IoT Device Friendly and Communication-Efficient Federated Learning via Joint Model Pruning and Quantization

Pavana Prakash, *Graduate Student Member, IEEE*, Jiahao Ding, *Graduate Student Member, IEEE*, Rui Chen, *Graduate Student Member, IEEE*, Xiaoqi Qin, *Member, IEEE*, Minglei Shu, *Member, IEEE*, Qimei Cui, *Senior Member, IEEE*, Yuanxiong Guo, *Senior Member, IEEE*, and Miao Pan, *Senior Member, IEEE*

***Abstract*—Federated learning (FL) through its novel applications and services has enhanced its presence as a promising tool in the Internet of Things (IoT) domain. Specifically, in a multiaccess edge computing setup with a host of IoT devices, FL is most suitable since it leverages distributed client data to train high-performance deep learning (DL) models while keeping the data private. However, the underlying deep neural networks (DNNs) are huge, preventing its direct deployment onto resource-constrained computing and memory-limited IoT devices. Besides, frequent exchange of model updates between the central server and clients in FL could result in a communication bottleneck. To address these challenges, in this article, we introduce GWEP, a model compression-based FL method. It utilizes joint quantization and model pruning to reap the benefits of DNNs while meeting the capabilities of resource-constrained devices. Consequently, by reducing the computational, memory, and network footprint of FL, the low-end IoT devices may be able to participate in the FL process. In addition, we provide theoretical guarantees of FL convergence. Through empirical evaluations, we demonstrate that our approach significantly outperforms the baseline algorithms by being up to 10.23 times faster with 11 times lesser communication rounds, while achieving high-model compression, energy efficiency, and learning performance.**

***Index Terms*—Federated learning (FL), gradient compression, Internet of Things (IoT) devices, model pruning, quantization.**

Manuscript received 28 August 2021; revised 1 December 2021; accepted 7 January 2022. Date of publication 25 January 2022; date of current version 25 July 2022. The work of Pavana Prakash, Jiahao Ding, Rui Chen, and Miao Pan was supported in part by the U.S. National Science Foundation under Grant CNS-2029569 and Grant CNS-2107057. The work of Qimei Cui was supported by the National Youth Top-Notch Talent Support Program. The work of Yuanxiong Guo was supported in part by the U.S. National Science Foundation under Grant CNS-2029685 and Grant CNS-2106761. *(Corresponding author: Minglei Shu.)*

Pavana Prakash, Jiahao Ding, Rui Chen, and Miao Pan are with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77004 USA (e-mail: pprakash3@uh.edu; jding7@uh.edu; rchen19@uh.edu; mpan2@uh.edu).

Xiaoqi Qin and Qimei Cui are with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: xiaoqiqin@bupt.edu.cn; cuiqimei@bupt.edu.cn).

Minglei Shu is with the Shandong Artificial Intelligence Institute, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250353, China (e-mail: smlsmile1624@163.com).

Yuanxiong Guo is with the Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX 78249 USA (e-mail: guoyuanxiong@gmail.com).

Digital Object Identifier 10.1109/JIOT.2022.3145865

I. Introduction

THE UBIQUITOUS trend of connecting physical objects to the Internet has enabled Internet of Things (IoT) to reshape the technological landscape. The assistance of deep learning (DL) to gain deep insights onto the data has further fuelled innovative IoT applications and devices, such as wearables, smart home devices, etc. [1]. However, these compute-intensive applications seek low latency and high bandwidth and also generate a remarkable volume of data on the devices. This creates additional challenges, such as data management, storage, security, etc. Hence, the need for enhancing the speed and quality of data processing calls for a paradigm shift from the typical central cloud processing to edge computing. Under the architecture of multiaccess edge computing (MEC), the core of computation and services are performed at locations close to where the data are generated. This enables in realizing applications that demand very high bandwidth and low latency. MEC together with the leap of DL and the wealth of data at end devices have opened up countless possibilities for meaningful applications. However, utilizing centralized machine-learning (ML) algorithms in this setting is inefficient since uploading and storing private bulk data creates a large communication and storage requirement. Moreover, adversarial access to data could pose privacy implications [2], [3]. Hence, to address these issues, decentralized methods such as federated learning (FL) is popularly opted.

FL is an emerging distributed ML framework where end devices collaboratively train a global model with locally available data, under the coordination of a central server in a decentralized manner [4]. With joint training offered by FL without the need to share their private data, coupled with the edge provided by MEC, FL in MEC is deemed to be a propitious solution to enable on-device training at data sources. This conflux has indeed paved the way for many widespread applications ranging from wearable smart healthcare devices [5], smart vehicles [6] to the classic Google GBoard [7].

While the underlying overparametrized deep neural networks (DNNs) have shown impressive results on ML tasks, they consume considerable energy, memory bandwidth, storage, and computational resources. This is due to the fact that a huge DNN often contains a large number of parameters and the

size of these models have grown bigger and deeper over time. Apparently, as opposed to intensive computation task processing by cloud computing, MEC is a resource-constrained computing and delay-sensitive environment. Further, while large operating devices such as central servers have higher processing capabilities without restriction on power, IoT devices are memory-constricted and battery-powered [8]. Thus, being both computational and memory intensive, the direct deployment of DNNs on IoT devices is confined, leading to on-device usage as a challenge.

Additionally, the incurring communication cost is a principal constraint in edge computing-assisted FL. Since a large number of users transmit their local updates of model parameters and stochastic gradients to the parameter server, and the server broadcasts the model updates, FL involves a high communication bandwidth requirement. The problem is worsened when working with high-capacity models with a huge number of parameters. Therefore, the need for an efficient FL process has stemmed from the desire for reduced computation, memory accesses, storage, as well as communication requirements, while maintaining the performance at par.

Notably, model compression is a growing area of research that reduces the size of DNNs without compromising on the model quality and accuracy of performance. In addition, it reduces the number of memory accesses that generally consume considerable time and energy. This results in increased memory bandwidth for fetching compressed model parameters [9], thus improving the inference time. For example, quantization represents the model parameters and gradients with smaller bit widths such that the intermediate data are stored in lower precision numerical formats such as int8. This reduces the computational demand of the complex neural networks, and the frequency of memory accesses. Moreover, since it replaces the floating-point operations, fetching fixed-point representations with shorter bit width for weights and gradients requires lesser memory bandwidth. By running calculations on smaller bits and sparser vectors, the inference time is reduced. Besides, pruning to a smaller compression ratio significantly reduces the memory requirements and model size, which is highly correlated to the inference time [10]. Within the realm of model compression and communication-efficient FL, a number of notable works have proposed various schemes. Most communication-efficient methods such as [11] relieve the communication bottleneck by targeting client-to-server communication but, fail to consider the computational capabilities of the devices, limiting their usage on IoT devices. On the contrary, pioneer works on model compression such as [12] have aimed to compress the model making it suitable for resource-constrained devices specifically. However, most of these works aim at single machine setup and do not consider a distributed setting restricting its wide applicability.

Moreover in FL, while more focus is typically on relieving the communication bottleneck, the latest surge of research has paved the way for the rapidly expanding 5G and the upcoming 6G networks, which mitigate communication burdens [8]. This makes computations nearly comparable with communications. For instance, a single step of local computation on the ResNet50 model over GPU consumes few hundreds of milliseconds [13], which nearly corresponds to the time taken to transmit over a wireless connection with a transmission rate of 1 Gb/s. Therefore, it is also essential to alleviate the computation burdens while making FL methods more communication efficient.

Hence, in this article, we jointly address the challenges of implementing DNNs over resource constricted devices and propose an efficient FL scheme named GWEP, by applying double quantization jointly with model pruning. Model pruning is a popular model compression method that removes a significant portion of the network weights, while nearly preserving the test accuracy as the original dense models. It not only reduces the model size but also reduces the inference time and cost while raising the power efficiency. Correspondingly, quantization is another compression scheme that reduces the number of bits representing each connection. Hence, lower bit mathematical operations of the quantized parameters yield large computational gains and require less memory accesses thereby reducing memory bandwidth and latency. Ultimately, GWEP consolidates the benefits of reducing storage, communication, and computation requirements along with accelerated training, into a single framework.

With the above motivation, our salient contributions can be summarized as follows.

1) We propose a novel IoT device friendly, computation, storage, and communication-efficient mechanism, called GWEP. It comprises weight and gradient quantizations in conjunction with model pruning and error feedback. By reducing model redundancy and computation complexity, it enables originally large-sized DNNs to be effectively deployed on resource-constrained IoT devices.
2) We integrate our algorithm with the FL training procedure to enhance the communication efficiency and accelerate the training process. In addition, it enhances the learning performance in the delay-sensitive MEC environment relieving the communication bottleneck issue.
3) We establish a theoretical guarantee of convergence of the proposed algorithm using a distributed adaptive stochastic gradient method with adaptive learning rate in the nonconvex stochastic settings.
4) We illustrate the effectiveness of our proposed algorithm through performance evaluation of applying over real-world data sets to train large-sized networks for image classification tasks. Our proposition reduces the model size significantly and improves the training time and communication cost while achieving high test accuracy.

The remainder of this article is organized as follows. Some preliminaries are explained in Section II. Sections III and IV describe the system model and our proposed method in detail, respectively. Section V elucidates the main results with convergence analysis. We perform simulations and provide performance evaluation with results in Section VI. Section VII surveys existing literature related to our work, while Section VIII concludes this article.

TABLE I
LIST OF IMPORTANT NOTATIONS AND DEFINITIONS

Notation	Definition
f	Non-convex loss function
f^*	Optimal solution
m_t	First moment estimate of the gradient
v_t	Second moment estimate of the gradient
$\hat{w}_t$	Quantized model weights
$\tilde{w}_t$	Pruned model weights
g_t	Gradient estimation
$\hat{g}_t$	Quantized gradients
$Q_g(.)$	Gradient quantization mapping
$Q_w(.)$	Weight quantization mapping
τ_g	Gradient quantized level
τ_w	Weight quantized level
τ_p	Pruning level
α_t	Base learning rate
β_t	Momentum parameter
θ_t	Exponential moving average parameter
ϵ	Adaptive learning rate hyperparameter
Δ_t^n	Noisy gradient estimation
Δ_t	Noisy gradient estimation aggregate
$\odot$	entry-wise (Hadamard) product
$\|.\|$	ℓ_2-norm of a vector
$\|.\|_\infty$	maximum norm of a vector

Important notations and definitions used in this article are listed as follows in Table I.

II. PRELIMINARIES

A. Federated Learning

In FL settings, the participating clients (IoT devices) train a shared global model in a decentralized manner. In a typical FL setup, the server sends a global model to all connected clients and each client calculates local model updates based on its local training data. Only these updates are transmitted to the server to aggregate and obtain an updated global model, preserving the users' data privacy. This process continues until a global convergence of the model is attained.

B. Pruning

Weight pruning is one of the most promising model compression methods that remove the less contributing weights and connections, to produce a compressed version of a model that is suitable for on-device deployment. Besides, it nearly preserves the test accuracy attained by denser models. Among other pruning techniques such as pruning before or after training, dynamic or incremental pruning is a newer pruning technique where the network is pruned incrementally during training. The pruning mask is updated based on the network state at every few iterations, allowing to explore different sparsity masks. This saves the effort of determining an optimal mask in prior and keeping it fixed unlike pruning before the training method. Further, in saliency-based weight pruning, the system repeats to learn which weights are important and removes the least important weights. These result in a monotonic increase in sparsity, eliminating the need for retraining, which is mandated in methods of pruning after training. Hence, we consider this approach in our algorithm to perform model pruning.

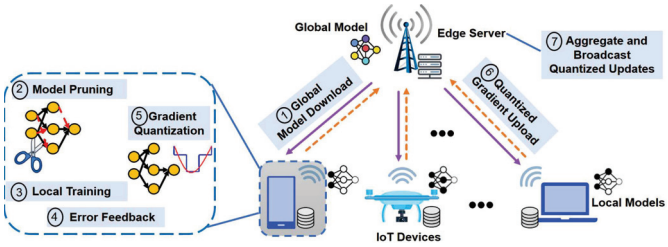

Fig. 1. Overview of the proposed communication-efficient FL algorithm.

C. Quantization

Generally, the quantization process reduces the number of bits required to represent a number. This is especially vital in DL since running a DNN for a task such as an image classification, results in millions of multiplication and addition operations in the hardware. Hence, performing mathematical operations and intermediate calculations on the quantized parameters owning lower bits, result in large computational gains. Besides, quantized data requires lesser memory access thereby reducing memory bandwidth.

III. SYSTEM MODEL

We consider a user equipment (UE) level-edge level of an MEC-assisted FL system, consisting of one edge (parameter) server and a set $\mathcal{N} = \{1, 2, \ldots, N\}$ of N IoT edge devices (clients or UEs). An overview of the system model using the proposed method is depicted in Fig. 1. The output of the system for an ML problem is characterized by model parameter w that captures the learning model with the loss function f. Our goal is to minimize the system-level global loss of the learning model. This nonconvex loss function is considered as the stochastic learning problem

$$\min_w f(w) \coloneqq \min_w \frac{1}{N} \sum_{n=1}^{N} f_n(w) \tag{1}$$

where $f_n(w)$ is the local objective function of each client n. It can be defined as the expected loss of its local sample distribution as

$$f_n(w) \coloneqq \mathbb{E}_{\xi \sim P}\left[\tilde{f}(w, \xi)\right] \tag{2}$$

where ξ is a random variable with unknown probability distribution P. But, without access to the underlying distribution P of the random variable ξ, it is improbable to access the exact gradient $\nabla f(w)$. Rather, we use a stochastic estimate of $\nabla f(w)$ as $g_t \coloneqq \nabla f(w_t, \xi_t)$ with w_t from the given sampled sequence $\{\xi_t\}$. We assume a synchronous model for both communication and computation.

To achieve model compression, we use the deterministic weight quantization method since it achieves better performance for quantizing weights [14]. We consider the popular deterministic ternary compression method used in works such as [15], which converts the full precision of 32 floating-point bits to b-bit integer. Accordingly, we describe the weight quantizer $Q_w(.)$ as follows.

Example 1: For a weight vector w, we can define a general quantization mapping $Q_w(.) : \mathbb{R}^n \to \mathbb{R}^n$ as

$$Q_w(w_i) = \begin{cases} +a, & \text{if } w_i > a/2 \\ 0, & \text{if } |w_i| \leq a/2 \\ -a, & \text{if } w_i < -a/2 \end{cases} \tag{3}$$

in quantization set $\{-1, -(k-1/k), \ldots, -(1/k), 0, (1/k), \ldots, (k-1/k), 1\}$. Here, $a = \|w\|_\infty$ denotes the range of the original vector and $k = 2^{b-1} - 1$.

While deterministic weight quantization achieves better performance than stochastic weight quantization, deterministic gradient quantization makes the quantized gradient biased, and the analysis more complex. Hence, we consider a more general stochastic linear quantization of gradients such as the method proposed in [16] as follows.

Example 2: The stochastic quantization mapping $Q_g(\cdot) : \mathbb{R}^d \to \mathbb{R}^d$ with quantization level p, probability $\mathbf{p}$, $a = \|g\|$, and integer m such that $|g_i|/a \in [m/p, (m+1)/p]$, can be defined as

$$Q_g(g_i) = a \cdot \text{sgn}(g_i) \cdot \begin{cases} m/p, & \text{w.p. } 1 - \mathbf{p}\left(\frac{|g_i|}{a}, p\right) \\ (m+1)/p, & \text{otherwise.} \end{cases} \tag{4}$$

The weight and gradient quantization functions $Q_w(.)$ and $Q_g(.)$ defined based on (3) and (4), respectively, yield the corresponding quantized weights $\hat{w}$ and gradients $\hat{g}$. Here, $Q_g(g_t)$ is an unbiased estimator of g_t. Additionally, we adopt saliency-based weight pruning as the next model compression technique. Motivated by [17], we apply mask $\mathbf{m} \in \{0,1\}^d$ to weight vector $w \in \mathbb{R}^d$ in d-dimension, to obtain a sparse model $\dot{w} \coloneqq \mathbf{m} \odot w$.

IV. Efficient Federated Learning With GWEP

In this section, we present our algorithm called GWEP, which comprises weight and gradient quantizations, model pruning, and error feedback in concert. This sequence of model compression is introduced in the novel setting of FL. By integrating such a sequence with the FL training procedure, we can design a resource, computation, and communication-efficient distributed training algorithm.

More specifically, the proposed scheme consists of three primary components. First, we perform downlink compression by quantizing the weights of the global model update. As training large-sized models generate global model updates of the size of hundreds of megabytes, it may be undesirable for small-sized devices to download them. Besides, although downlink bandwidth used by the server to broadcast the updated global model is much larger than the uplink bandwidth used by the IoT devices to transmit their local updates, for a substantial number of participants, the difference between the upload and download speeds may not be significant enough to ignore the impact of the downlink as shown in [18]. Second, to lessen the burden of computation as well as to reduce the latency at inference, the received model update is pruned to a desired sparsity. Third, we relieve the communication bottleneck on the uplink by quantizing each of the gradient updates sent at every global communicating round from all the connected clients to the server. As a result, we accomplish to address the strong requirements of computation and storage, and the communication bottleneck issues simultaneously. In addition, we incorporate an adaptive learning rate to further accelerate the training process and error feedback to alleviate any introduced compression errors.

Algorithm 1 FL With Quantization and Pruning (GWEP)

Parameters: Weight and gradient quantization functions $Q_w(.)$, $Q_g(.)$. Choose $\{\alpha_t\}$, $\{\beta_t\}$, $\{\theta_t\}$, target sparsity ratio S_f. Initial values: m_0^n, v_0^n, $e_1^n = 0$.
Server executes:
Initialize w_0 and send to clients
for each global iteration $t = 0, 1, \cdots, T$ **do**
  for each client $n \in \mathcal{N}$ in parallel **do**
    $\hat{g}_t^n \leftarrow$ **clientUpdate**$(\hat{w}_t, t)$
  end for
  Aggregate updates $\hat{g}_t = \frac{1}{N}\sum_{n=1}^{N} \hat{g}_t^n$
  Global model update $w_{t+1} = w_t - \hat{g}_t$
  Server sends quantized weights $\hat{w}_{t+1} = Q_w(w_{t+1})$
end for

On each client, **clientUpdate**$(\hat{w}_t, t)$:
Pruning process:
if f$|t$ **then**
  Compute current sparsity ratio S_t using (5)
  if $S_t < S_f$ **then**
    Compute mask $\mathbf{m} \leftarrow m_t(\hat{w}_t)$
    Apply mask to obtain pruned weights $\dot{w}_t \leftarrow \mathbf{m} \odot \hat{w}_t$
  else
    No pruning $\dot{w}_t \leftarrow \hat{w}_t$
  end if
end if
Local training process:
Stochastic gradient for mini-batch ρ at set local iterations, $g_t^n = \nabla f(\dot{w}_t, \rho)$
$m_t^n = \beta_t m_{t-1}^n + (1-\beta_t) g_t^n$
$v_t^n = \theta_t v_{t-1}^n + (1-\theta_t)[g_t^n]^2$
$\Delta_t^n = \alpha_t \frac{m_t^n}{\sqrt{v_t^n + \epsilon}}$
Gradient quantization and Error feedback:
Send to server $\rightarrow$ $\hat{g}_t^n = Q_g(\Delta_t^n + e_t^n)$
Error accumulation
$e_{t+1}^n = \Delta_t^n + e_t^n - \hat{g}_t^n$

As shown in Algorithm 1, weight quantization is executed at the server, while the remaining steps are performed on each client. Particularly, model pruning is performed at clients instead of the server owing to two reasons. First, with the knowledge of the private training data residing on the users loaded into the model, the model accuracy is much better compared to pruning without data at the server [19]. Second, since FL at the edge is run on different types of devices, this gives a flexibility of choosing different pruning levels and strategies at different devices as per their requirements. We intend to include different pruning levels in the theoretical analysis as part of our future work. The comprehensive procedure of the

algorithm consists of four stages and we now give detailed descriptions of each stage.

Weight Quantization: The initial global model w_0 is initialized at the edge server. Before broadcasting the global model to all the clients, the weights are quantized based on (3) to obtain $\hat{w}_t$, following steps 4–6 in Algorithm 1.

Model Pruning: The quantized weight update broadcasted by the server is pruned at each client to obtain pruned weight, $\grave{w}_t$. Note that we employ a dynamic scheme of pruning during training so that the mask is adapted at every few iterations, based on the weights and stochastic gradient changes during training. The binary mask is computed and applied to the weights during the forward execution. Such weights do not get updated in the backpropagation step and the backpropagated gradients flow through these masks. We use magnitude-based unstructured weight pruning where instead of pruning based on a fixed threshold such as [12], we perform automated gradual pruning by increasing the sparsity incrementally as in [9]. The weight mask $\mathbf{m}$ computation directly depends on the sparsity condition. For each pruning frequency, once the current sparsity of the network weights is computed, the mask computation is triggered if the target sparsity ratio is not met. Hence, the mask is updated until the desired target sparsity S_f is reached. The intuition is to rapidly prune the network in the initial stage where the redundancy is higher and then to gradually reduce when there are fewer weights spared. With an initial sparsity value S_i starting at t_0 training step over a span of T global iterations (communication rounds), the sparsity at iteration t inline with the pruning frequency $\mathbf{f}$ can be evaluated as

$$S_t = S_f + (S_i - S_f)\left(1 - \frac{t - t_0}{T\mathbf{f}}\right)^3. \tag{5}$$

In effect, weights with smaller magnitude are pruned to achieve the preset target network sparsity as summarized in steps 12–16 of the algorithm.

Gradient Quantization: We further quantize the gradients evaluated over the pruned model to help reduce the uplink bandwidth requirements from the clients to the server. We replace the full precision of 32-bit floating point with a lower bit width INT8 format. The stochastic gradient g_t is computed over the pruned weight $\grave{w}_t$ which was quantized previously in step 4. Given its robustness to hyperparameter settings and fast convergence, we use the adaptive moment estimation algorithm, Adam [20]. We apply this gradient-based optimization of the stochastic objective function albeit on the quantized and pruned weights in this article. The computation then follows the iterates of this optimizer for the set number of local iterations over mini-batch ρ. We adopt an adaptive learning rate over the quantized and pruned weights, that accelerates training and avoids the effort of choosing a fixed learning rate and parameter tuning. The gradient quantization $Q_g(.)$ is then applied to the obtained gradient along with error feedback e_t, to arrive at the quantized gradient $\hat{g}_t^n$. This is transmitted to the server instead of the unquantized original gradient to relieve the uplink communication bottleneck. This local training and gradient quantization are summarized in steps 18–22.

The compressive sequence of our method introduces two main challenges. First, while adequate model compression along with good learning performance is achieved, the resulting values, however, include errors introduced by the compression. This leads to lower generalization and hence convergence of the system is affected. Second, a challenge persists in choosing the compression levels of each of the methods in GWEP, since there exists a tradeoff between compression and learning performance. In order to overcome this, we theoretically analyze the impact of pruning and quantization levels on convergence in detail in Section V. Further, to reduce the impact of compression errors, we incorporate error compensation into the FL procedure as illustrated below.

Error Feedback: We utilize an error compensation term e_t, to alleviate the errors introduced by the compression. As shown in [21], adding error feedback mitigates the error caused by compression and can help in recovering the actual performance accuracy. The main idea is to accumulate the compressed error and add it to the next compression step as shown in steps 22 and 23 of the algorithm. This leads to accelerating the global convergence.

Under synchronous mode, the above steps are performed at each client in parallel and sent to the server. The server gathers the gradient updates and aggregates to compute $\hat{g}_t$ as shown in step 8 of the algorithm. The updated global model is computed based on $\hat{g}_t$ (step 9) and the weights are then quantized before sending it back to the clients. This process continues until a global convergence of the model is attained.

V. Convergence Analysis

In this section, we first derive the upper estimate of the gradient in (1) of the quantized and pruned model characterized by the adaptive learning rate. Then, we derive the convergence rate of our algorithm for nonconvex problems in FL settings. We state the necessary standard assumptions and definitions required in the context of analyzing stochastic algorithms as follows.

Assumption 1: The assumptions about the objective function f can be summarized as follows.

1) The gradient ∇f is L-Lipschitz continuous such that, $\|\nabla f(u) - f(v)\| \leq L\|u - v\|\ \forall u, v \in \mathbb{R}^d$, for a constant $L > 0$,
2) The minimum value is lower bounded, i.e., $f^* = \min_{w \in \mathbb{R}^d} f(w) > -\infty$,
3) The gradient estimation g_t is upper bounded and unbiased, i.e., $\|g_t\| \leq G$ and $\mathbb{E}[g_t] = \nabla f(w_t)$.

In order to establish convergence in the case of biased compression operators, existing methods commonly use the contraction property and then rely on the guarantees stated in [22]. Likewise, we state the following assumptions with regard to our compressors.

Assumption 2: The quantization functions $Q(.)$ satisfy the τ-contraction property. Accordingly, the weight quantization involving a constant $0 < \tau_w \leq 1$ satisfies the following:

$$\|w - Q_w(w)\| \leq (1 - \tau_w)\|w\|. \tag{6}$$

Similarly, for a sequence of gradients, there exists a constant $0 < \tau_g \leq 1$ such that the following inequality holds:

$$\left\| g - Q_g(g) \right\| \leq (1 - \tau_g)\|g\|. \tag{7}$$

Besides, the pruning level τ_p can be assessed to be a parameter that determines the quality of pruning such that $\tau_p \in [0, 1]$ [17]. Hence, it follows the following assumptions.

Assumption 3: For a given pruning level τ_p, the inequality $\|w - \hat{w}\| \leq \sqrt{\tau_p}\|w\|$ is satisfied.

In order to establish the convergence rate, we make the following assumptions about the optimizer parameters, namely, base learning rate α_t, momentum parameter β_t, and exponential moving average parameter θ_t. This is based on the hyperparameter values used in the sufficient condition in [23].

Assumption 4: Let $\alpha_t = (\alpha/\sqrt{t})$, $\beta_t \in [0, \beta]$, where $\beta \in [0, 1]$ and $\theta_t = 1 - (\theta/t)$. Let constant $\mathcal{A} = \prod_{i=1}^{P}([\theta_i]/[\theta'])$, where θ' is a positive constant satisfying $\beta^2 < \theta' < 1$ and $P = \max\{i|\theta_i < \theta'\}$.

To illustrate the convergence analysis of Algorithm 1, we state theorem, useful lemmas, and consider the following relations. Let $\mathbb{E}_t[.]$ be the conditional expectation with respect to w_t conditioned on $\{w_{t-1}, w_{t-2}, \ldots, w_1\}$. We denote the step size relations as $\hat{\eta}_t^n = (\alpha_t/[\sqrt{\hat{v}_t^n + \epsilon}])$, $\hat{v}_t^n = \theta_t \hat{v}_{t-1}^n + (1 - \theta_t)[\sigma_t^{n^2}]$, and $\sigma_t^{n^2} = \mathbb{E}[g_t^{n^2}]$. We further denote $\Delta_t^n = \alpha_t([m_t^n]/[\sqrt{v_t^n + \epsilon}])$ on each client and denote $\Delta_t = (1/N)\sum_{n=1}^{N} \Delta_t^n$ as the aggregate of Δ_t^n. The upper bound of this aggregated gradient and the estimation of the noisy gradient incorporating the error feedback can be computed using the following two lemmas.

Lemma 1 (Upper Estimate of Δ_t [24]): Using the definition of m_t satisfying $\|m_t\|^2 \leq G^2$ and from the definition of Δ_t, its upper estimation can be derived as follows: $\sum_{t=1}^{T} \|\Delta_t\|^2 \leq (G^2/\epsilon)\sum_{t=1}^{T}(\alpha^2/t)$.

Lemma 2 (Noisy Gradient Estimation [24]): With the error feedback e_t^n as defined in Algorithm 1 and Δ_t^n, it holds that

$$\mathbb{E}\left[\frac{1}{N^2}\sum_{t=1}^{T}\sum_{n=1}^{N}\left\|e_t^n\right\|\left\|\Delta_t^n\right\|\right] \leq \frac{1 - \tau_g}{N\tau_g}\sum_{n=1}^{N}\mathbb{E}\left[\sum_{t=1}^{T}\left\|\Delta_t^n\right\|^2\right].$$

In our work, we discuss the convergence rate of our algorithm in terms of the average of the ℓ_2 norm of the gradient, computed over the quantized and pruned weights. In order to attain its value, we first compute the upper estimate of $\nabla f(\hat{w}_t)$ through the following lemma which yields a value based on the weighted norm of the step size.

Lemma 3 (Estimate on Quantized and Pruned Weights): For a model $\hat{w}_t$ that is weight quantized and pruned, we have the following estimate:

$$\mathbb{E}\left[\left\|\nabla f\left(\hat{w}_t\right)\right\|^2\right] \leq \frac{\sqrt{G^2 + \epsilon d}}{\alpha\sqrt{T}N}\sum_{n=1}^{N}\mathbb{E}\left[\sum_{t=1}^{T}\left\|\nabla f\left(\hat{w}_t\right)\right\|^2_{\hat{\eta}_t^n}\right].$$

Proof: The proof is provided in Appendix A. ∎

Utilizing the Lipschitz continuity of the gradient, we arrive at the relation with the pruning and quantization levels by introducing the following lemma.

Lemma 4 (Upper Estimate of M_t): We obtain the upper estimate as follows for $M_t = \mathbb{E}[\langle \nabla f(\hat{w}_t), \Delta_t\rangle + L\|\Delta_t\|^2]$:

$$\sum_{t=1}^{T} M_t \leq \frac{1}{\sqrt{\mathcal{A}}(1 - \sqrt{\gamma})}\left[\left(\frac{L(2 - \tau_g)G^2\alpha_t^2}{\epsilon\tau_g} + \mathcal{B}\theta\right)\sum_{t=1}^{T}\frac{1}{t} + \frac{4(1 - \tau_w)LDG\alpha_t\sqrt{T}}{\sqrt{\epsilon}}\left(1 + \sqrt{\tau_p}\right)\right] - \frac{(1 - \beta)}{2N}\sum_{n=1}^{N}\mathbb{E}\left[\sum_{t=1}^{T}\left\|\nabla f\left(\hat{w}_t\right)\right\|^2_{\hat{\eta}_t^n}\right]. \tag{8}$$

Proof: Refer to the proof provided in Appendix B. ∎

Theorem 1: Considering the sequence of iterative points $\hat{w}_t$ generated by Algorithm 1, with Assumptions 1–4 satisfied and the iterates $\|\hat{w}_t\| \leq D$ upper bounded, the convergence result of Algorithm 1 holds as

$$\mathbb{E}\left[\left\|\nabla f\left(\hat{w}_t\right)\right\|^2\right] \leq \frac{\mathcal{C} + \mathcal{D}\sum_{t=1}^{T}\frac{1}{t}}{\sqrt{T}} + \mathcal{E}$$

where $\mathcal{B}$ is from Lemma 4 as (16), as shown at the bottom of the p. 11

$$\mathcal{C} = \frac{2\sqrt{G^2 + \epsilon d}}{(1 - \beta)\alpha}\left(f(w_1) - f^*\right)$$

$$\mathcal{D} = \frac{2\sqrt{G^2 + \epsilon d}}{(1 - \beta)\alpha\sqrt{\mathcal{A}}(1 - \sqrt{\gamma})}\left(\frac{L(2 - \tau_g)G^2\alpha^2}{\epsilon\tau_g} + \mathcal{B}\theta\right)$$

and

$$\mathcal{E} = \frac{8\sqrt{G^2 + \epsilon d}(1 - \tau_w)\mathrm{LDG}}{\sqrt{\mathcal{A}}(1 - \sqrt{\gamma})\sqrt{\epsilon}(1 - \beta)}\left(1 + \sqrt{\tau_p}\right).$$

Proof: Refer to the proof in Appendix C. ∎

Remark 1: Theorem 1 demonstrates that the proposed algorithm converges to the neighborhood of stationary point of Problem (3) up to a constant related to the weight and gradient quantization levels (τ_w and τ_g) and is affected by the quality of pruning (τ_p). Moreover, the limit point of the iterate is affected only by the pruning and weight quantization levels. Lower sparsity implies less pruning with smaller τ_p and higher bit quantization implies less quantization leading to a faster convergence. Contraction factors are a function of T.

Extension to Other Compression Methods: Along with pruning, while we used quantization in our theoretical analysis, GWEP can be generalized to other compression schemes as long as they satisfy the properties stated in Assumption 2. For example, Q_g in Algorithm 1 can be replaced by other compressors such as top-k sparsification or biased quantization methods. Theoretically, using the contraction property for the gradient quantization helps us to establish convergence guarantees. Error feedback not only alleviates the error caused by compression but also aids in recovering the actual performance accuracy. Therefore, by incorporating it, we can generalize our scheme to biased compressors as well, where the value of τ can be set to 1 in (7). Furthermore, through it, the unbiased compressors can achieve better convergence rates, and biased compressors can benefit from rates close to its uncompressed counterparts as shown in [21].

VI. Performance Evaluation

In this section, we evaluate the performance of the proposed communication and delay-efficient FL via simulations.

A. Settings and Data

To evaluate the performance of our proposed scheme, we apply the proposed algorithm to train large-sized DNNs. We perform image classification tasks on large-sized popular DNNs: 1) ResNet20 [25] which is a 20 layers deep convolutional neural network and 2) LeNet5 [26] which is a convolutional neural network comprising seven layers with a composition of three convolutional layers, two subsampling layers, and two fully connected layers. Classification is performed using the following image data sets on ResNet20 and LeNet5, respectively. 1) *CIFAR-10:* containing labeled samples of color images in ten classes and 6000 images per class divided into 50 000 training and 10 000 test images and 2) *MNIST:* a database of grayscale images of handwritten digits with ten classes of 60 000 training and 10 000 test examples. We choose three baseline algorithms as benchmarks that are most related to our work: 1) FedAvg [4] is the vanilla FL algorithm with the dense model; 2) QAdam [24] employs a double quantization of weights and gradients on a single machine and distributed settings; and 3) DPF [17] adopts magnitude-based pruning with error feedback. Further, we jointly vary the sparsity ratio and quantization levels and use them to compare the performance of GWEP. These are indicated as "GWEP (quantization bits and sparsity ratio)" in Fig. 3(a)–(d).

For all evaluations but energy, we perform simulations over NVIDIA RTX8000. For energy computation experiments, we use the Jetson TX2 module occupied with one NVIDIA Pascal GPU to model the edge computing environment in practice for IoT devices. In all of our evaluations, we use mini-batch sized 128, five local computation rounds with ten clients communicating with one parameter server in the FL setup. Besides, since the convergence variables depend on the hyperparameters, we empirically choose their values satisfying Assumption 4 with ϵ as 10^{-5}, β as 0.9, and θ as 0.999 with an initial learning rate of 0.001 for MNIST and 0.005 for CIFAR-10 with step size $\alpha/\sqrt{t}$ consistent with the theory.

B. Compression

A challenge persists in choosing the compression levels of each of the methods in GWEP, since there exists a tradeoff between compression and learning performance. In order to overcome this, we use the theoretical analysis from our derived theorem to assess the impact of pruning and quantization levels on convergence. Hence, we elucidate the results of varying the sparsity levels of the model and quantization bits used for weight and gradient quantizations. To verify Remark 1 that pruning level affects the convergence rate, we test the S_f values incrementally between 0.01 and 0.98 to achieve the desired compression and obtain between 10% and 98% sparse models, respectively. For results in Fig. 2(a), we trained LeNet5 on MNIST for 60 communication rounds and set the initial sparsity value S_i as 0 and increased the sparsity gradually following (5) to achieve the desired S_f. As observed,

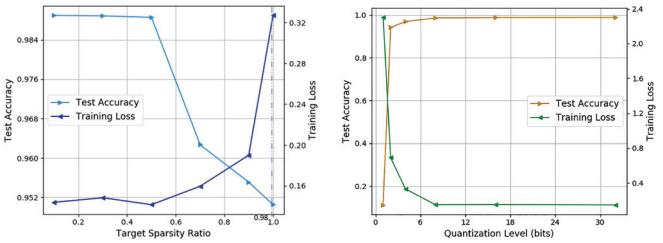

Fig. 2. Performance of GWEP at different compression levels. (a) GWEP at different sparsity levels. (b) GWEP at different quantized bits.

even at an extremely high sparsity level of 98%, with model redundancy significantly removed, GWEP exhibits a reasonable performance of around 95% test accuracy on MNIST. Correspondingly, on ResNet20, we observed 85.09% on a 98% sparse model. Further, in the realm of Definition 3, we use the same training settings as above and alter the quantization bits from 1 to 32 to evaluate its impact on model performance. As seen in Fig. 2(b), for an 8-bit quantization and above, we achieve 98.62% top-1 accuracy and for 2 bits, the accuracy is still reasonable at 94.2%. Hence, the higher bit quantization setting performs better which is also shown in the theoretical convergence analysis. Thus, finding a balance between performance and communication time, we set 8-bit quantization levels implementing the error feedback.

C. Convergence Analysis

For the main results, we train ResNet20 over CIFAR-10 for 200 communication rounds and LeNet5 over MNIST for 100 communication rounds, setting the target sparsity ratio as 0.5, targeting a resulting 50% sparse model. The pruning frequency is set to 2 as a large value has a negligible impact on the final model quality. We set 8-bit quantization levels for both weight and gradient quantizations, implementing the error feedback term. For ease of comparison, in the baselines, we set 8-bit quantization in QAdam and 50% sparsity in DPF. From the convergence performance of different methods on MNIST LeNet5 shown in Fig. 3(a), GWEP has a convergence rate comparable with FedAvg and is considerably faster than DPF. The convergence of QAdam is close to GWEP, albeit slower. A similar convergence trend is observed on training ResNet20 over CIFAR-10 as shown in Fig. 3(c). The performance accuracy of GWEP mainly exceeds the baseline values as shown in Fig. 3(b) for MNIST and Fig. 3(d) for CIFAR. In fact, a small improvement in the accuracy is seen which can be attributed to a reduction in overfitting and increased generalization due to unstructured pruning. This is also in line with the finding in [12], [27], etc. Furthermore, we examine the performance of GWEP by scaling the number of clients in FL to 100 and 250 and observe that the number of training rounds remains nearly the same and the training time is still reasonable between 6 and 10 min, with high top-1 accuracy for 100 training rounds on LeNet5 on MNIST as shown in Fig. 5(d).

D. Discussion on Time and Energy

We record the time as the time taken for training the respective models using different methods. Since we perform the

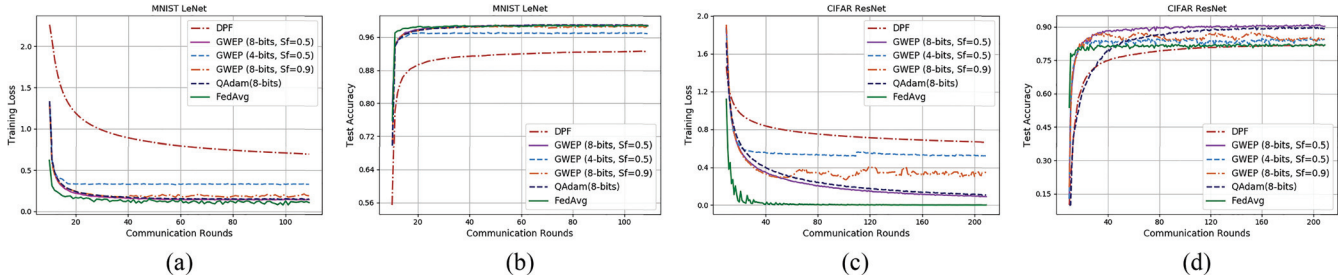

Fig. 3. Training loss and test accuracy showing convergence rate of different methods for fixed number of communication rounds. (a) Training loss on MNIST. (b) Test accuracy on MNIST. (c) Training loss on CIFAR. (d) Test accuracy on CIFAR.

TABLE II
PERFORMANCE EVALUATION OF DIFFERENT METHODS FOR FIXED NUMBER OF COMMUNICATION ROUNDS

Methods	LeNet5 Acc.	$T = 100$ Time (s)	ResNet20 Acc.	$T = 200$ Time (s)
GWEP	**98.89%**	**313.2**	**90.9%**	**2534.4**
QAdam	98.82%	396	89.01%	2869.2
DPF	92.63%	396	81.95%	2912.4
FedAvg	98.86%	316.8	81.34%	5396.4

TABLE III
PERFORMANCE EVALUATION FOR PRESET TEST ACCURACY

Methods	LeNet5 T	97% acc. Time (s)	ResNet20 T	82% acc. Time (s)
GWEP	**12**	**116.9**	**18**	**719.5**
QAdam	20	151.6	48	954.4
DPF	130	479.2	198	7365.9
FedAvg	30	117.3	92	4645.9

tests in simulated settings, the time values technically comprise both local computation and communication time. To record the time values, first, we fix the communication rounds T similar to the ones in the previous sections as 100 for MNIST and 200 for CIFAR-10 and summarize the results in Table II. Next, instead of fixing the rounds, we set an attainable accuracy value of 97% for MNIST and 82% for CIFAR-10 and when this preset accuracy is met, we tabulate the results in Table III. As observed, our algorithm outperforms the baseline methods in both the cases, both in terms of overall time and test accuracy. With the regularization properties of network pruning, GWEP even outperforms the corresponding dense model on ResNet20. For LeNet5 trained on MNIST, GWEP performs better than the baselines and attains nearly the same test accuracy as the dense model, FedAvg. Hence, GWEP not only reduces the computation and communication cost but also retains the performance of the original dense model. Using our algorithm, the real gain comes from faster inference and accelerated training. On ResNet20, GWEP is more than $\times 10$ faster than the baseline DPF and $\times 6.45$ faster than its dense counterpart, FedAvg as shown in Fig. 4(a). GWEP also consumes fewer communication rounds of less than half the number that original FedAvg takes and 90% lesser than DPF and 60% lesser than QAdam. The energy values of GWEP are lesser up to $\times 6$ in comparison with the baselines as seen in Fig. 4(b). Overall, our method reduces the time and energy drastically, without compromising on the learning performance.

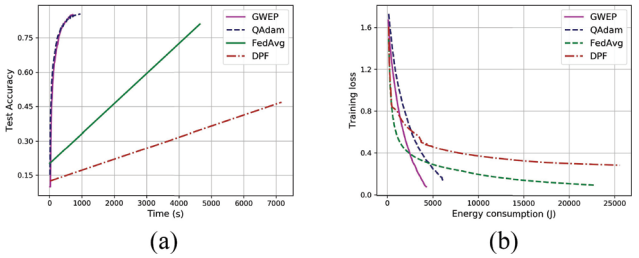

Fig. 4. Time and energy values on ResNet20 trained on CIFAR-10. (a) Accuracy versus time. (b) Loss versus consumed energy.

E. Communication Efficiency

With sufficient compression, the main gain of GWEP is the reduction in communication cost. We fix the test accuracy to be attained by all the methods as 97% for MNIST and 82% for CIFAR-10 and observe the number of communication rounds required by each method. As shown in Fig. 5(a) and (b), GWEP substantially reduces the number of communication rounds required for training. T values for MNIST on GWEP are smaller by $\times 10.83$, $\times 2.5$, and $\times 1.6$ than the baselines DPF, FedAvg, and QAdam. Similarly, smaller on CIFAR-10 by $\times 11$, $\times 5.1$, and $\times 2.6$ than DPF, FedAvg, and QAdam, respectively. Furthermore, since FL mainly deals with data heterogeneity across devices, we extend the tests on non-IID data. Without additional tuning of hyperparameters, we run MNIST with non-IID data on ten clients for 1000 communication rounds. MNIST data are first sorted by labels and then partitioned into 200 shards of size 300 and each client is assigned 20 shards, resulting in non-IID data distribution among the clients. As shown in Fig. 5(c), GWEP outperforms the baseline methods with a higher test accuracy of 87.25% and faster convergence. This added communication efficiency makes our method suitable for resource-constrained environments such as IoT devices for edge training.

As seen from the simulation results, GWEP exhibits a substantial saving in the overall training time which includes the time taken at all stages of FL. This includes the local computation on each user device, communication of gradients to the server by all clients, and transmission of global model updates from the server to all connected clients. We attribute this speedup to our method adopting model compression in threefold. First, as a result of quantization, communication of the quantized parameters is faster since they comprise fewer bits than the full bit-sized model parameters. This has a dual

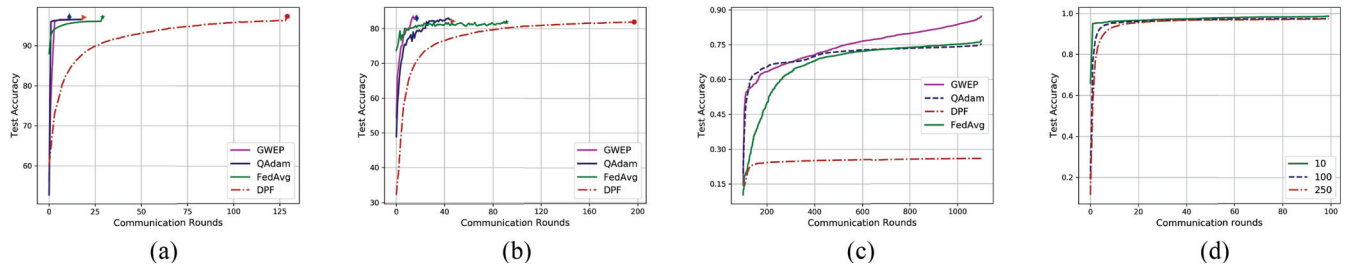

Fig. 5. Communication rounds of GWEP in comparison with different methods on MNIST and CIFAR-10 for fixed accuracy values and non-IID data sets. (a) Communication rounds on MNIST. (b) Communication rounds on CIFAR-10. (c) Performance on non-IID data. (d) Scaling GWEP's number of clients.

advantage of not only reducing the communication time and cost but also reducing the memory bandwidth due to less memory accesses. Second, by setting the unimportant weights of the model to zero through pruning, subsequent training and computations are based on the remaining important weights. GWEP also inherits the properties of pruning and quantization of faster inference, reduction of memory accesses, and bandwidth due to the smaller-sized fixed-point model parameters and gradients. Further, in our method, since we prune all the layers unlike most other works where pruning is primarily focused on the fully connected layers, GWEP results in a more sparse model. Therefore, the local computation time and cost are reduced, accelerating the training. Hence, by coalescing this method with the FL process, we successfully achieve a reduction in communication cost and time, which is a necessity in the time-sensitive MEC environment. Likewise, the reduction in computation requirements and memory accesses makes it suitable for resource and memory-constrained IoT devices. In essence, these strengthen the goals of accelerating the overall time taken using GWEP and making way for low-end devices to be able to participate in the FL process.

VII. Related Work

Different aspects of IoT devices and FL have been studied extensively in the literature and several works focus on various challenges posed by them. For example, from the privacy perspective, Zheng *et al.* [28] highlighted the potential threats and privacy issues of linkable data in smart IoT systems and Xiong *et al.* [29] used differential privacy for privacy protection in FL with non-IID data. Moreover, Pang *et al.* [30] focused on heterogeneity by implementing a reinforcement learning (RL)-based intelligent central server with the capability of recognizing heterogeneity. In order to tackle the overall training delay in FL over mobile devices, Prakash *et al.* [31] balanced the tradeoff between wireless communication and local computation. Furthermore, in the context of model compression in FL and in edge networks, previous works have used different compression techniques in the FL settings. Sattler *et al.* [32] proposed sparse ternary compression to enable downstream compression for gradient sparsification. Li *et al.* [33] focused on energy-efficient FL over mobile-edge devices by adopting flexible top-k with dynamic batches to achieve gradient sparsification. Lossy compression along with Federated Dropout was used for reducing communication bottleneck in [34]. Wang *et al.* [35] used atomic decomposition of the given gradient for communication-efficient learning. Deep gradient compression to reduce the communication bandwidth was proposed in [36].

While pruning is now a popular model compression technique, the earliest work on model pruning was performed in [37] where parameters were pruned based on saliency and the network was retrained. The process was further improved in state of the art in [12] with one-shot pruning of a dense model followed by fine-tuning. Based on this are many recent works such as [38] with variations. Guerra *et al.* [39] proposed auto-pruning techniques where the former prunes the network by optimizing a set of auxiliary parameters instead of the original weights. Model pruning in coordination with the FL training procedure was introduced in [19] where the clients share samples from the private data sets with the server. However, while this retains the performance accuracy, it defeats the purpose of FL's guarantee of training data privacy. Besides, their proposed sample-less pruning affects the performance of the model negatively. In our work, however, the data are maintained at the client providing privacy guarantees.

Few notable works on weight quantization are BinaryConnect [40] where each weight is binarized using the sign function and [15], where a threshold-based ternary function is optimized to get an approximated solution. Reference [41] which used stochastically quantized gradients on the unbiased gradient and [16] which utilized a general stochastic linear quantization are popular works in gradient quantization. A combination of quantization and pruning is seen in works, such as [39] which prunes a quantized neural network and [42] which performs pruning and quantization in parallel. Our pruning methodology is more close to [9] which proposed magnitude-based pruning during training by increasing the sparsity ratio gradually and [17], where the dynamic model pruning is performed with an error feedback that dynamically adapts the mask at every few iterations. However, in our work, we do not maintain a simultaneous dense model, and we use a different reparameterization strategy.

VIII. Conclusion

In this article, we introduced an IoT device friendly and communication-efficient FL algorithm named, GWEP, via threefold model compression. By performing model compression through joint model pruning and quantization of model parameters and gradients, we illustrated that our proposed approach can effectively reduce storage, communication, and computation requirements, accelerating the training process. We theoretically analyzed the FL convergence and discussed

the extension to other suitable methods. Through empirical evaluations, we demonstrated that our approach significantly outperforms the baseline methods in terms of reducing the training time, communication rounds, and energy, while still achieving high test accuracy and compression. The results have shown that our FL method exhibits a great potential in accommodating originally large-sized DNNs over resource-constrained IoT devices.

APPENDIX A
PROOF OF LEMMA 3

Proof: Based on $\hat{v}_t^n = \theta_t \hat{v}_{t-1}^n + (1-\theta_t)[\sigma_t^{n^2}]$, and $\sigma_t^{n^2} = \mathbb{E}[g_t^{n^2}]$, we have

$$\left\|\hat{v}_t^n\right\|_1 = \theta_t \left\|\hat{v}_{t-1}^n\right\|_1 + (1-\theta_t)\left\|\sigma_t^n\right\|_1^2. \tag{9}$$

Then, using Assumption 1, we have the inequality, $\|\hat{v}_t^n\|_1 \le G^2$. Therefore, by induction, the following inequality holds:

$$\left\|\hat{v}_t^n + \epsilon\right\|_1 \le G^2 + \epsilon d. \tag{10}$$

Using this inequality, we can express the gradient estimation over N clients as follows:

$$\begin{aligned}
\left\|\nabla f\left(\dot{w}_t\right)\right\|^2 &= \frac{1}{N}\sum_{n=1}^{N} \frac{\left\|\nabla f\left(\dot{w}_t\right)\right\|^2}{\sqrt{\left\|\hat{v}_t^n + \epsilon\right\|_1}} \sqrt{\left\|\hat{v}_t^n + \epsilon\right\|_1} \\
&= \frac{1}{N}\sum_{n=1}^{N} \sqrt{\left\|\hat{v}_t^n + \epsilon\right\|_1} \sum_{j=1}^{d} \frac{\left|\nabla_j f\left(\dot{w}_t\right)\right|^2}{\sqrt{\sum_{j=1}^{d} \hat{v}_{t,j}^n + \epsilon}} \\
&\le \frac{1}{N}\sum_{n=1}^{N} \sqrt{\left\|\hat{v}_t^n + \epsilon\right\|_1} \alpha_t^{-1} \sum_{j=1}^{d} \frac{\alpha_t}{\sqrt{\hat{v}_{t,j}^n + \epsilon}} \left|\nabla_j f\left(\dot{w}_t\right)\right|^2 \\
&= \frac{1}{N}\sum_{n=1}^{N} \sqrt{\left\|\hat{v}_t^n + \epsilon\right\|_1} \alpha_t^{-1} \sum_{j=1}^{d} \hat{\eta}_{t,j}^n \left|\nabla_j f\left(\dot{w}_t\right)\right|^2 \\
&\le \frac{\sqrt{G^2 + \epsilon d}}{\alpha_t N} \sum_{n=1}^{N} \left\|\nabla f\left(\dot{w}_t\right)\right\|_{\hat{\eta}_t^n}^2.
\end{aligned} \tag{11}$$

Equation (11) is obtained using (10) and the weighted norm of the step size relation of $\hat{\eta}_t^n$. Then, with the definition of α_t and for t chosen randomly from $\{1, 2, \ldots, T\}$, we can thus deduce

$$\mathbb{E}\left[\left\|\nabla f\left(\dot{w}_t\right)\right\|^2\right] \le \frac{\sqrt{G^2 + \epsilon d}}{\alpha\sqrt{T}N} \sum_{n=1}^{N} \mathbb{E}\left[\sum_{t=1}^{T} \left\|\nabla f\left(\dot{w}_t\right)\right\|_{\hat{\eta}_t^n}^2\right]. \qquad \blacksquare$$

APPENDIX B
PROOF OF LEMMA 4

Proof: Using the definitions, $v_t^n = \theta_t v_{t-1}^n + (1-\theta_t)[g_t^n]^2$ and $\hat{v}_t^n = \theta_t v_{t-1}^n + (1-\theta_t)[\sigma_t^{n^2}]$ and steps close to [23] and [24], we obtain the following results:

$$\begin{aligned}
&\Delta_t^n - \frac{\beta\alpha_t}{\sqrt{\theta_t}\alpha_{t-1}} \Delta_{t-1}^n \\
&\quad = -(1-\beta)\hat{\eta}_t^n g_t^n + \mathcal{P} + \mathcal{Q} + \mathcal{R} + \mathcal{S} + \mathcal{T} \\
&\quad \mathcal{P} = \hat{\eta}_t^n g_t^n \frac{(1-\theta_t) g_t^n}{\sqrt{v_t^n + \epsilon}} \left[\frac{\beta m_{t-1}^n}{\sqrt{v_t^n + \epsilon}\sqrt{\theta_t v_{t-1}^n + \theta_t \epsilon}}\right.
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}\left\langle \nabla f\left(\dot{w}_t\right), \Delta_t \right\rangle &= \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}\left\langle \nabla f\left(\dot{w}_t\right), \Delta_t^n \right\rangle \\
&= \frac{1}{N}\sum_{n=1}^{N} \frac{\beta\alpha_t}{\sqrt{\theta_t}\alpha_{t-1}} \mathbb{E}\left\langle \nabla f\left(\dot{w}_t\right), \Delta_{t-1}^n \right\rangle + \mathbb{E}\left\langle \nabla f\left(\dot{w}_t\right), \Delta_t^n - \frac{\beta\alpha_t}{\sqrt{\theta_t}\alpha_t} \Delta_{t-1}^n \right\rangle \\
&= \frac{1}{N}\sum_{n=1}^{N} \frac{\beta\alpha_t}{\sqrt{\theta_t}\alpha_{t-1}} \left[\mathbb{E}\left\langle \nabla f\left(\dot{w}_{t-1}\right), \Delta_{t-1}^n \right\rangle + \mathbb{E}\left\langle \nabla f(\dot{w}_t) - \nabla f(\dot{w}_{t-1}), \Delta_{t-1}^n \right\rangle\right] + \mathbb{E}\left\langle \nabla f(\dot{w}_t), -(1-\beta)\hat{\eta}_t^n g_t^n \right\rangle \\
&\quad + \mathbb{E}\left\langle \nabla f(\dot{w}_t), \mathcal{P} \right\rangle + \mathbb{E}\left\langle \nabla f(\dot{w}_t), \mathcal{Q} \right\rangle + \mathbb{E}\left\langle \nabla f(\dot{w}_t), \mathcal{R} \right\rangle + \mathbb{E}\left\langle \nabla f(\dot{w}_t), \mathcal{S} \right\rangle + \mathbb{E}\left\langle \nabla f(\dot{w}_t), \mathcal{T} \right\rangle
\end{aligned} \tag{12}$$

$$\begin{aligned}
&\sum_{t=1}^{T}\left[\frac{1}{N}\sum_{n=1}^{N} \frac{\beta\alpha_t}{\sqrt{\theta_t}\alpha_{t-1}} \left[\mathbb{E}\left\langle \nabla f(\dot{w}_{t-1}), \Delta_{t-1}^n \right\rangle + \mathbb{E}\left\langle \nabla f(\dot{w}_t) - \nabla f(\dot{w}_{t-1}), \Delta_{t-1}^n \right\rangle\right]\right] \\
&\le \sum_{t=1}^{T}\left[\frac{1}{N}\sum_{n=1}^{N} \frac{\beta\alpha_t}{\sqrt{\theta_t}\alpha_{t-1}} \left[\mathbb{E}\left\langle \nabla f(\dot{w}_{t-1}), \Delta_{t-1}^n \right\rangle + \left\{L\mathbb{E}\left[\left\|(w_t - \hat{w}_t) - \dot{w}_t\right\|\right] + L\mathbb{E}\left[\left\|(w_{t-1} - \hat{w}_{t-1}) - \dot{w}_{t-1}\right\|\right]\right.\right. \\
&\qquad \left.\left. + L\mathbb{E}\left[\left\|w_t - \hat{w}_t\right\|\right] + L\mathbb{E}\left[\left\|w_{t-1} - \hat{w}_{t-1}\right\|\right] + L\mathbb{E}\left[\left\|w_t - w_{t-1}\right\|\right]\right\}\left\|\Delta_{t-1}^n\right\|\right]\right] \\
&\le \sum_{t=1}^{T}\left[\frac{\beta\alpha_t}{\sqrt{\theta_t}\alpha_{t-1}}\left[\frac{1}{N}\sum_{n=1}^{N} \mathbb{E}\left\langle \nabla f(\dot{w}_{t-1}), \Delta_{t-1}^n \right\rangle + \frac{1}{N^2} L(2-\tau_g) \sum_{n=1}^{N} \mathbb{E}\left\|\Delta_{t-1}^n\right\|^2\right.\right. \\
&\qquad \left.\left. + \frac{1}{N^2} L(2-\tau_g) \sum_{n=1}^{N} \mathbb{E}\left\|\Delta_{t-1}^n\right\|\left\|e_{t-1}^n\right\| + \frac{1}{N} 2(1-\tau_w) LD \sum_{n=1}^{N} \mathbb{E}\left\|\Delta_{t-1}^n\right\| + \frac{1}{N} 2\sqrt{\tau_p}(1-\tau_w) LD \sum_{n=1}^{N} \mathbb{E}\left\|\Delta_{t-1}^n\right\|\right]\right]
\end{aligned} \tag{13}$$

$$\mathbb{E}\left\langle \nabla f(\dot{w}_t), -(1-\beta)\hat{\eta}_t^n g_t^n \right\rangle = -(1-\beta)\sum_{t=1}^{T}\left[\mathbb{E}\left\langle \nabla f(\dot{w}_t), \hat{\eta}_t^n g_t^n \right\rangle\right] = -\frac{(1-\beta)}{N}\sum_{t=1}^{T}\left[\sum_{n=1}^{N} \mathbb{E}\left[\left\|\nabla f\left(\dot{w}_t\right)\right\|_{\hat{\eta}_t^n}^2\right]\right] \tag{14}$$

$$+\frac{(1-\beta)g_t^n}{\sqrt{v_t^n+\epsilon}+\sqrt{\hat{v}_t^n+\epsilon}}\Bigg]$$

$$\mathcal{Q}=-\hat{\eta}_t^n\sigma_t^n\frac{(1-\theta_t)g_t^n}{\sqrt{v_t^n+\epsilon}}\frac{(1-\beta)\sigma_t^n}{\sqrt{v_t^n+\epsilon}+\sqrt{\hat{v}_t^n+\epsilon}}$$

$$\mathcal{R}=\hat{\eta}_t^n\epsilon\frac{(1-\theta_t)}{\sqrt{v_t^n+\epsilon}}\frac{\beta m_{t-1}^n}{\sqrt{v_t^n+\epsilon}\sqrt{\theta_t v_{t-1}^n+\theta_t\epsilon}}$$

$$\mathcal{S}=\hat{\eta}_t^n\sqrt{\sigma_t^{n^2}+\epsilon}\frac{(1-\theta_t)g_t^n}{\sqrt{v_t^n+\epsilon}}\frac{\beta m_{t-1}^n}{\sqrt{\theta_t v_{t-1}^n+\theta_t\epsilon}}\frac{\sqrt{1-\theta_t}g_t^n}{\sqrt{v_t^n+\epsilon}+\sqrt{\theta_t v_{t-1}^n+\theta_t\epsilon}}\frac{\sqrt{1-\theta_t}\sqrt{\sigma_t^{n^2}+\epsilon}}{\sqrt{\hat{v}_t^n+\epsilon}+\sqrt{\theta_t v_{t-1}^n+\theta_t\epsilon}}$$

$$\mathcal{T}=\hat{\eta}_t^n\sqrt{\sigma_t^{n^2}+\epsilon}\frac{(1-\theta_t)\sqrt{\epsilon}}{\sqrt{v_t^n+\epsilon}}\frac{\beta m_{t-1}^n}{\sqrt{\theta_t v_{t-1}^n+\theta_t\epsilon}}\frac{\sqrt{1-\theta_t}\sqrt{\epsilon}}{\sqrt{v_t^n+\epsilon}+\sqrt{\theta_t v_{t-1}^n+\theta_t\epsilon}}\frac{\sqrt{1-\theta_t}\sqrt{\sigma_t^{n^2}+\epsilon}}{\sqrt{\hat{v}_t^n+\epsilon}+\sqrt{\theta_t v_{t-1}^n+\theta_t\epsilon}}.$$

Using the values from (7), (6), and Assumption 3, we obtain (13), shown at the bottom of the previous page, for the first term in (12), shown at the bottom of the previous page. Similarly, since the learning rate is independent of the gradient, the second term of (12) can be obtained as (14), shown at the bottom of the previous page, and the remaining terms follow similar proof from [24]. Using these results, we can hence derive the upper bound for the term $\mathbb{E}\langle\nabla f(\grave{w}_t),\Delta_t\rangle$ as

$$\mathbb{E}\big\langle\nabla f(\grave{w}_t),\Delta_t\big\rangle\le\sum_{t=1}^{T}\Bigg[\frac{\beta\alpha_t}{\sqrt{\theta_t}\alpha_{t-1}}M_{t-1}+\mathcal{B}(1-\theta_t)-\frac{1}{N}\sum_{n=1}^{N}\frac{1-\beta}{2}\mathbb{E}\Big[\big\|\nabla f(\grave{w}_t)\big\|^2_{\hat{\eta}_t^n}\Big]\Bigg] \tag{19}$$

where the value of $\mathcal{B}$ is derived as in [24] as (16), shown at the bottom of the page.

Using the above results and (19) in the definition of M_t, (15), shown at the bottom of the page, can be deduced. Using the results from Lemmas 1–3 in (15), we can thus show the upper estimate of M_t as (8). ■

Appendix C
Proof of Theorem 1

Proof: With $\tilde{w}_t=w_t-(1/N)\sum_{n=1}^{N}e_t^n$, we can write

$$\tilde{w}_{t+1}=w_{t+1}-\frac{1}{N}\sum_{n=1}^{N}e_{t+1}^n. \tag{20}$$

From Algorithm 1, we have $w_{t+1}=w_t-\hat{g}_t$, $e_{t+1}^n=\Delta_t^n+e_t^n-\hat{g}_t^n$, and $\hat{g}_t=\frac{1}{N}\sum_{n=1}^{N}\hat{g}_t^n$. Using the above relations and the definition of Δ_t, we can write (20) as (21)

$$\tilde{w}_{t+1}=w_t-\hat{g}_t-\frac{1}{N}\sum_{n=1}^{N}\big(\Delta_t^n+e_t^n-\hat{g}_t^n\big)$$

$$\sum_{t=1}^{T}M_t\le\frac{1}{\sqrt{\mathcal{A}}(1-\sqrt{\gamma})}\Bigg[\sum_{t=1}^{T}\frac{1}{N}L(2-\tau_g)\sum_{n=1}^{N}\mathbb{E}\big\|\Delta_{t-1}^n\big\|^2+\frac{1-\tau_g}{\tau_g N}L(2-\tau_g)\sum_{n=1}^{N}\mathbb{E}\big\|\Delta_{t-1}^n\big\|^2+\frac{2(1-\tau_w)LD}{N}\sum_{n=1}^{N}\mathbb{E}\big\|\Delta_{t-1}^n\big\|+\frac{2\sqrt{\tau_p}(1-\tau_w)LD}{N}\sum_{n=1}^{N}\mathbb{E}\big\|\Delta_{t-1}^n\big\|+\mathcal{B}(1-\theta_t)\Bigg]-\frac{(1-\beta)}{2N}\sum_{n=1}^{N}\mathbb{E}\sum_{t=1}^{T}\Big[\big\|\nabla f(\grave{w}_t)\big\|^2_{\hat{\eta}_t^n}\Big] \tag{15}$$

$$\mathcal{B}=\frac{5\alpha G^3(1-\beta)}{2\epsilon\sqrt{\theta}}\Bigg[\frac{\beta}{(1-\beta)\sqrt{\theta_1\mathcal{A}(1-\gamma)}}+1\Bigg]^2+\frac{5\alpha G^3}{2\epsilon\sqrt{\theta}}+\frac{5\alpha d\sqrt{\epsilon}\beta^2}{2\sqrt{\theta}(1-\beta)\theta_1\mathcal{A}(1-\gamma)}+\frac{5\alpha\sqrt{G^2+\epsilon}G^2\beta^2}{2\sqrt{\theta}(1-\beta)\theta_1\mathcal{A}(1-\gamma)\epsilon}+\frac{5\alpha\sqrt{G^2+\epsilon}\beta^2 d}{2\sqrt{\theta}(1-\beta)\theta_1\mathcal{A}(1-\gamma)} \tag{16}$$

$$\mathbb{E}\big[f(\tilde{w}_{t+1})\big]\le\mathbb{E}\Bigg[f(\tilde{w}_t)+\frac{1}{N}\sum_{n=1}^{N}\big\langle\nabla f(\tilde{w}_t),\Delta_t^n\big\rangle+\frac{L}{2N^2}\sum_{n=1}^{N}\big\|\Delta_t^n\big\|^2+\frac{1}{N}\sqrt{\tau_p}(1-\tau_w)LD\sum_{n=1}^{N}\big\|\Delta_t^n\big\|+\frac{1}{N}(1-\tau_w)LD\sum_{n=1}^{N}\big\|\Delta_t^n\big\|+\frac{L}{N^2}\sum_{n=1}^{N}\big\|e_t^n\big\|\big\|\Delta_t^n\big\|\Bigg]\le\mathbb{E}\big[f(\tilde{w}_t)\big]+M_t \tag{17}$$

$$\mathbb{E}\Big[\big\|\nabla f(\grave{w}_t)\big\|^2\Big]\le\frac{\sqrt{G^2+\epsilon d}}{\alpha\sqrt{T}N}\sum_{n=1}^{N}\mathbb{E}\Bigg[\sum_{t=1}^{T}\big\|\nabla f(\grave{w}_t)\big\|^2_{\hat{\eta}_t^n}\Bigg]\le\frac{2\sqrt{G^2+\epsilon d}}{(1-\beta)\alpha\sqrt{T}}\Bigg[f(w_1)-f^*+\frac{1}{\sqrt{\mathcal{A}}(1-\sqrt{\gamma})}\Bigg[\frac{L(2-\tau_g)G^2\alpha^2}{\epsilon\tau_g}+\mathcal{B}\theta\sum_{t=1}^{T}\frac{1}{t}+\frac{4(1-\tau_w)LDG\alpha\sqrt{T}}{\sqrt{\epsilon}}\big(1+\sqrt{\tau_p}\big)\Bigg]\Bigg] \tag{18}$$

$$= \left(w_t - \frac{1}{N}\sum_{n=1}^{N} e_t^n\right) - \hat{g}_t + \hat{g}_t - \Delta_t$$
$$= \tilde{w}_t - \Delta_t. \tag{21}$$

Considering that f and the stochastic gradients g_t satisfy Assumption 1, that is, by the Lipschitz continuity of the gradient of f and Schwartz inequality, we obtain

$$\begin{aligned} f(\tilde{w}_{t+1}) &\leq f(\tilde{w}_t) + \langle \nabla f(\tilde{w}_t), \Delta_t \rangle + \frac{L}{2}\|\Delta_t\|^2 \\ &= f(\tilde{w}_t) + \left\langle \nabla f(\hat{w}_t), \Delta_t \right\rangle + \frac{L}{2}\|\Delta_t\|^2 \\ &\quad + \left\langle \left(\nabla f(w_t) - \nabla f\left(\hat{w}_t\right)\right) - \nabla f(\hat{w}_t), \Delta_t \right\rangle \\ &\quad + \left\langle \nabla f(w_t) - \nabla f(\hat{w}_t), \Delta_t \right\rangle + \langle \nabla f(\tilde{w}_t) - \nabla f(w_t), \Delta_t \rangle. \end{aligned} \tag{22}$$

Using the definitions of quantization and pruning parameters as in (7), (6), Assumption 3, and the upper estimate as in (8), we obtain (17), shown at the bottom of the previous page. Taking summation on both sides of the inequality, the following holds:

$$f^* \leq \mathbb{E}\left[f(\tilde{w}_{t+1})\right] \leq f(w_1) + \sum_{t=1}^{T} M_t. \tag{23}$$

Rearranging the terms above as per Lemma 3 and using the value of M_t from (8), we can hence derive (18), shown at the bottom of the previous page. Thereby proving the theorem, $\mathbb{E}[\|\nabla f(\hat{w}_t)\|^2] \leq ([\mathcal{C} + \mathcal{D}\sum_{t=1}^{T}(1/t)]/\sqrt{T}) + \mathcal{E}$. ■

REFERENCES

[1] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, "Federated learning for Internet of Things: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 1622–1658, 3rd Quart., 2021.

[2] P. Prakash, J. Ding, H. Li, S. M. Errapotu, Q. Pei, and M. Pan, "Privacy preserving facial recognition against model inversion attacks," in *Proc. GLOBECOM IEEE Global Commun. Conf.*, Taipei, Taiwan, Dec. 2020, pp. 1–6.

[3] M. Wu *et al.*, "Evaluation of inference attack models for deep learning on medical data," 2020, *arXiv:2011.00177*.

[4] H. B. McMahan, E. Moore, D. Ramage, S. Hampson and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016, *arXiv:1602.05629*.

[5] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "FedHealth: A federated transfer learning framework for wearable healthcare," *IEEE Intell. Syst.*, vol. 35, no. 4, pp. 83–93, Jul.-Aug. 2020.

[6] 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. 23920–23935, 2020.

[7] A. Hard *et al.*, "Federated learning for mobile keyboard prediction," 2018, *arXiv:1811.03604*.

[8] D. Shi, L. Li, R. Chen, P. Prakash, M. Pan, and Y. Fang, "Towards energy efficient federated learning over 5G+ mobile devices," 2021, *arXiv:2101.04866*.

[9] M. H. Zhu and S. Gupta, "To prune, or not to prune: Exploring the efficacy of pruning for model compression," in *Proc. 6th Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, Apr. 2018. [Online]. Available: https://openreview.net/forum?id=Sy1iIDkPM

[10] D. Gao, X. He, Z. Zhou, Y. Tong, K. Xu, and L. Thiele, "Rethinking pruning for accelerating deep inference at the edge," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2020, pp. 155–164.

[11] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *Proc. NIPS Workshop Private Multi-Party Mach. Learn.*, Barcelona, Spain, Dec. 2016. [Online]. Available: https://arxiv.org/abs/1610.05492

[12] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural network," in *Proc. 29th Conf. Neural Inf. Process. Syst.*, Montreal, QC, Canada, Dec. 2015, pp. 1135–1143.

[13] P. Goyal *et al.*, "Accurate, large minibatch SGD: Training imagenet in 1 hour," 2017, *arXiv:1706.02677*.

[14] L. Hou, R. Zhang, and J. T. Kwok, "Analysis of quantized models," in *Proc. 7th Int. Conf. Learn. Represent.*, New Orleans, LA, USA, May 2019. [Online]. Available: https://openreview.net/forum?id=ryMIoAqYX

[15] F. Li, B. Zhang, and B. Liu, "Ternary weight networks," 2016, *arXiv:1605.04711*.

[16] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "QSGD: Communication-efficient SGD via gradient quantization and encoding," in *Proc. 31st Conf. Neural Inf. Process. Syst.*, Long Beach, CA, USA, Dec. 2017, pp. 1709–1720.

[17] T. Lin, S. U. Stich, L. Barba, D. Dmitriev, and M. Jaggi, "Dynamic model pruning with feedback," in *Proc. Int. Conf. Learn. Represent.*, Addis Ababa, Ethiopia, Apr. 2020. [Online]. Available: https://openreview.net/forum?id=SJem8lSFwB

[18] C. Philippenko and A. Dieuleveut, "Bidirectional compression in heterogeneous settings for distributed or federated learning with partial participation: Tight convergence guarantees," 2020, *arXiv:2006.14591*.

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

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, San Diego, CA, USA, May 2015. [Online]. Available: https://openreview.net/forum?id=8gmWwjFyLj

[21] S. P. Karimireddy, Q. Rebjock, S. U. Stich, and M. Jaggi, "Error feedback fixes SignSGD and other gradient compression schemes," in *Proc. 36th Int. Conf. Mach. Learn. (ICML)*, Long Beach, CA, USA, Jun. 2019, pp. 3252–3261.

[22] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, "Sparsified SGD with memory," in *Proc. 32nd Conf. Neural Inf. Process. Syst.*, Montreal, QC, Canada, Dec. 2018, pp. 4452–4463.

[23] F. Zou, L. Shen, Z. Jie, W. Zhang, and W. Liu, "A sufficient condition for convergences of adam and RMSprop," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Long Beach, CA, USA, Jun. 2019, pp. 11127–11135.

[24] C. Chen, L. Shen, H. Huang, Q. Wu, and W. Liu, "Quantized adam with error feedback," 2020, *arXiv:2004.14180*.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.

[26] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[27] D. Blalock, J. J. G. Ortiz, J. Frankle, and J. Guttag, "What is the state of neural network pruning?" 2020, *arXiv:2003.03033*.

[28] X. Zheng, Z. Cai, and Y. Li, "Data linkage in smart Internet of Things systems: A consideration from a privacy perspective," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 55–61, Sep. 2018.

[29] Z. Xiong, Z. Cai, D. Takabi, and W. Li, "Privacy threat and defense for federated learning with non-IID data in AIoT," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1310–1321, Feb. 2022.

[30] J. Pang, Y. Huang, Z. Xie, Q. Han, and Z. Cai, "Realizing the heterogeneity: A self-organized federated learning framework for IoT," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3088–3098, Mar. 2021.

[31] P. Prakash, J. Ding, M. Wu, M. Shu, R. Yu, and M. Pan, "To talk or to work: Delay efficient federated learning over mobile edge devices," 2021, *arXiv:2111.00637*.

[32] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-IID. data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3400–3413, Sep. 2020.

[33] L. Li, D. Shi, R. Hou, H. Li, M. Pan, and Z. Han, "To talk or to work: Flexible communication compression for energy efficient federated learning over heterogeneous mobile edge devices," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, May 2021, pp. 1–10.

[34] S. Caldas, J. Konečny, H. B. McMahan, and A. Talwalkar, "Expanding the reach of federated learning by reducing client resource requirements," 2018, *arXiv:1812.07210*.

[35] H. Wang, S. Sievert, S. Liu, Z. Charles, D. Papailiopoulos, and S. Wright, "ATOMO: Communication-efficient learning via atomic sparsification," in *Proc. 32nd Conf. Neural Inf. Process. Syst.*, Montreal, QC, Canada, Dec. 2018, pp. 9872–9883.

[36] Y. Lin, S. Han, H. Mao, Y. Wang, and B. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," in *Proc. 6th Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, Apr. 2018. [Online]. Available: https://openreview.net/pdf?id=SkhQHMW0W

[37] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Proc. Adv. Neural Inf. Process. Syst.*, 1990, pp. 598–605.

[38] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," in *Proc. 4th Int. Conf. Learn. Represent.*, San Juan, Puerto Rico, May 2016. [Online]. Available: https://arxiv.org/abs/1510.00149

[39] L. Guerra, B. Zhuang, I. Reid, and T. Drummond, "Automatic pruning for quantized neural networks," 2020, *arXiv:2002.00523*.

[40] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 3123–3131.
[41] W. Wen *et al.*, "Terngrad: Ternary gradients to reduce communication in distributed deep learning," in *Proc. 31st Conf. Neural Inf. Process. Syst.*, Long Beach, CA, USA, Dec. 2017, pp. 1509–1519.
[42] F. Tung and G. Mori, "CLIP-Q: Deep network compression learning by in-parallel pruning-quantization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 2018, pp. 7873–7882.

Pavana Prakash (Graduate Student Member, IEEE) received the B.Eng. degree in instrumentation technology from the JSS Academy of Technical Education, VT University, Belgaum, India, in 2010. She is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX, USA.

She worked as a Technical Lead and a Software Engineer in the computer networking industry from 2010 to 2017. Her research interests include deep learning privacy, wireless edge networks compatible federated learning, and distributed optimization.

Ms. Prakash is a Student Member of ACM.

Jiahao Ding (Graduate Student Member, IEEE) received the B.S. degree in electronic information engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2017. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX, USA.

His research interests include private data analytics, trustworthy machine learning, and distributed optimization.

Mr. Ding is a Student Member of AAAI.

Rui Chen (Graduate Student Member, IEEE) received the B.S. degree from the Marine Electrical Engineering College, Dalian Maritime University, Dalian, China, in 2018. She is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX, USA.

Her major research interests include data-driven optimization, federated learning, differential privacy, resource management in wireless networks, and wireless for AI.

Xiaoqi Qin (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electrical and computer engineering from Virginia Tech, Blacksburg, VA, USA, in 2011, 2013, and 2016, respectively.

She is currently an Associate Professor with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing, China. Her research focuses on exploring performance limits of next-generation wireless networks and developing innovative solutions for intelligent and efficient machine-type communications.

Minglei Shu (Member, IEEE) received the B.S. degree in automation, the M.Sc. degree in power electronics and power transmission, and the Ph.D. degree in communication and information systems from Shandong University, Jinan, China, in 2003, 2006, and 2016, respectively.

He is currently working with the Shandong Artificial Intelligence Institute, Qilu University of Technology (Shandong Academy of Sciences), Jinan. His research interests include computer vision, medical image segmentation, IoT medical care, and wireless sensor networks.

Qimei Cui (Senior Member, IEEE) received the B.E. and M.S. degrees in electronic engineering from Hunan University, Changsha, China, in 2000 and 2003, respectively, and the Ph.D. degree in information and communications engineering from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2006.

She has been a Full Professor with the School of Information and Communication Engineering, BUPT, since 2014. She was a Visiting Professor with the Department of Electronic Engineering, University of Notre Dame, Notre Dame, IN, USA, in 2016. Her research interests include B5G/6G wireless communications, mobile computing, and IoT.

Prof. Cui won the Best Paper Award at IEEE ISCIT 2012, IEEE WCNC 2014, and WCSP 2019, the Honorable Mention Demo Award at ACM MobiCom 2009, and the Young Scientist Award at URSI GASS 2014. She serves as an Editor for *Science China Information Science* and a Guest Editor for the *EURASIP Journal on Wireless Communications and Networking*, *International Journal of Distributed Sensor Networks*, and *Journal of Computer Networks and Communication*. She serves as the Technical Program Chair for APCC 2018, the Track Chair for IEEE/CIC ICCC 2018, and the Workshop Chair for WPMC 2016. She also serves as a Technical Program Committee Member of several international conferences, such as the IEEE ICC, the IEEE WCNC, the IEEE PIMRC, the IEEE ICCC, the WCSP 2013, and the IEEE ISCIT 2012.

Yuanxiong Guo (Senior Member, IEEE) received the B.Eng. degree in electronics and information engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2009, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Florida, Gainesville, FL, USA, in 2012 and 2014, respectively.

Since 2019, he has been an Assistant Professor with the Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX, USA. His current research interests include machine learning, data-driven decision making, security and privacy with applications to Internet of Things, and edge computing.

Dr. Guo was a recipient of the Best Paper Award in the IEEE Global Communications Conference 2011. He is on the Editorial Board of IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY and servers as the Track Co-Chair for IEEE VTC 2021-Fall.

Miao Pan (Senior Member, IEEE) received the B.Sc. degree in electrical engineering from Dalian University of Technology, Dalian, China, in 2004, the M.A.Sc. degree in electrical and computer engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 2007, and the Ph.D. degree in electrical and computer engineering from the University of Florida, Gainesville, FL, USA, in 2012.

He is currently an Associate Professor with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX, USA. His research interests include wireless/AI for AI/wireless, deep learning privacy, cybersecurity, and underwater communications and networking.

Dr. Pan was a recipient of the NSF CAREER Award in 2014. His work won the IEEE Technical Committee on Green Communications and Computing Best Conference Paper Awards 2019 and the Best Paper Awards at ICC 2019, VTC 2018, Globecom 2017, and Globecom 2015, respectively. He is an Editor of IEEE OPEN JOURNAL OF VEHICULAR TECHNOLOGY, an Associate Editor of IEEE INTERNET OF THINGS (IOT) JOURNAL (Area 5: Artificial Intelligence for IoT), and used to be an Associate Editor of IEEE INTERNET OF THINGS (IOT) JOURNAL (Area 4: Services, Applications, and Other Topics for IoT) from 2015 to 2018. He has also been serving as a Technical Organizing Committee for several conferences, such as the TPC Co-Chair for Mobiquitous 2019 and ACM WUWNet 2019. He is a member of AAAI and ACM.