FedGreen: Federated Learning with Fine-Grained Gradient Compression for Green Mobile Edge Computing

Peichun Li*, Xumin Huang*, Miao Pan[†] and Rong Yu*

*School of Automation, Guangdong University of Technology, Guangzhou, China [†]Department of Electrical and Computer Engineering, University of Houston, Houston, USA Email: peichun@mail2.gdut.edu.cn, huangxu_min@163.com, mpan2@uh.edu, yurong@ieee.org

Abstract—Federated learning (FL) enables devices in mobile edge computing (MEC) to collaboratively train a shared model without revealing the local data. Gradient compression could be applied to FL to alleviate the communication overheads but the existing schemes still face challenges. To deploy green MEC, we propose FedGreen, which enhances the original FL with fine-grained gradient compression to control the total energy consumption of the devices. Specifically, we introduce the relevant operations including device-side gradient reduction and server-side element-wise aggregation to facilitate the gradient compression in FL. According to a public dataset, we evaluate the contributions of the compressed local gradients with respect to different compression ratios. Furthermore, we investigate a learning accuracy-energy efficiency tradeoff problem and the optimal compression ratio and computing frequency are derived for each device. Experimental results show that given the 80% test accuracy requirement, compared with the baseline schemes, FedGreen reduces at least 32% of the total energy consumption of the devices.

Index Terms—Federated learning, gradient compression, mobile edge computing, resource management

I. INTRODUCTION

Federated learning (FL) is a promising distributed machine learning framework that enables multiple devices to jointly train a shared model by their private datasets while preserving the training data privacy [1]. In FL, a parameter server with the central position distributes an initialized learning model to the devices. Each device trains the model by the local dataset and submits the local gradients to the parameter server. All local gradients are aggregated to update the global model. Then the updated global model is sent to each device to perform a new local model training task. The iterative training procedure is repeated until convergence. Recently, the emerged computing paradigm named mobile edge computing (MEC) is applied to facilitate the execution of FL [2]. Massive edge devices in MEC posse versatile sensors to collect raw data and have under-utilized resources to execute the FL algorithm. Many research efforts have been devoted to optimizing the performance of FL in MEC.

Researchers have integrated gradient compression into FL to compress the local gradients of the devices and decrease the number of bits transmitted to the parameter server. For example, some less important local gradients were clipped based on the magnitude [3] and let a small number of bits

represent the gradient values [4]. Similarly, a universal vector quantization scheme was studied in [5]. But these methods neglected that different devices have different channel states, computing capabilities and energy consumption rates such that they could require different compression ratios to match with their energy states. In addition to the uniform gradient compression, device scheduling is introduced to provide unified management for all devices according to diverse optimization goals [6]–[8]. The methods select specific devices to perform local training tasks and this could accelerate the training procedure of FL to a degree. But the methods directly limit the amount of training data and cause the unbalanced usage of all devices' data.

Toward green deployment of MEC, FL with gradient compression still faces great challenges. To execute the FL algorithm, local model training requires each device to consume a certain number of computation resources [9], [10]. At the same time, wireless bandwidth is necessitated since learning in a decentralized manner takes hundreds of communication rounds until convergence. But devices in MEC are generally battery-limited. For a green MEC system, the total energy consumption of the devices should be controlled to create energy savings and avoid battery degradation. In turn, a variety of devices in MEC may have heterogeneous resources in terms of computation, communication, and power [11]. In FL with gradient compression, the computing frequency and compression ratio of each device should be optimized to match the hardware configuration and channel status.

To promote the FL with gradient compression, we adopt different compression ratios for different devices in MEC and study a learning accuracy-energy efficiency tradeoff problem. We present a comprehensive scheme called by FedGreen, which enhances the original FL with fine-grained gradient compression to achieve green MEC. Specifically, we first present a basic method that enables each device to choose a specific ratio to compress the local gradient after local model training. As a consequence, a device can switch to a small compression ratio to report more accurate gradient information in the resource-sufficient state, and a large one to decrease the communication overheads and save energy in the resource-deficient state. We further study how to derive an acceptable compression ratio for each device. According to a public dataset, the quantitative relationship between gradient compression ratio and global model accuracy is formulated. To balance the accuracy performance and total energy consumption of the devices, we investigate the learning accuracy-energy efficiency tradeoff problem for FL with gradient compression, and jointly optimize the compression ratios and computing frequency of the devices. Extensive experimental results are provided to validate the efficiency and effectiveness of the proposed scheme.

The main contributions of the paper are summarized as follows.

- We design a fine-grained gradient compression method by combining device-side gradient reduction and server-side element-wise aggregation. Based on current techniques, FedGreen enables different devices to compress the local gradients on demand, according to the energy states.
- We present a learning accuracy-energy efficiency tradeoff problem for FL with gradient compression. The compression ratio and computing frequency of each device are jointly optimized to ensure the algorithm performance of FL while reducing the total energy consumption.
- We conduct experiments to validate the overall performance of FedGreen. Compared with the baseline schemes of FL, FedGreen saves energy on the devices and achieves fine-grained gradient compression for green MEC.

The rest of this paper is organized as follows. We describe the fine-grained gradient compression method in Section II. Section III discusses the learning accuracy-energy efficiency tradeoff problem and its theoretical analysis. Experiment evaluation of our framework is shown in Section IV. Finally, Section V concludes this paper.

II. FINE-GRAINED GRADIENT COMPRESSION

A. Device-side Gradient Compression

Without loss of generality, we take the two-dimension convolution layer as an example, and consider the layerwise gradient compression. The three-stage gradient compression consists of sparsification, quantization and encoding. Let $v \in \mathbb{R}^N$ denote the original gradients before compression and $N = C_{out} \times C_{in} \times K \times K$, where C_{out} , C_{in} and K are the #output channels, #input channels and kernel size, respectively. Here, we use 32 bits to represent a float number.

Kernel-wise gradient sparsification. We define the *kernel* with shape of $K \times K$ as the basic unit of the gradient sparsification. As shown in the left of Fig. 1, we calculate the L2 norm of each kernel in one layer. Given a pruning rate of $\rho \in [0, 1)$, we zero-out the first $\lfloor \rho C_{out} C_{in} \rfloor$ kernels with smallest norm. Let $v_s = v \odot m$, $v_s \in \mathbb{R}^N$ represent the sparse gradient after pruning, where m is the binary mask with shape of $C_{out} \times C_{in}$ and \odot is the Hadamard product with broadcasting. Furthermore, let \hat{v} represent the non-zero entries in v_s , and $\hat{v} \in \mathbb{R}^M$, where $M = \lceil (1 - \rho)C_{out}C_{in} \rceil K^2$. Given a mask m and the non-zero gradient \hat{v} , we can obtain



Fig. 1. Gradient sparsification and quantization.

 v_s by $v_s = \mathcal{R}(\hat{v}, m)$, where \mathcal{R} is the function that reconstructs the sparse gradient from the dense one with respect to m. Compared with the traditional filter-wise method that zeroes out a whole filter to compress the gradient, kernel-wise sparsification achieves fine-grained pruning while maintaining a small mask size.

Lemma 1. For any sparse gradient \boldsymbol{v}_s obtained from $\boldsymbol{v} \in \mathbb{R}^N$ and $\rho \in [0, 1)$ by kernel-wise sparsification, we have $\|\boldsymbol{v} - \boldsymbol{v}_s\|_2^2 \leq \rho \|\boldsymbol{v}\|_2^2$.

Stochastic gradient quantization. Motivated by QSGD in [12], we propose a reinforced stochastic quantization scheme for the pruned gradient \hat{v} . Let $|\hat{v}|_{\min}$ and $|\hat{v}|_{\max}$ be the minimum and maximum value of $|\hat{v}|$, and $\Delta = |\hat{v}|_{\max} - |\hat{v}|_{\min}$. Let j index the entries in \hat{v} , and L be the number of quantization levels. We can quantize any non-zero scalar $\hat{v}^j \in \hat{v}$ by

$$\mathcal{Q}(\hat{v}_j, L) = \operatorname{sgn}(\hat{v}^j) \cdot \left[\delta(\left| \hat{v}^j \right| - \left| \hat{v} \right|_{\min}; L) + \left| \hat{v} \right|_{\min} \right], \quad (1)$$

where $\operatorname{sgn}(\hat{v}^j) \in \{-1, +1\}$ denote the sign of \hat{v}^j , and $\tilde{x} = \delta(x; L)$ is the stochastic quantization function that maps $x \in [0, \Delta]$ to $\tilde{x} \in \{0, \frac{\Delta}{L-1}, \frac{2\Delta}{L-1}, \cdots, \Delta\}$. Let $l \in \{0, 1, \cdots, L-1\}$ be an integer such that $x \in [\frac{\Delta l}{L}, \frac{\Delta(l+1)}{L}]$. Hence, $[\frac{\Delta l}{L}, \frac{\Delta(l+1)}{L}]$ is the quantization interval of x. Then, we have

$$\delta(x;L) = \begin{cases} \Delta(l+1)/L & \text{with probability } xL/\Delta - l \\ \Delta l/L & \text{otherwise.} \end{cases}$$
(2)

After applying $\mathcal{Q}(\hat{v}^j, L)$ for all $\hat{v}^j \in \hat{v}$, we obtain the quantizated gradient \tilde{v} . Naturally, \tilde{v} can be represented by a tuple $(\tilde{v}', \operatorname{sgn}(\hat{v}), |\hat{v}|_{\min}, |\hat{v}|_{\max})$, where \tilde{v}' is the index gradient with each entry of $\log_2 L$ bits. Given a mask m and its quantized gradient \tilde{v} , we can obtain the sparse form of quantized gradient $v_q = \mathcal{R}(\tilde{v}, m)$. An example of quantization process is provided in right of Fig. 1.

Lemma 2. For any quantizated gradient $v_q \in \mathbb{R}^N$ computed from pruned gradient $v_s = \mathcal{R}(\hat{v}, m), \hat{v} \in \mathbb{R}^M$ by the above scheme with L levels, we have $||v_q - v_s||_2 \leq M\Delta/(L-1)$.

Lossless encoding. Now, we obtain a binary mask m and a tuple $(\tilde{v}', \operatorname{sgn}(\hat{v}), |\hat{v}|_{\min}, |\hat{v}|_{\max})$. Since m is sparse, we utilize compressed sparse row (CSR) format to represent m and obtain $\mathcal{E}_{csr}(m)$. Furthermore, due to the statistical characteristics of \tilde{v}' that smaller indices are more frequent, we apply Huffman coding to reduce the data size and get $\mathcal{E}_{H}(\tilde{v}')$. Finally, we get an encoded tuple with five parts $(\mathcal{E}_{csr}(m), \mathcal{E}_{H}(\tilde{v}'), \operatorname{sgn}(\hat{v}), |\hat{v}|_{\min}, |\hat{v}|_{\max})$.

Lemma 3. Given any convolution gradient $v \in \mathbb{R}^N$, by combining the above compression schemes with pruning rate



Fig. 2. Gradient aggregation.

 ρ and quantization levels *L*, the number of bits to communicate $(\mathcal{E}_{csr}(\boldsymbol{m}), \mathcal{E}_{H}(\tilde{\boldsymbol{v}}'), sgn(\hat{\boldsymbol{v}}), |\hat{\boldsymbol{v}}|_{min}, |\hat{\boldsymbol{v}}|_{max})$ is upper bounded by

$$C_{out}C_{in} + \lceil (1-\rho)C_{out}C_{in} \rceil K^2 (1 + \log_2 L) + 64.$$
 (3)

Specifically, we use a fixed L = 8 for convolution layer and L = 4 for fully connected layer during the implementation and the compression ratio is only determined by ρ . Naturally, according to Lemma 3, there is a near-linear relationship between ρ and the size of compressed gradient. We can directly acquire the gradient pruning rate ρ for a given compression ratio. Unlike previous gradient compression methods [5], [12] that only reduce the local gradient size with a predefined set of compression ratios, the proposed method can perform fined-grained gradient compression in large range sizes. Note that the computation cost of local gradient compression is negligible compared to that of local model training.

B. Server-side Element-wise aggregation

There exist I devices that collaboratively a shared model, and we utilize $\mathcal{I} = \{1, 2, \dots, I\}$ to denote the device set. After collecting the compressed gradients uploaded from different devices, the parameter server first decodes the compressed gradients and obtains $\{v_{q,i}, \forall i \in \mathcal{I}\}$. Then the parameter server is responsible to compute the global gradient \overline{v} .

Let $v_{q,i} = \{v_{q,i}^k\}$ $(k = 1, 2, \dots, N)$ represent the compressed gradient uploaded from device *i*, and $m_i = \{m_i^k\}$ be its corresponding mask. The aggregated gradient can be expressed in an element-wise manner by $\overline{v} = \{\overline{v}^k\}$ $(k = 1, 2, \dots, N)$. The *k*-th entry \overline{v}^k is calculated by

$$\overline{v}^{k} = \begin{cases} \frac{1}{\sum_{i} m_{i}^{k} D_{i}} \sum_{i} v_{q,i}^{k} m_{i}^{k} D_{i}, & \text{if } \sum_{i} m_{i}^{k} D_{i} > 0\\ 0, & \text{otherwise} \end{cases}, \quad (4)$$

where D_i is the number of training data of device *i*.

We take an example to compute \overline{v} in the multilayer perceptron case, as illustrated by Fig. 2. We consider a simple application scenario of FL consisting of two devices. Here, we paint all gradient values of \overline{v} with 2 colors. Based on Eqn. (4), the connections in blue are only updated by device 1, the connections in orange are only updated by device 2, and the connections in black are updated by both devices 1 and 2, etc. Note that the method is straightforward to be extended to the case of convolution layer.



Fig. 3. Test accuracy on proxy dataset with respect to $1/\alpha$

C. Compressed Gradient Information

Referring to the tradeoff between model accuracy and communication overhead in [13], we infer that a high compression ratio in gradient compression leads to the deterioration of global model accuracy. In the following, we study the quantitative relationship between the compression ratio and global model accuracy.

We refer to the parameter fitting method in the previous work [14] and conduct experiments of gradient compression to measure different global model accuracy with respect to different compression ratios α of the devices. To this end, a naive fitting approach is to directly acquire the corresponding global model accuracy by enumerating a set of compression ratios in FL. However, the cost incurred by multiple times of decentralized training may overtake the gain of the parameter fitting itself. Alternatively, we propose to explore the prior knowledge of parameter fitting on a proxy task with public dataset, and then transfer it to the target task with decentralized dataset. Note that the idea of proxy dataset is widely used in the study of neural architecture search [15]. The overall parameter fitting experiments are performed in an offline manner, and the prior knowledge of this one-time fitting can be transferred into many FL tasks.

We adopt the CINIC [16] as the proxy dataset for the parameter fitting experiment. The details of the hyperparameter settings are shown in Section IV. In the experiments, we apply the control variate method. Given a unified compression ratio of the device, we evaluate the global model accuracy as $F(\alpha)$ after gradient compression. According to the previous work of model compression [17], there is a logarithmic relationship between the inversion of compression ratio and the compressed model accuracy. We observe that this relationship is also achieved in FL with gradient compression. Hence, we are motivated to formulate the global model accuracy $F(\alpha)$ by

$$F(\alpha) = \kappa_1 \log_2(\kappa_2/\alpha - \kappa_3) + \kappa_4, \tag{5}$$

where parameters κ_1 , κ_2 , κ_3 and κ_4 are experimentally fitted to measure the training performance with the given unified compression ratio α . The experimental results are presented in Fig. 3. We obtain the constant parameters $\kappa_1 = 0.024$, $\kappa_2 =$ 19.221, $\kappa_3 = 2.561$, $\kappa_4 = 0.609$. With the decrease of α , more accurate gradient information is collected from the compressed local gradients, which is helpful to improve the global model accuracy. For example, when α is small enough (e.g., $\alpha \leq 25$), the compressed local gradient is able to reveal sufficient and accurate gradient information and at this time, the global model accuracy is almost identical to that of the conventional FL algorithm. Therefore, we consider that for a single device, a lower compression ratio α could also give rise to more accurate gradient information and vice versa.

III. PROBLEM FORMULATION AND SOLUTION

A. Learning Accuracy-Energy Efficiency Tradeoff Problem

To study FedGreen, we consider an application scenario of FL including a parameter server co-located with a base station and I devices. Let f_i and r_i denote the computing frequency (CPU cycles/second) and uplink data rate (bps) of device *i*, respectively. Given the model structure, the original weight and gradient size S and computing workloads per training sample W are easy to calculate. The number of local epochs n for model training could be set empirically. Similar to [18], we pay attention to the total latency of local model training and gradient uploading in each communication round of the global model training. Specifically, with the instruction of the parameter server, each device independently trains the published global model with the local dataset. The local training time is nWD_i/f_i . During the gradient uploading process, due to a compression ratio α_i , gradient uploading time of device i is consumed by $S/(\alpha_i r_i)$. For multiple access of the devices in the uplink data transmission, we consider device *i* communicates with the base station via the frequency domain multiple access technology. For device *i*, the uplink data rate r_i is calculated by

$$r_i = b_i \log_2(1 + \frac{p_i |h_i|^2}{N_0 b_i}),\tag{6}$$

where b_i and p_i indicate the available bandwidth and transmitter power of device *i* respectively, h_i represents channel gain between the device and base station, and N_0 indicates the noise power-spectral-density. In addition to the time consumption, the amount of energy consumed for local model training is $\varepsilon_i f_i^2 n D_i W_i$, where ε_i is an energy coefficient of the device. As introduced by [19], ε_i indicates the effective switched capacitance relying on the chip architecture.

In FL with gradient compression, a low compression ratio α_i reduces the data size of the local gradient and causes less communication overheads to device i in the uplink data transmission. But this leads to less accurate gradient information, the global model accuracy will be degraded to a degree. Besides, similar to the conventional FL algorithm, we consider the influence of the amount of local training data D_i when evaluating the contribution of the gradient information submitted by device i. We utilize D_i/\mathcal{D} as a weighting factor of device i, where $\mathcal{D} = \sum_i D_i$. Ultimately, we measure the overall contribution of all the compressed local gradients from the devices by

$$\mathcal{F}(\{\alpha_i\}_{1 \le i \le I}) = \frac{1}{\mathcal{D}} \sum_{1 \le i \le I} D_i F(\alpha_i), \tag{7}$$

where $F(\alpha_i)$ is computed by Eqn. (5) to roughly measure the training performance of device *i* after the device compresses its local gradient by ratio α_i . Until now, we introduce $\mathcal{F}(\{\alpha_i, \forall i\})$ as a new performance metric to evaluate the learning performance of FL with gradient compression.

Considering the learning performance of FL and total energy consumption of all the devices, there exists a tradeoff problem in FL with gradient compression. The goal function can be expressed by

$$\mathcal{G} = \mathcal{F}(\{\alpha_i\}_{1 \le i \le I}) - \varpi J \sum_{i=1}^{I} \left(\frac{p_i S}{\alpha_i r_i} + \varepsilon_i f_i^2 n D_i W\right), \quad (8)$$

where J is the predefined number of global iterations and ϖ is a presetting weighting factor. To achieve the goal, we jointly optimize the compression ratio α_i and computing frequency f_i of each device $i, 1 \leq i \leq I$. At the same time, there are essential constraints for the tradeoff problem. The compression ratio α_i is equal to or larger that 1 and f_i^{\max} is an upper limit of f_i . Moreover, a parameter server-defined latency constraint should be satisfied for each device. Here, T^{\max} is the training delay requirement of a single global iteration for each device. Finally, we summarize the whole problem with necessary constraints as follows.

$$\begin{array}{ll} (\text{P1}): & \max \ \mathcal{G} \\ \text{subject to:} & \alpha_i \ge 1, \forall i, \\ & 0 < f_i \le f_i^{\max}, \forall i, \\ & \frac{S}{\alpha_i r_i} + \frac{n D_i W}{f_i} \le T^{\max}, \forall i, \\ \text{variables}: & \alpha_i, f_i, \forall i \end{array}$$
(9)

For efficiency guarantee, the above optimization problem is solved by each device in a decentralized manner. Specifically, finding the optimal solution $\{f_i^*, \alpha_i^*\}$ for device *i* only requires its own hardware states and channel state information. Hence, each device can dynamically update its training strategy to cope with the time-varying environment during the training period. After solving the problem, each device utilizes a suitable computation frequency to perform the local training task and afterward compress the local gradient with a specific ratio. In this paper, we design FedGreen to reduce the total energy consumption of all the devices in the goal function and also consider a latency constraint for each device. Our scheme is beneficial to achieve FL with fine-grained gradient compression for green MEC.

B. Solution

To tackle the above optimization problem, we first pay attention to the bottom computing resource allocation problem. With the decisions of gradient compression $\{\alpha_i, \forall i\}$, the subproblem that only involves the decision variables $\{f_i, \forall i\}$ is formulated to minimize the total energy consumption cost of the devices, which is expressed as follows

$$\begin{array}{ll} (\text{P1-Bottom}): & \min \ \sum_{1 \leq i \leq I} \left(\frac{p_i S}{\alpha_i r_i} + \varepsilon_i f_i^2 n D_i W_i \right) \\ \text{subject to:} & 0 < f_i \leq f_i^{\max}, \forall i, \\ & \frac{S}{\alpha_i r_i} + \frac{n D_i W}{f} \leq T^{\max}, \forall i \\ \text{variable:} & f_i, \forall i \end{array}$$

$$\begin{array}{ll} (10) \\ \end{array}$$

Since the total energy consumption increases with the increase of f_i , we realize that if α_i is confirmed for device *i*, f_i is solved according to the time delay constraint and the upper limit. Hence, the solution of f_i is

$$f_i^* = \min(\frac{nD_iW}{T^{\max} - \frac{S}{\alpha_i r_i}}, f_i^{\max}).$$
(11)

Note that when f_i^{\max} is large enough, f_i is straightforwardly solved according to the equalized the time delay constraint. We utilize an intermediate variable $\beta_i \in [0, 1]$ to suppose that for device i,

$$\begin{cases} \frac{S}{\alpha_i r_i} = \beta_i T^{\max},\\ \frac{n D_i W}{f_i} = (1 - \beta_i) T^{\max}. \end{cases}$$
(12)

Considering that $f_i \leq f_i^{\max}$, a lower limit of β_i is required by

$$\beta_i^{\min} = 1 - \frac{nD_iW}{f_i^{\max}T^{\max}}.$$
(13)

We derive the partial derivatives of \mathcal{G} with respect to β_i ,

$$\begin{pmatrix}
\frac{\partial \mathcal{G}}{\partial \beta_i} = \frac{r_i T^{\max}}{S} \frac{D_i \kappa_1 \kappa_2}{\mathcal{D}(\kappa_2 \lambda_i - \kappa_3) \ln 2} \\
- \varpi J(p_i T^{\max} + 2\varepsilon_i \frac{n^3 D_i^3 W^3}{(T^{\max})^2 (1 - \beta)^3}), \\
\frac{\partial^2 \mathcal{G}}{\partial \beta_i^2} = -(\frac{r_i T^{\max}}{S})^2 \frac{D_i \kappa_1 \kappa_2^2}{\mathcal{D}(\kappa_2 \lambda_i - \kappa_3)^2 \ln 2} \\
- 6 \varpi J \frac{\varepsilon_i n^3 D_i^3 W^3}{(T^{\max})^2 (1 - \beta)^4} < 0,
\end{cases}$$
(14)

where $\lambda_i = 1/\alpha_i$. Clearly, \mathcal{G} is concave on β_i and we solve the optimal solution of β_i based on the first-order optimality condition $\partial \mathcal{G}/\partial \beta_i = 0$. But it is difficult to directly solve β_i . Alternatively, we apply the binary search method to seek an approximate solution of β_i within the range $[\beta_i^{\min}, 1]$. Finally, α_i and f_i are solved by substituting the approximate solution of β_i into Eqn. (12).

IV. PERFORMANCE EVALUATION

A. Experiment Setting

We consider the application of FL for image classification on the CIFAR-10 dataset, with 16 mobile devices. The hyperparameters for the FL algorithm are shown as follows: local epoch 1, batch size 64, learning rate 0.05, the number of global iterations 300, and decay rate per round 0.996 by default. For the IID data setting, we shuffle the training samples and uniformly dividing them to all the devices. For the non-IID setting, we consider heterogeneous partition with distribution of $\mathbf{p}_c \sim \text{Dir}_{c,i}(0.5)$ and allocate a proportion $\mathbf{p}_{c,i}$ of the training samples of class c to device i. We conduct the experiments on VGG-9 model [20]. The original gradient size and computing workloads per training sample are empirically measured as S = 111.7 Mb and W = 0.98 megacycles, respectively. The parameter settings for the hardware configuration and channel status of the devices are shown as follows: energy coefficient $\varepsilon_i \sim U[5 \times 10^{-27}, 1 \times 10^{-26}]$, computing frequency $f_i^{\text{max}} \sim U[1.5, 4]$ GHz, power-spectral-density $N_0 = -114$ dBm, available bandwidth $b_i \sim U[0.8, 5]$ MHz. We set the weighting factor $\varpi = 1 \times 10^{-4}$ and $T_{\text{max}} = 100$ seconds by default.

B. Performance Evaluation

Convergence of the binary search solution. We first show that the binary search method can converge finally and achieve the approximately optimal solution for problem P1. Fig. 4(a) shows the evolution of the compression strategies of three randomly selected devices and the convergence of goal function. In our scheme, a device with lower computation or communication capacity is suggested to adopt a large compression ratio in gradient compression, and vice versa.

Impact of local epoch n. A larger n encourages each device to perform more iterations of the local model training. But it may lead to the divergence of the local gradients. As shown in Fig. 4(b), we observe a degradation of the final test accuracy when n > 4, which matches with the existing study in [21]. Besides, with the increase of n, the energy consumption of local model training increases drastically. Hence, it is recommended to use a moderate local epoch (e.g., n < 5) to save energy and avoid the gradient divergence.

Impact of weighting factor ϖ . We discuss the impact of the weighting factor on the performance of our scheme. Fig. 4(c) shows that we conduct experiments under IID data setting with respect to ϖ . A large value of ϖ means that the parameter server would like to reduce the total energy consumption of the devices. The convergence accuracy of the FL algorithm could be a sacrifice at this time. Based on the experiments, it is empirically suggested to adjust the tradeoff parameter from 5×10^{-5} to 5×10^{-4} .

Comparison with baseline. Next, we compare our scheme FedGreen with the following baseline schemes.

- **Random**. Each device utilizes a random strategy of gradient compression and α_i is randomly selected from [50, 300]. Computing frequency f_i is calculated according to Eqn. (11).
- Uniform. All the devices utilize an identical ratio for gradient compression [4]. We calculate the average compression ratio α of FedGreen, and obtain {α_i = α, ∀i}.
- Selection. Motivated by [6], we exclude the top 25% of the devices with the largest energy consumption in the uniform policy.

The convergence curves of these schemes over the consumed energy consumption under the IID and non-IID setting are shown as Figs. 5(a) and 5(b), respectively. With the same energy consumption requirement, our scheme outperforms the existing schemes to improve the global model accuracy. Meanwhile, as the record of our experiments, FedGreen achieves the best final test accuracy in both IID and non-IID settings. In addition, we provide the experiment results of required energy consumption for achieving 80% test accuracy in Fig. 5(c). We realize that FedGreen is indeed superior to the above baseline schemes, which consumes the least energy for the convergence performance. Particularly, to achieve the same test accuracy of 80%, compared with the Selection scheme, FedGreen reduces 32% and 57% of the energy consumption under the IID and non-IID setting, respectively.



Fig. 4. The convergence of binary search solution and the impact of different parameters on the overall performance of FedGreen.



Fig. 5. Performance comparison with various baseline schemes on CIFAR-10 dataset.

V. CONCLUSION

We introduce fine-grained gradient compression for FL in MEC, and proposes FedGreen to dynamically adjust the compression ratios of the devices in an energy-efficient way. We present the basic operations to enable different devices to adopt different compression ratios on demand. Furthermore, we pay attention to the overall performance of FL with gradient compression and study a learning accuracy-energy efficiency tradeoff problem. Based on the applicable methods, we find the approximately optimal compression ratio and computing frequency for each device. Numerical experiments demonstrate that our scheme outperforms the baseline schemes in saving energy on the device side while guaranteeing the accuracy performance of FL in MEC.

ACKNOWLEDGMENT

Rong Yu is the corresponding author of this paper. The work is supported in part by National Natural Science Foundation of China (No. 61971148, No. 62001125), Guangxi Natural Science Foundation, China (No. 2018GXNSFDA281013), and Foundation for Science and Technology Project of Guilin City (No. 20190214-3). The work of M. Pan was supported in part by the U.S. National Science Foundation under grants CNS-1801925, CNS-2029569, and CNS 2107057.

REFERENCES

- B. McMahan, E. Moore, D. Ramage *et al.*, "Communication-efficient learning of deep networks from decentralized data," in *AISTATS*, 2017.
- [2] Y. Mao, C. You, J. Zhang *et al.*, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 2017.
- [3] S. Shi, K. Zhao, Q. Wang *et al.*, "A convergence analysis of distributed sgd with communication-efficient gradient sparsification." in *IJCAI*, 2019, pp. 3411–3417.

- [4] F. Sattler, S. Wiedemann *et al.*, "Robust and communication-efficient federated learning from non-i.i.d. data," *IEEE TNNLS*, pp. 1–14, 2019.
- [5] N. Shlezinger, M. Chen, Y. C. Eldar et al., "Uveqfed: Universal vector quantization for federated learning," *IEEE Trans. Signal Process.*, 2020.
- [6] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *ICC*, 2019, pp. 1–7.
- [7] W. Xia, T. Q. Quek, K. Guo *et al.*, "Multi-armed bandit based client scheduling for federated learning," *IEEE TWC*, pp. 1–1, 2020.
- [8] D. Ye, R. Yu, M. Pan, and Z. Han, "Federated learning in vehicular edge computing: A selective model aggregation approach," *IEEE Access*, vol. 8, pp. 23 920–23 935, 2020.
- [9] L. Li, D. Shi, R. Hou *et al.*, "To talk or to work: Flexible communication compression for energy efficient federated learning over heterogeneous mobile edge devices," in *INFOCOM*, 2021, pp. 1–10.
- [10] D. Shi, L. Li, R. Chen *et al.*, "Towards energy efficient federated learning over 5g+ mobile devices," *arXiv preprint arXiv:2101.04866*, 2021.
- [11] R. Yu and P. Li, "Toward resource-efficient federated learning in mobile edge computing," *IEEE Network*, vol. 35, no. 1, pp. 148–155, 2021.
- [12] D. Alistarh, D. Grubic *et al.*, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," in *NeurIPS*, 2017.
- [13] J. Konečný, H. B. McMahan, F. X. Yu *et al.*, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint* arXiv:1610.05492, 2016.
- [14] Y. Zhan, P. Li, Z. Qu et al., "A learning-based incentive mechanism for federated learning," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6360– 6368, 2020.
- [15] B. Zoph, V. Vasudevan *et al.*, "Learning transferable architectures for scalable image recognition," in *CVPR*, 2018, pp. 8697–8710.
- [16] L. N. Darlow et al., "Cinic-10 is not imagenet or cifar-10," arXiv preprint arXiv:1810.03505, 2018.
- [17] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," in *ICLR*, 2016.
- [18] Z. Yang, M. Chen, W. Saad *et al.*, "Energy efficient federated learning over wireless communication networks," *IEEE TWC*, vol. 20, no. 3, pp. 1935–1949, 2021.
- [19] X. Huang, P. Li, R. Yu et al., "Fedparking: A federated learning based parking space estimation with parked vehicle assisted edge computing," *IEEE Transactions on Vehicular Technology*, 2021.
- [20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.
- [21] H. Wang, M. Yurochkin, Y. Sun et al., "Federated learning with matched averaging," in ICLR, 2020.