in\mathcal{N}_k^{q,e,j}}\sum_{t=1}^{T}\eta\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1},\xi_{k,n}^{q,e,t-1})\right\|^2\\
&\leq\frac{3W}{|\mathcal{K}|}\sum_{k\in\mathcal{K}}\sum_{j=1}^{W}\mathbb{E}\left\|\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{n\in\mathcal{N}_k^{q,e,j}}\sum_{t=1}^{T}\eta[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1},\xi_{k,n}^{q,e,t-1})-\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})]\right\|^2\\
&+\frac{3W}{|\mathcal{K}|}\sum_{k\in\mathcal{K}}\sum_{j=1}^{W}\mathbb{E}\left\|\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{n\in\mathcal{N}_k^{q,e,j}}\sum_{t=1}^{T}\eta[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})-\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e})]\right\|^2\\
&+\frac{3W}{|\mathcal{K}|}\sum_{k\in\mathcal{K}}\sum_{j=1}^{W}\mathbb{E}\left\|\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{n\in\mathcal{N}_k^{q,e,j}}\sum_{t=1}^{T}\eta[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e})]\right\|^2, \quad (49)
\end{aligned}$$

where we split stochastic gradient $\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1},\xi_{k,n}^{q,e,t-1})$ into three parts, namely, $[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e})]$,

$[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1},\xi_{k,n}^{q,e,t-1}) - \nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})]$, and $[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1}) - \nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e})]$.

The third term of the last step in (49) is derived as

$$\begin{aligned}
&\frac{3W}{|\mathcal{K}|}\sum_{k\in\mathcal{K}}\sum_{j=1}^{W}\mathbb{E}\left\|\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{n\in\mathcal{N}_k^{q,e,j}}\sum_{t=1}^{T}\eta[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e})]\right\|^2\\
&\leq \frac{3\eta^2WTE}{|\mathcal{K}|}\sum_{k\in\mathcal{K}}\sum_{j=1}^{W}\sum_{e=1}^{E}\sum_{t=1}^{T}\mathbb{E}\|\nabla F_n(\boldsymbol{w}_{k,n}^{q,e})\|^2\\
&\leq 3\eta^2W^2E^2T^2G^2. \qquad (50)
\end{aligned}$$

Through plugging (44), (46), and (50) into (49), the upperbound of $\mathbb{E}\|\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_G^q\|^2$ is derived as (48), which ends the proof of Lemma 5.

Proof of the Convergence: Based on Lemma 2, 3, 4, and 5, we use L-smoothness in Assumption 1 to give convergence analysis. We begin with

$$F(\boldsymbol{w}_G^{q+1})\leq F(\boldsymbol{w}_G^q)+\langle\nabla F(\boldsymbol{w}_G^q),\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_G^q\rangle+\frac{L}{2}\|\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_G^q\|^2. \qquad (51)$$

Then, by taking expectations on both sides of (51), we obtain

$$\begin{aligned}
&\mathbb{E}[F(\boldsymbol{w}_G^{q+1})]-\mathbb{E}[F(\boldsymbol{w}_G^q)]\\
&\leq \mathbb{E}\langle\nabla F(\boldsymbol{w}_G^q),\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_G^q\rangle+\frac{L}{2}\mathbb{E}\|\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_G^q\|^2. \qquad (52)
\end{aligned}$$

First, we analyze $\mathbb{E}\langle\nabla F(\boldsymbol{w}_G^q),\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_G^q\rangle$ by considering a sum of inner products over all model weights, which is denoted as

$$\begin{aligned}
&\mathbb{E}\langle\nabla F(\boldsymbol{w}_G^q),\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_q\rangle=\sum_{j=1}^{W}\mathbb{E}\langle\nabla F^j(\boldsymbol{w}_G^q),\boldsymbol{w}_G^{q+1,j}-\boldsymbol{w}_G^{q,j}\rangle\\
&=\sum_{j=1}^{W}\mathbb{E}\Bigg\langle\nabla F^j(\boldsymbol{w}_G^q),\\
&-\frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}}\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{n\in\mathcal{N}_k^{q,e,j}}\sum_{t=1}^{T}\eta\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})\Bigg\rangle\\
&=-\sum_{j=1}^{W}\mathbb{E}\langle\nabla F^j(\boldsymbol{w}_G^q),\eta ET\nabla F^j(\boldsymbol{w}_G^q)\rangle\\
&-\sum_{j=1}^{W}\mathbb{E}\Bigg\langle\nabla F^j(\boldsymbol{w}_G^q),\\
&\frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}}\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{n\in\mathcal{N}_k^{q,e,j}}\sum_{t=1}^{T}\eta\Big[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})-\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e})\Big]\Bigg\rangle, \qquad (53)
\end{aligned}$$

where the last step splits the result into two parts with respect to a reference point $\eta ET\nabla F^j(\boldsymbol{w}_{k,n}^{q,e})$. For the first term in the last step of (53), it is derived as

$$\begin{aligned}
&-\sum_{j=1}^{W}\mathbb{E}\langle\nabla F^j(\boldsymbol{w}_G^q),\eta ET\nabla F^j(\boldsymbol{w}_G^q)\rangle\\
&=-\eta ET\sum_{j=1}^{W}\left\|\nabla F^j(\boldsymbol{w}_G^q)\right\|^2. \qquad (54)
\end{aligned}$$

For the second term in the last step of (53), it is derived as

$$\begin{aligned}
&-\sum_{j=1}^{W}\mathbb{E}\Bigg\langle\nabla F^j(\boldsymbol{w}_G^q),\\
&\frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}}\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{n\in\mathcal{N}_k^{q,e,j}}\sum_{t=1}^{T}\eta\Big[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})-\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e})\Big]\Bigg\rangle\\
&=-\sum_{j=1}^{W}\eta ET\mathbb{E}\Bigg\langle\nabla F^j(\boldsymbol{w}_G^q),\\
&\frac{1}{|\mathcal{K}|ET}\sum_{k\in\mathcal{K}}\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{n\in\mathcal{N}_k^{q,e,j}}\sum_{t=1}^{T}\Big[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})-\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e})\Big]\Bigg\rangle\\
&\leq\frac{\eta ET}{2}\sum_{j=1}^{W}\mathbb{E}\|\nabla F^j(\boldsymbol{w}_G^q)\|^2+\\
&\frac{\eta\sum_{k\in\mathcal{K}}\sum_{j=1}^{W}\mathbb{E}\left\|\sum_{e=1}^{E}\frac{1}{\Gamma_k^{q,e,j}}\sum_{n\in\mathcal{N}_k^{q,e,j}}\sum_{t=1}^{T}\Big[\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e,t-1})-\nabla F_n^j(\boldsymbol{w}_{k,n}^{q,e})\Big]\right\|^2}{2ET|\mathcal{K}|}\\
&\leq\frac{\eta ET}{2}\sum_{j=1}^{W}\mathbb{E}\|\nabla F^j(\boldsymbol{w}_G^q)\|^2\\
&+\frac{W\phi^2NE\eta^3L^2T^3+2W\eta TL^2D^2\sum_{e=1}^{E}\sum_{n=1}^{N}\rho_{n,e}}{2\Gamma^*}, \qquad (55)
\end{aligned}$$

where the second step is obtained from (38) and the last step is obtained from Lemma 3.

By plugging (54) and (55) into (53), $\mathbb{E}\langle\nabla F(\boldsymbol{w}_G^q),\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_G^q\rangle$ is derived as

$$\begin{aligned}
&\mathbb{E}\langle\nabla F(\boldsymbol{w}_G^q),\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_G^q\rangle\leq-\frac{\eta ET}{2}\sum_{j=1}^{W}\left\|\nabla F^j(\boldsymbol{w}_G^q)\right\|^2\\
&+\frac{W\phi^2NE\eta^3L^2T^3+2W\eta TL^2D^2\sum_{e=1}^{E}\sum_{n=1}^{N}\rho_{n,e}}{2\Gamma^*}. \qquad (56)
\end{aligned}$$

Finally, we plug the upperbound of $\mathbb{E}\|\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_G^q\|^2$ into (52) to obtain the convergence upperbound. First, we take the sum over global communication round $q=1,2,...,Q$ on both sides of (52) and obtain that

$$\begin{aligned}
&\mathbb{E}[F(\boldsymbol{w}^0)]-\mathbb{E}[F(\boldsymbol{w}^*)]=\sum_{q=1}^{Q}\mathbb{E}[F(\boldsymbol{w}_G^{q+1})]-\sum_{q=1}^{Q}\mathbb{E}[F(\boldsymbol{w}_G^q)]\\
&\leq-\sum_{q=1}^{Q}\sum_{j=1}^{W}\frac{\eta ET}{2}\left\|\nabla F^j(\boldsymbol{w}_G^q)\right\|^2+\sum_{q=1}^{Q}\frac{L}{2}\mathbb{E}\|\boldsymbol{w}_G^{q+1}-\boldsymbol{w}_G^q\|^2\\
&+\sum_{q=1}^{Q}\frac{W\phi^2NE\eta^3L^2T^3+2W\eta TL^2D^2\sum_{e=1}^{E}\sum_{n=1}^{N}\rho_{n,e}}{2\Gamma^*}. \qquad (57)
\end{aligned}$$

By plugging (48) into (57), we obtain

$$\frac{\eta ET}{2}\sum_{j=1}^{W}\sum_{q=1}^{Q}\left\|\nabla F^j(\boldsymbol{w}_G^q)\right\|^2 \le \mathbb{E}[F(\boldsymbol{w}^0)]-\mathbb{E}[F(\boldsymbol{w}^*)] + \frac{3QLW^2\eta^2E^2T^2\phi^2}{2} + \frac{(3QW^2\eta^2L^3T^2D^2E+QW\eta TL^2D^2)\sum_{e=1}^{E}\sum_{n=1}^{N}\rho_{n,e}}{\Gamma^*} + \frac{3QW^2E^2L^3T^4\eta^4\phi^2N+W\phi^2NEQ\eta^3L^3T^3}{2\Gamma^*} + \frac{3QLW^2\eta^2E^2T^2N\hat{\sigma}^2}{2\Gamma^*}, \tag{58}$$

which completes the proof of Theorem 1.

B. Appendix B - Proof of Theorem 2

According to (6) and (30), pruning ratio $\rho_{n,e}$ is calculated as

$$(W_{n,\text{conv}}+(1-\rho_{n,e})W_{n,\text{fully}})\left(\frac{TC_n}{f_n}+\frac{\hat{q}}{R^{\text{up}}_{n,k,e}}\right)\le T_{\text{th}}, \tag{59}$$

$$T^{\text{cmp-conv}}_{n,e}+T^{\text{com-conv}}_{n,k,e}+(1-\rho_{n,e})(T^{\text{cmp-fully}}_{n,e}+T^{\text{com-fully}}_{n,k,e})\le T_{\text{th}}, \tag{60}$$

where $T^{\text{cmp-conv}}_{n,e}=W_{n,\text{conv}}TC_n/f_n$, $T^{\text{com-conv}}_{n,k,e}=W_{n,\text{conv}}\hat{q}/R^{\text{up}}_{n,k,q}$, $T^{\text{cmp-fully}}_{n,e}=W_{n,\text{fully}}TC_n/f_n$, and $T^{\text{com-fully}}_{n,k,e}=W_{n,\text{com-fully}}\hat{q}/R^{\text{up}}_{n,k,e}$. Then, $\rho_{n,e}$ is deduced as (31), which ends the proof of Theorem 2.

C. Appendix C - Proof of Lemma 1

The objective function in (33) is equal to

$$F(X)=\sum_{e=1}^{E}\sum_{n=1}^{N}f(x_{n,e})=\sum_{e=1}^{E}\sum_{n=1}^{N}\left(1-\frac{x_{n,e}V_1-V_2}{x_{n,e}V_3+V_4}\right), \tag{61}$$

where $V_1,V_2,V_3,V_4>0$, and $0\le x_{n,e}\le 1$. To prove the lemma, we just need to analyze the convexity of the function $f(x_{n,e})$. The first derivative is derived as

$$f^{'}(x_{n,e})=-\frac{V_1V_4+V_2V_3}{(V_3x_{n,e}+V_4)^2}. \tag{62}$$

Then, the second derivative is calculated as

$$f^{''}(x_{n,e})=\frac{2V_3(V_1V_4+V_2V_3)(V_3x_{n,e}+V_4)}{(V_3x_{n,e}+V_4)^4}\ge 0. \tag{63}$$

Therefore, the objective function in (33) is convex. Also, both constraints in (27) and (28) are convex. As a result, the optimization problem in (33) in convex, which ends the proof of Lemma 1.

D. Appendix D - Proof of Theorem 3

Based on the optimization in (33) and constraint (27), the Lagrange function is denoted as

$$\mathcal{L}(b_{n,e},\lambda)=H_2\sum_{e=1}^{E}\sum_{n=1}^{N}\left(1-\frac{R^{\text{up}}_{n,k,e}(T_{\text{th}}-T^{\text{cmp-conv}}_{n,e})-\hat{q}W_{n,\text{conv}}}{R^{\text{up}}_{n,k,e}T^{\text{cmp-fully}}_{n,e}+\hat{q}W_{n,\text{fully}}}\right) + \lambda\left(\sum_{n=1}^{N}b_{n,e}-1\right), \tag{64}$$

where λ is the Lagrange multiplier. Then, the Karush-Kuhn-Tucker (KKT) conditions are deduced as

$$\frac{\partial\mathcal{L}}{\partial b_{n,e}}=\lambda-\frac{(V_1V_4+V_2V_3)B\log_2\left(1+\frac{g^e_{n,k}p_n}{\sigma^2}\right)}{\left(b_{n,e}BV_3\log_2\left(1+\frac{g^e_{n,k}p_n}{\sigma^2}\right)+V_4\right)^2}=0, \tag{65}$$

$$\lambda\left(\sum_{n\in\mathcal{N}_k}b_{n,e}-1\right)=0, \tag{66}$$

$$\lambda\ge 0. \tag{67}$$

Based on the KKT conditions, the optimal bandwidth allocation is achieved as Theorem 3, which ends the proof of Theorem 3.

REFERENCES

[1] B. Custers, A. Sears, F. Dechesne, I. Georgieva, T. Tani, and S. van der Hof, "Eu personal data protection in policy and practice," *Hague, The Netherlands: TMC Asser Press*, 2019.
[2] B. M. Gaff, H. E. Sussman, and J. Geetter, "Privacy and big data," *Computer*, vol. 47, no. 6, pp. 7 – 9, 2014.
[3] B. P. Rimal, E. Choi, and I. Lumb, "A taxonomy and survey of cloud computing systems," *2009 Fifth International Joint Conference on INC, IMS and IDC*, pp. 44 – 51, Aug. 2009.
[4] P. Li, J. Li, Z. Huang, T. Li, C. Gao, S. Yiu, and K. Chen, "Multi-key privacy-preserving deep learning in cloud computing," *Future Gener. Comput. Syst.*, vol. 74, pp. 76 – 85, Sept. 2017.
[5] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," *In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pp. 1273 – 1282, 2017.
[6] D. Saxena, R. Gupta, and A. K. Singh, "A survey and comparative study on multi-cloud architectures: Emerging issues and challenges for cloud federation," *arxiv:2108.12831*, 2021.
[7] S. Luo, X. Chen, Q. Wu, Z. Zhou, and S. Yu, "Hfel: Joint edge association and resource allocation for cost-efficient hierarchical federated edge learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 10, pp. 6535 – 6548, Oct. 2020.
[8] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y. C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 2031 – 2063, 3rd Quart. 2020.
[9] B. Kar, W. Yahya, Y. Lin, and A. Ali, "Offloading using traditional optimization and machine learning in federated cloud-edge-fog systems: A survey," *IEEE Commun. Surveys Tuts.*, pp. 1 – 1, Early Access 2023.
[10] S. Liu, G. Yu, R. Yin, J. Yuan, L. Shen, and C. Liu, "Joint model pruning and device selection for communication-efficient federated edge learning," *IEEE Trans. Commun.*, vol. 70, no. 1, pp. 231–244, Jan. 2022.
[11] Y. Jiang, S. Wang, V. Valls, B. J. Ko, W.-H. Lee, K. K. Leung, and L. Tassiulas, "Model pruning enables efficient federated learning on edge devices," *IEEE Trans. Neural Netw. Learn Syst.*, pp. 1–13, 2022.
[12] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.
[13] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Advances in neural information processing systems*, 1990, pp. 598–605.

[14] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," vol. 28, 2015.

[15] H. Tanaka, D. Kunin, D. L. Yamins, and S. Ganguli, "Pruning neural networks without any data by iteratively conserving synaptic flow," vol. 33, pp. 6377–6389, 2020.

[16] N. Lee, T. Ajanthan, and P. Torr, "SNIP: Single-shot network pruning based on connection sensitivity," in *International Conference on Learning Representations*, 2019.

[17] 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. (CVPR)*, pp. 11 264 – 11 272, Jun. 2019.

[18] M. Shen, P. Molchanov, H. Yin, and J. M. Alvarez, "When to prune? a policy towards early structural pruning," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 12 237–12 246.

[19] T. Lin, S. U. Stich, L. Barba, D. Dmitriev, and M. Jaggi, "Dynamic model pruning with feedback," in *International Conference on Learning Representations*, 2020.

[20] D. Wen, K.-J. Jeon, and K. Huang, "Federated dropout—a simple approach for enabling federated learning on resource constrained devices," *IEEE Wireless Commun. Lett.*, vol. 11, no. 5, pp. 923–927, May 2022.

[21] N. Bouacida, J. Hou, H. Zang, and X. Liu, "Adaptive federated dropout: Improving communication efficiency and generalization for federated learning," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2021, pp. 1–6.

[22] G. Cheng, Z. Charles, Z. Garrett, and K. Rush, "Does federated dropout actually work?" in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2022, pp. 3387–3395.

[23] S. Horváth, S. Laskaridis, M. Almeida, I. Leontiadis, S. Venieris, and N. Lane, "Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout," in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 12 876–12 889.

[24] X. Zhang, Y. Liu, X. Liu, A. Argyriou, and Y. Han, "D2D-assisted federated learning in mobile edge computing networks," *in IEEE WCNC*, May 2021.

[25] T. T. Phuong and L. T. Phong, "Decentralized descent optimization with stochastic gradient signs for device-to-device networks," *IEEE Wireless Commun. Lett.*, vol. 10, no. 9, pp. 1939 – 1943, Sept. 2021.

[26] J. Yuan, M. Xu, X. Ma, A. Zhou, X. Liu, and S. Wang, "Hierarchical federated learning through LAN-WAN orchestration," *arxiv:2010.11612*, 2020.

[27] J. Ren, W. Ni, G. Nie, and H. Tian, "Research on resource allocation for efficient federated learning," *arxiv:2104.09177*, 2021.

[28] M. S. AI-Abiad, M. Z. Hassan, and M. J. Hossain, "Energy efficient federated learning in integrated fog-cloud computing enabled internet-of-things networks," *arxiv:2107.03520*, 2021.

[29] W. Lim, J. S. Ng, Z. Xiong, D. Niyato, C. Miao, and D. I. Kim, "Dynamic edge association and resource allocation in self-organizing hierarchical federated learning networks," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3640 – 3653, Dec. 2021.

[30] R. Saha, S. Misra, and P. K. Deb, "FogFL: Fog-assisted federated learning for resource-constrained IoT devices," *IEEE Internet Things J.*, vol. 8, no. 18, pp. 8456 – 8463, May. 2021.

[31] W. Wu, L. He, W. Lin, and R. Mao, "Accelerating federated learning over reliability-agnostic clients in mobile edge computing systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 7, pp. 1539 – 1551, Jul. 2021.

[32] T. Q. Dinh, D. N. Nguyen, D. T. Hoang, T. V. Pham, and E. Dutkiewic, "In-network computation for large-scale federated learning over wireless edge networks," *IEEE Trans. Mobile Comput.*, pp. 1 – 15, Early Access 2022.

[33] S. Horvath, S. Laskaridis, M. Almeida, I. Leontiadis, S. I. Venieris, and N. D. Lane, "Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout," *in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS'21)*, 2021.

[34] D. Wen, M. Bennis, and K. Huang, "Joint parameter-and-bandwidth allocation for improving the efficiency of partitioned edge learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 12, pp. 8272–8286, Dec. 2020.

[35] B. Hawks, J. Duarte, N. J. Fraser, A. Pappalardo, N. Tran, and Y. Umuroglu, "Ps and Qs: Quantization-aware pruning for efficient low latency neural network inference," *arxiv:2102.11289*, 2021.

[36] S. Ghadimi and G. H. Lan, "Stochastic first-and zeroth-order methods for nonconvex stochastic programming," *SIAM J. Optim.*, vol. 23, no. 4, pp. 2341 – 2368, 2013.

[37] S. Shi, K. Zhao, Q. Wang, Z. Tang, and X. Chu, "A convergence analysis of distributed sgd with communication-efficient gradient sparsification," *in Proc. 28th Int. Joint Conf. Artif. Intell.*, pp. 3411 – 3417, Aug. 2019.

[38] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Hierarchical federated learning with quantization: Convergence analysis and system design," *IEEE Trans. Wireless Commun.*, vol. 22, no. 1, pp. 2 – 18, Jan. 2023.

[39] F. Zhou and G. Gong, "A distributed hierarchical SGD algorithm with sparse global reduction," *arxiv:1903.05133*, 2019.

[40] J. Wang, S. Wang, R. Chen, and M. Ji, "Demystifying why local aggregation helps: Convergence analysis of hierarchical SGD," *in Proc. 36th Conf. Artif. Intell. (AAAI'36)*, pp. 8648 – 8556, Jun. 2022.

[41] C. Feng, H. H. Yang, D. Hu, Z. Zhao, T. Q. S. Quek, and G. Min, "Mobility-aware cluster federated learning in hierarchical wireless networks," *IEEE Trans. Wireless Commun.*, vol. 21, no. 10, pp. 8441 – 8458, Oct. 2022.

[42] S. U. Stich, J. B. Cordonnier, and M. Jaggi, "Sparsified sgd with memory," *in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS'18)*, pp. 4447 – 4458, Dec. 2018.

[43] P. L. Bartlett, "The sample complexity of pattern classification with neural networks: The size of the weights is more important than the size of the network," *IEEE Trans. Inf. Theory*, vol. 44, no. 2, pp. 525 – 536, Mar. 1998.

[44] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," *in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS'16)*, pp. 901 – 909, Dec. 2016.

[45] Z. Chen, W. Yi, Y. Liu, and A. Nallanathan, "Knowledge-aided federated learning for energy-limited wireless networks," *IEEE Trans. Commun.*, pp. 1–1, 2023.