Model-Based Reinforcement Learning for Quantized Federated Learning Performance Optimization

Nuocheng Yang*, Sihua Wang*‖, Mingzhe Chen†, Christopher G. Brinton‡, Changchuan Yin*, Walid Saad§, and Shuguang Cui¶

*Beijing Laboratory of Advanced Information Network, ‖State Key Laboratory Of Networking And Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China, Emails: {YangNuoCheng,sihuawang,ccyin}@bupt.edu.cn.
†Department of Electrical and Computer Engineering and Institute for Data Science and Computing, University of Miami, Coral Gables, FL, 33146, USA, Email: mingzhe.chen@miami.edu.
‡School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA, Email: cgb@purdue.edu.
§Bradley Department of Electrical and Computer Engineering, Virginia Tech, Arlington, VA, 22203, USA, Email: walids@vt.edu.
¶Shenzhen Research Institute of Big Data (SRIBD) and the Future Network of Intelligence Institute (FNii), Chinese University of Hong Kong, Shenzhen, 518172, China, Email: cui@gmail.com.

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10001466

***Abstract*—This paper considers improving wireless communication and computation efficiency in federated learning (FL) via model quantization. In the proposed bitwidth FL scheme, edge devices train and transmit quantized versions of their local FL model parameters to a coordinating server, which, in turn, aggregates them into a quantized global model and synchronizes the devices. With the goal of jointly determining the set of participating devices in each training iteration and the bitwidths employed at the devices, we pose an optimization problem for minimizing the training loss of quantized FL under a device sampling budget and delay requirement. Our analytical results show that the improvement of FL training loss between two consecutive iterations depends on not only the device selection and quantization scheme, but also on several parameters inherent to the model being learned. As a result, we propose, a model-based reinforcement learning (RL) method to optimize action selection over iterations. Compared to model-free RL, the proposed approach leverages the derived mathematical characterization of the FL training process to discover an effective device selection and quantization scheme without imposing additional device communication overhead. Numerical evaluations show that the proposed FL framework can achieve the same classification performance while reducing the number of training iterations needed for convergence by 20% compared to model-free RL-based FL.**

***Keywords*—Bitwidth federated learning, FL training loss optimization, model-based reinforcement learning.**

I. Introduction

Federated learning (FL) is an emerging edge learning technique that enables a collection of devices to collaboratively train a shared machine learning model without sharing their individual datasets [1]. The local training and device-server communication processes in FL can each have a significant impact on performance. To minimize the delays incurred from these processes, recent methods have called for machine learning quantization at each device [1]. In such schemes, the training and communication processes operate directly on quantized versions of the learning models, reducing the burden on device resources. However, efficient deployment of quantized FL over wireless networks poses several research challenges related to the integration of quantization bitwidth considerations with the resulting FL training performance.

Recent works [2]–[6] studied several important problems related to the implementation of quantized FL over wireless networks. These prior works assumed that certain key parameters of the model being learned – such as smoothness and gradient diversity constants – are known in advance of the training process. Under these assumptions, traditional optimization methods can be used to capture the relationship between quantization error and FL performance so as to find the optimal FL training policy. In practice, however, these model parameters cannot be obtained by the central server until the FL training process has completed. To address this challenge, one promising approach is to employ reinforcement learning (RL) [1] to allow the server to estimate these parameters over time through interaction with the devices during the training process, allowing discovery of a more effective FL policy.

Recently, a number of works such as [7]–[9] have employed model-free RL algorithms to configure system parameters for FL performance optimization. However, in these schemes, the coordinating server must collect many observations of different FL training policies by interacting with the devices over the environment, resulting in considerable delay for finding the optimal policy and encumbering FL convergence speed.

The main contribution of this paper is a novel methodology for optimizing quantized FL algorithms over wireless networks which avoids continual interaction with devices through *model-based RL* for training parameter estimation. To our best knowledge, *this is the first work that provides a systematic analysis of the integration of quantization bitwidth optimization into FL.*

In particular, we first propose a novel quantized FL framework in which distributed wireless devices train and transmit their locally trained FL models to a coordinating server based on variable bitwidths. The server selects an appropriate set of devices to execute the FL algorithm with variable quantized bitwidths in each iteration.

Then, we formulate joint device selection and FL model

This work was supported in part by the National Natural Science Foundation of China under Grants 61871041, 61629101 and 61671086, in part by Beijing Natural Science Foundation-Haidian Original Innovation Foundation (L192003), in part by office of Naval Research grant N00014-21-1-2472 and National Science Foundation grant CNS-2146171, in part by the U.S. National Science Foundation under Grant CNS-2114267, in part by the Basic Research Project No. HZQB-KCZYZ-2021067 of Hetao Shenzhen-HK ST Cooperation Zone, in part by Guangdong Research Projects No. 2017ZT07X152 and No. 2019CX01X104, and in part by the Guangdong Provincial Key Laboratory of Future Networks of Intelligence (Grant No. 2022B1212010001).

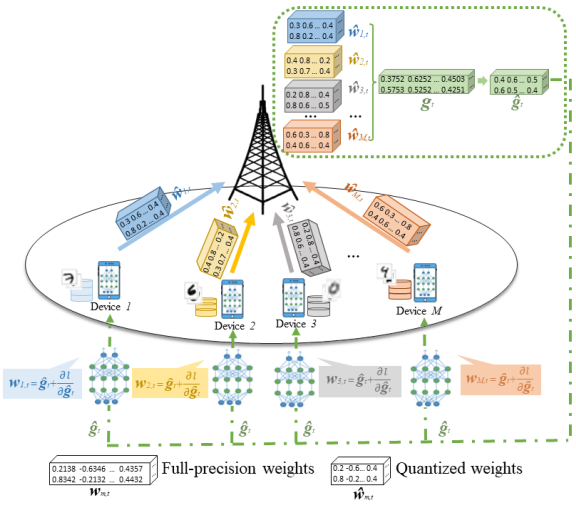

Fig. 1. Illustration of our proposed low bitwidth federated learning methodology deployed over multiple devices and one base station in a wireless network.

quantization as an optimization problem whose goal is to minimize training loss while accounting for communication and computation heterogeneity across the devices. To solve this problem, we analytically characterize the expected training convergence rate of our quantized FL framework. Given linear estimates of different ML loss properties, we show that the FL training process can be mathematically described as a Markov decision process (MDP). To learn the optimal solution of the formulated MDP, we construct a model-based RL method that infers the action (i.e., device selection and quantization scheme) that maximizes the expected reward (i.e., minimize global model loss) at each training iteration.

Our subsequent numerical evaluations for a real-world ML task show that our proposed methodology can improve training convergence speed by 20% compared to a model-free RL-based approach.

II. System Model and Problem Formulation

Consider a wireless network that consists of a set $\mathcal{M}$ of M devices connected upstream to a parameter server. These devices are aiming to collaboratively train a machine learning model, as shown in Fig. 1. Each device m has N_m training data samples, and each training data sample n consists of an input feature vector $\boldsymbol{x}_{m,n} \in \mathbb{R}^{N_I \times 1}$ and (in the case of supervised learning) a corresponding label vector $\boldsymbol{y}_{m,n} \in \mathbb{R}^{N_O \times 1}$. The objective of the system is to minimize the global loss function over all data samples, i.e.,

$$F\left(\boldsymbol{g}\right) = \min_{\boldsymbol{g}} \frac{1}{N} \sum_{m=1}^{M} \sum_{n=1}^{N_m} f\left(\boldsymbol{g}, \boldsymbol{x}_{m,n}, \boldsymbol{y}_{m,n}\right), \tag{1}$$

where $\boldsymbol{g} \in \mathbb{R}^{V \times 1}$ is a vector that captures the global FL model of dimension V trained across the devices, with $N = \sum_{m=1}^{M} N_m$ being the total number of training data samples across all devices. $f\left(\boldsymbol{g}, \boldsymbol{x}_{m,n}, \boldsymbol{y}_{m,n}\right)$ is a loss function (e.g., squared error) that measures the accuracy of the generated global FL model $\boldsymbol{g}$ in building a relationship between the input vector $\boldsymbol{x}_{m,n}$ and the output vector $\boldsymbol{y}_{m,n}$.

A. Training Process of Low Bitwidth Federated Learning

In FL, the devices and the server iteratively exchange their model parameters to find the optimal global model $\boldsymbol{g}$ that minimizes the global loss function in (1). However, due to limited computing and communication resources, devices may not be able to train and transmit large sized model parameters (e.g., as in the case of deep learning). To reduce the computation and transmission delays, bitwidth federated learning was proposed in [10]. Compared to the widely studied case of federated averaging [1], the FL model parameters in bitwidth FL are quantized. The overall training process of bitwidth FL is given as follows:

1) The server quantizes the initialized global learning model and broadcasts it to each device.
2) Each device calculates the training loss using the quantized global learning model and its collected data samples.
3) Based on the calculated training loss, the quantized learning model in each device is updated.
4) Each device quantizes its updated learning model.
5) The server selects a subset of devices for local FL model transmission.
6) The server aggregates the collected local FL models into a global FL model that will be transmit to devices.

Steps 2-6 are repeated until the optimal vector $\boldsymbol{g}$ is found.

In bitwidth FL, each device uses a quantized FL model to calculate the training loss and gradient vectors during the training process. Therefore, the quantization scheme in bitwidth FL will affect the resource requirements of FL model training and transmission. This is significantly different from quantization-based FL algorithms [2]–[4], [6] that must recover the quantized FL model during the training process, thus introducing additional computational complexity and reducing training efficiency.

Next, we formally explain this training process.

1) Calculation of Training Loss of Each Device: We first introduce the method for computing each device's training loss in step 2. The weights of each device's local FL model are quantized into α_t bits. Here, the full-precision neural network is transformed into a quantized neural network (QNN). When $\alpha_t = 1$, each QNN weight has two possible values, namely -1/0 or +1, which is referred to as a binary neural network (BNN) [11]. Given the input vector $\boldsymbol{h}_{m,t}^{k}$ and the weight vector $\hat{\boldsymbol{g}}_t^k$ of the neurons in layer k, the output of each layer k at iteration t is given by

$$\boldsymbol{h}_{m,t}^{k+1} = \begin{cases} \sigma\left(\boldsymbol{h}_{m,t}^{k} \odot \hat{\boldsymbol{g}}_t^k\right), & \text{if } \alpha_t = 1, \\ \sigma\left(\sum_{i=0}^{\alpha_t - 1} 2^i \left(\boldsymbol{h}_{m,t}^{k} \odot \hat{\boldsymbol{g}}_t^k\right)\right), & \text{if } \alpha_t > 1, \end{cases} \tag{2}$$

where $\sigma\left(\cdot\right)$ is the activation function and $\odot$ represents the inner product for vectors with bitwise operations. Given the outputs of all neuron layers $\boldsymbol{h}_{m,t} = [\boldsymbol{h}_{m,t}^{1}, \ldots, \boldsymbol{h}_{m,t}^{K}]$, the cross-entropy loss function can be expressed based on the neurons in an output layer $\boldsymbol{h}_{m,t}^{K}$ as

$$\begin{aligned} f\left(\hat{\boldsymbol{g}}_t, \boldsymbol{x}_{m,n}, \boldsymbol{y}_{m,n}\right) = &- \boldsymbol{y}_{m,n}^{T} \log\left(\boldsymbol{h}_{m,t}^{K}\right) + \\ &\left(1 - \boldsymbol{y}_{m,n}^{T}\right) \log\left(1 - \boldsymbol{h}_{m,t}^{K}\right), \end{aligned} \tag{3}$$

where $\hat{\boldsymbol{g}}_t = [\hat{\boldsymbol{g}}_t^1, ..., \hat{\boldsymbol{g}}_t^k, ..., \hat{\boldsymbol{g}}_t^K]$ is the quantized global FL model.

2) FL Model Update: A backward propagation (BP) algorithm based on stochastic gradient descent is used to update the parameters of the QNN. The update function is

$$\boldsymbol{w}_{m,t+1} = \hat{\boldsymbol{g}}_t - \lambda \sum_{n \in \mathcal{N}_{m,t}} \frac{\partial f\left(\boldsymbol{g}, \boldsymbol{x}_{m,n}, \boldsymbol{y}_{m,n}\right)}{\partial \boldsymbol{g}}, \tag{4}$$

where λ is the learning rate, $\mathcal{N}_{m,t}$ is the subset of training data samples (i.e., minibatch) selected from device m's training dataset $\mathcal{N}_m$ at iteration t, $\boldsymbol{w}_{m,t+1}$ is the updated local FL model of device m at iteration $t+1$, and

$$\frac{\partial f}{\partial \boldsymbol{g}} = \frac{\partial f_{m,t}}{\partial \hat{\boldsymbol{g}}_t} \times \frac{\partial \hat{\boldsymbol{g}}_t}{\partial \boldsymbol{g}_t} = \frac{\partial f_{m,t}}{\partial \hat{\boldsymbol{g}}_t} \times \mathrm{Htanh}\left(\boldsymbol{g}_t\right), \tag{5}$$

where $\boldsymbol{g}_t$ represents the full-precision weights. $\mathrm{Htanh}\left(x\right) = \max\left(-1, \min\left(1, x\right)\right)$ is used to approximate the derivative of the quantization function that is not differentiable. From (4) and (5), we can see that the weights are updated with full-precision values since the changes of the learning model update at each step are small.

3) FL Model Quantization at Device: As each local FL model is updated, the full-precision weights must be completely quantized into α_t bits. We adopt the quantization scheme described in [12]:

$$\hat{\boldsymbol{w}}_{m,t}^k\left(\alpha_t\right) = \begin{cases} \mathrm{sign}\left(\boldsymbol{w}_{m,t}^k\right), & \text{if } \alpha_t = 1, \\ \frac{R\left(\left(2^{\alpha_t}-1\right)\boldsymbol{w}_{m,t}^k\right)}{2^{\alpha_t}-1}, & \text{if } 1 < \alpha_t < 32, \\ \boldsymbol{w}_{m,t}^k, & \text{if } \alpha_t = 32, \end{cases} \tag{6}$$

where $\mathrm{sign}\left(x\right) = \begin{cases} +1, & \text{if } x \geqslant 0, \\ -1, & \text{otherwise,} \end{cases}$ and $R\left(\cdot\right)$ is the rounding function $R\left(x\right) = \begin{cases} \lfloor x \rfloor, & \text{if } x \leqslant \frac{\lfloor x \rfloor + \lceil x \rceil}{2}, \\ \lceil x \rceil, & \text{otherwise.} \end{cases}$

4) FL Model Transmission and Aggregation: Due to limited wireless bandwidth, the server may need to select a subset of devices in each iteration that will upload their local FL models for aggregation into the global model. Given the quantized local FL model $\hat{\boldsymbol{w}}_{m,t}$ of each device m at each iteration t, the update of the global FL model at iteration t is given by

$$\boldsymbol{g}_t\left(\boldsymbol{u}_t, \alpha_t\right) = \sum_{m=1}^{M} \frac{u_{m,t} N_{m,t}}{\sum_{m=1}^{M} u_{m,t} N_{m,t}} \hat{\boldsymbol{w}}_{m,t}\left(\alpha_t\right), \tag{7}$$

where $\frac{u_{m,t}N_{m,t}}{\sum_{m=1}^{M} u_{m,t}N_{m,t}}$ is a scaling update weight of $\hat{\boldsymbol{w}}_{m,t}$, with $N_{m,t}$ being the number of data samples used to train $\hat{\boldsymbol{w}}_{m,t}$ at device m. $\boldsymbol{g}_t\left(\boldsymbol{u}_t\right)$ is the global FL model at iteration t, and $\boldsymbol{u}_t = [u_{1,t}, \ldots, u_{M,t}]$ is the device selection vector, with $u_{m,t} = 1$ indicating that device m will upload its quantized local FL model $\hat{\boldsymbol{w}}_{m,t}$ to the server at iteration t, and $u_{m,t} = 0$ otherwise.

5) FL Model Quantization at the server: The server must in turn quantize each device's local FL model in low bitwidth that can be directly used to calculate the training loss at each device. This is given by

$$\hat{\boldsymbol{g}}_t^k = \begin{cases} \mathrm{sign}\left(\boldsymbol{g}_t^k\right), & \text{if } \alpha_t = 1, \\ \frac{R\left(\left(2^{\alpha_t}-1\right)\boldsymbol{g}_t^k\right)}{2^{\alpha_t}-1}, & \text{if } 1 < \alpha_t < 32, \\ \boldsymbol{g}_t^k, & \text{if } \alpha_t = 32. \end{cases} \tag{8}$$

B. Training Delay of Low Bitwidth Federated Learning

We next study the training delay of bitwidth FL. From the training steps, we can see that the delay consists of four components: (a) time used to calculate the training loss, (b) FL model update delay, (c) FL model quantization delay, and (d) FL model transmission delay. Since the local FL models are updated with full-precision values, the FL model update delay does not depend on the number of quantization bits α_t, and thus we do not consider it. The other components are:

1) Time Used to Calculate the Training Loss: The time needed to calculate the training loss depends on the number of multiplication operations in (2) and (3). From (2), we can see that the computational complexity of each multiplication operation is related to the number of bits α_t used to represent each element in FL model vector. In particular, given α_t, the time needed to calculate the training loss is given by [13]

$$l_{m,t}^{\mathrm{C}}\left(\alpha_t\right) = \rho \frac{\alpha_t^2 N^{\mathrm{C}}}{Bf}, \tag{9}$$

where ρ is the the time consumption coefficient depending on the chip of each device and N^{C} is the number of multiplication operations in the neural network. f and B represent the frequency of the central processing unit (CPU) and the number of bits that can be processed by the CPU in one clock cycle.

2) FL Model Quantization Delay: Since the updated local FL model is in full-precision, each device must quantize its updated local FL model using (6) to reduce transmission delay. Given α_t, the quantization delay can be represented as

$$l_{m,t}^{\mathrm{Q}}\left(\alpha_t\right) = \begin{cases} 0, & \text{if } \alpha_t = 1 \text{ or } \alpha_t = 32, \\ \frac{D}{Bf}, & \text{if } 1 < \alpha_t < 32. \end{cases} \tag{10}$$

where D is the number of neurons in the neural network. In (10), when $\alpha_t = 1$ or $\alpha_t = 32$, the quantization delay will be 0. When $\alpha_t = 1$, the value of quantized weight $\hat{\boldsymbol{w}}_{m,t}$ can be directly decided by the sign bit. When $\alpha_t = 32$, we do not need to quantize the weights since each full precision weight consists of 32 bits, i.e., $\hat{\boldsymbol{w}}_{m,t} = \boldsymbol{w}_{m,t}$.

3) FL Model Transmission Delay: To generate the global FL model that is aggregated by each quantized local FL model, each device must transmit $\hat{\boldsymbol{w}}_{m,t}$ to the server. To this end, we adopt an orthogonal frequency division multiple access (OFDMA) transmission scheme for transmitting the quantized local FL models. In particular, the server can allocate a set $\mathcal{U}$ of U uplink orthogonal resource blocks (RBs) to the devices for quantized weight transmission. Let W be the bandwidth of each RB and P be the transmit power of each device. The uplink channel capacity between device m and the server over each RB i is

$$c_{m,t}\left(u_{m,t}\right) = u_{m,t} W \log_2\left(1 + \frac{P h_{m,t}}{\sigma_N^2}\right), \tag{11}$$

where $u_{m,t} \in \{0,1\}$ is the user association index, $h_{m,t}$ is the channel gain between device m and the server, and σ_N^2 represents the variance of additive white Gaussian noise. Based on (11), the uplink transmission delay between device m and the server is $l_{m,t}^{\mathrm{T}}(u_{m,t},\alpha_t) = \frac{D\alpha_t}{c_{m,t}(u_{m,t})}$, where $D\alpha_t$ is the data size of the quantized FL parameters $\hat{\boldsymbol{w}}_{m,t}$.

Since the server has enough computational resources and sufficient transmit power, we do not consider the delay used for global FL model quantization and transmission. Thus, the time that the devices and the server require to jointly complete the update of their respective local and global FL models at iteration t is

$$l_t(\boldsymbol{u}_t,\alpha_t) = \max_{m\in\mathcal{M}} u_{m,t}\Big(l_{m,t}^{\mathrm{C}}(\alpha_t)+l_{m,t}^{\mathrm{Q}}(\alpha_t)+l_{m,t}^{\mathrm{T}}(u_{m,t},\alpha_t)\Big). \tag{12}$$

Here, $u_{m,t} = 0$ implies that device m will not send its quantized local FL model to the server, and thus not cause any delay.

C. Optimization Formulation

The goal of our optimization problem is to minimize the FL training loss while meeting a delay requirement on FL completion per iteration. This minimization problem involves jointly optimizing the device selection scheme and the quantization scheme, which is formulated as:

$$\min_{\boldsymbol{U},\boldsymbol{\alpha}} \quad F\left(\boldsymbol{g}\left(\boldsymbol{u}_T,\boldsymbol{\alpha}\right)\right), \tag{13}$$

$$\text{s.t. } u_{m,t}\in\{0,1\}, \alpha_t\in\{1,2,4,8,16,32\}, \forall m\in\mathcal{M}, \ \forall t\in\mathcal{T}, \tag{13a}$$

$$\sum_{m=1}^{M} u_{m,t} \leqslant U, \ \forall m\in\mathcal{M}, \forall t\in\mathcal{T}, \tag{13b}$$

$$l_t\left(\boldsymbol{u}_t,\boldsymbol{\alpha}\right) \leqslant \Gamma, \forall m\in\mathcal{M}, \forall t\in\mathcal{T}, \tag{13c}$$

where $\boldsymbol{U} = [\boldsymbol{u}_1,\ldots,\boldsymbol{u}_t,\ldots,\boldsymbol{u}_T]$ is a device selection matrix over all iterations with $\boldsymbol{u}_t = [u_{1,t},\ldots,u_{M,t}]$ being a user association vector at iteration t, $\boldsymbol{\alpha} = [\alpha_1,\ldots,\alpha_t,\ldots,\alpha_T]$ is a quantization precision vector of all devices for all iterations, and $\mathcal{T} = \{1,...,T\}$ is the training period. Γ is the (per iteration) delay constraint for completing FL training. (13a) implies that each device can quantize its local FL model and can only occupy at most one RB for FL model transmission. (13b) ensures that the server can only select at most U devices for FL model transmission per iteration. (13c) is a constraint on the FL training delay per iteration.

The problem in (13) is challenging to solve by conventional optimization algorithms due to the following reasons. First, the server needs the dataset information of each device to determine the optimal device selection and quantization scheme for minimizing the FL training loss. Second, as the stochastic gradient decent method is used to generate each local FL model, the relationship between the training loss and device selection as well as quantization scheme cannot by captured by the server via conventional optimization algorithms. This is because the stochastic gradient decent method enables each device to randomly select a subset of data samples in its local dataset for local FL model training, and hence, the server cannot directly optimize the training loss of each device. To address these challenges, we propose a model-based RL algorithm that enables the server to capture the relationship between the FL training loss and the chosen device selection and quantization scheme. Based on this relationship, the server can proactively determine $\boldsymbol{u}_t$ and α_t so as to minimize the FL training loss.

III. Optimization Methodology

In this section, we propose a model-based RL approach for optimizing the device selection scheme $\boldsymbol{U}$ and the quantization scheme $\boldsymbol{\alpha}$ in (13). Compared to traditional model-free RL approaches that require the server continually interact with edge devices to learn the device selection and quantization schemes, model-based RL approaches enable the server to mathematically model the FL training process, thus finding the optimal device selection and quantization scheme based on the learned state transition probability matrix.

We first introduce the components of the proposed model-based RL method. Here, a linear regression method is used to learn the dynamic environment model that is presented as a transition probability matrix in RL approach. Then, we detail the proposed model-based RL methodology method to find the optimal $\boldsymbol{U}$ and $\boldsymbol{\alpha}$.

A. Components of Model-Based RL Method

The proposed model-based RL method consists of six components: a) agent, b) action, c) states, d) state transition probability, e) reward, and f) policy, which are specified as follows:

- **Agent**: The agent that performs the proposed model-based RL algorithm is the server.
- **Action**: An action of the server is defined as $\boldsymbol{a}_t = [\boldsymbol{u}_t,\alpha_t] \in \mathcal{A}$, and it consists of the device selection scheme $\boldsymbol{u}_t$ and the quantization scheme α_t of all device at iteration t with $\mathcal{A}$ being the discrete sets of available actions.
- **States**: The state is defined as $s_t = F\left(\boldsymbol{g}_t\right) \in \mathcal{S}$, which measures the performance of global FL model at iteration t, with $F\left(\boldsymbol{g}_t\right)$ being the FL training loss and $\mathcal{S}$ being the sets of available states.
- **State Transition Probability**: The state transition probability $P\left(s_{t+1}|s_t,\boldsymbol{a}_t\right)$ is the probability of transitioning from state s_t to state s_t' when taking action $\boldsymbol{a}_t$, which is

$$P\left(s_{t+1}|s_t,\boldsymbol{a}_t\right) = \Pr\{s_{t+1}=s_t'|s_t,\boldsymbol{a}_t\}. \tag{14}$$

  Here, we note that in model-free RL algorithms, the server does not know the values of the state transition probability matrix. However, in our work, we analyze the convergence of FL convergence and estimate the FL training parameters in the FL convergence analytical results so as to calculate the state transition probabilities. Using the state transition probability matrix can reduce the interactions between the server and edge devices thus improving the convergence speed of RL.
- **Reward**: Based on the current state s_t and the selected action $\boldsymbol{a}_t$, the reward function of the server is

$$r\left(\boldsymbol{s}_t,\boldsymbol{a}_t\right) = -F\left(\boldsymbol{g}\left(\boldsymbol{u}_t,\alpha_t\right)\right), \tag{15}$$

where $F\left(\boldsymbol{g}\left(\boldsymbol{u}_t,\alpha_t\right)\right)$ is the training loss at iteration t.

- **Policy**: The policy is the probability of the agent choosing each action at a given state. We use a deep neural network parameterized by $\boldsymbol{\theta}$ to map the input state to the output action. Then, the policy is $\boldsymbol{\pi_\theta}\left(s_t,\boldsymbol{a}_t\right)=P\left(\boldsymbol{a}_t|s_t\right)$.

B. Calculation of State Transition Probability

We now detail the process of computing the state transition probability that is used to reduce the interactions between the server and edge devices thus improving the convergence speed of RL. To characterize how an action $\boldsymbol{a}_t=[\boldsymbol{u}_t,\alpha_t]$ affects the state transition in the considered bitwidth FL algorithm, we derive the following lemma:

Lemma 1. Given the user selection vector $\boldsymbol{u}_t$ and quantization scheme α_t, the upper bound of $\mathbb{E}\left(F\left(\boldsymbol{g}_{t+1}\right)\right)-\mathbb{E}\left(F\left(\boldsymbol{g}_t\right)\right)$ can be given by [14]

$$\begin{aligned}&\mathbb{E}\left(F\left(\boldsymbol{g}_{t+1}\right)\right)-\mathbb{E}\left(F\left(\boldsymbol{g}_t\right)\right)\\&\leq\frac{1}{2L}\left(-1+\frac{4\left(N-A\right)^2\mathbb{E}\left(\|\Delta\left(\alpha_t\right)\|+1\right)\zeta_2}{N^2}\right)\times||\nabla F\left(\boldsymbol{g}_t\right)||^2\\&\quad+\frac{\mathbb{E}\left\|\Delta\left(\alpha_t\right)\right\|+1}{2L}\left(\frac{4\left(N-A\right)^2\zeta_1}{N^2}+L^2\mathbb{E}\left\|\Delta\left(\alpha_t\right)\right\|\right)\\&\quad+\mathbb{E}\left(\Delta\left(\alpha_t\right)^2\right)=K\left(L,\zeta_1,\zeta_2|F(\boldsymbol{g}_t)^{(i)},\boldsymbol{a}_t^{(i)},F(\boldsymbol{g}_{t+1})^{(i)}\right),\end{aligned}\tag{16}$$

where $A=\sum_{m=1}^{M}u_{m,t}N_{m,t}$ is the sum of all selected devices' data samples that are used to train their local models. $1/L$, ζ_1, and ζ_2 are FL training parameters. $\Delta\left(\alpha_t\right)=\hat{\boldsymbol{g}}_t\left(\alpha_t\right)-\boldsymbol{g}_t$ is the quantization error of the global FL model that depends on the quantization scheme $\boldsymbol{\alpha}$ and $\mathbb{E}\left\|\Delta\left(\alpha_t\right)\right\|=M2^{-\alpha_t}$ is the unbiased quantization function defined in (6).

From Lemma 1, we see that the relationship between $\mathbb{E}\left(F\left(\boldsymbol{g}_{t+1}\right)\right)$ and $\mathbb{E}\left(F\left(\boldsymbol{g}_t\right)\right)$ (i.e., s_{t+1} and s_t) depends on the selected action $\boldsymbol{a}_t$ as well as the constants $1/L$, ζ_1, and ζ_2. However, we do not know the values of $1/L$, ζ_1, and ζ_2 since they are predefined in the approximation of the loss function. To find the tightest bound in (16), we must find the values of $1/L$, ζ_1, and ζ_2 so as to build the relationship between s_{t+1} and s_t and calculate the state transition probability $P\left(s_{t+1}|s_t,\boldsymbol{a}_t\right)$. To this end, a linear regression method [15] is used to determine the values of L, ζ_1, and ζ_2 since the relationship between $\mathbb{E}\left(F\left(\boldsymbol{g}_{t+1}\right)\right)-\mathbb{E}\left(F\left(\boldsymbol{g}_t\right)\right)$ and these constants are linear. The regression loss function defined as

$$\begin{aligned}\mathcal{J}\left(L,\zeta_1,\zeta_2\right)=\frac{1}{I}\sum_{i=1}^{I}\Bigg(&\left(\mathbb{E}\left(F\left(\boldsymbol{g}_{t+1}\right)^{(i)}\right)-\mathbb{E}\left(F\left(\boldsymbol{g}_t\right)^{(i)}\right)\right)\\&-K\left(L,\zeta_1,\zeta_2|F\left(\boldsymbol{g}_t\right)^{(i)},\boldsymbol{a}_t^{(i)},F\left(\boldsymbol{g}_{t+1}\right)^{(i)}\right)\Bigg)^2,\end{aligned}\tag{17}$$

where I is the number of real interactions between the server and the edge devices used to estimate $1/L$, ζ_1, and ζ_2. $\boldsymbol{b}^{(i)}=\left(F\left(\boldsymbol{g}_t\right)^{(i)},\boldsymbol{a}_t^{(i)},F\left(\boldsymbol{g}_{t+1}\right)^{(i)}\right)$ consists of the recorded FL training loss. The selected action observed by the server and devices will be used to estimate the values of $1/L$, ζ_1, and ζ_2. Given $\mathcal{B}=\left\{\boldsymbol{b}^{(0)},\ldots,\boldsymbol{b}^{(i)},\ldots,\boldsymbol{b}^{(I)}\right\}$, L, ζ_1, and ζ_2 are updated using a standard gradient descent method.

Algorithm 1 Model-based RL for device selection and quantization optimization

Input: The environment state $\mathcal{S}$, the action space $\mathcal{A}$.
Output: The device selection and quantization scheme.
1: Initialize policy $\boldsymbol{\pi_\theta}$, transition replay buffer $\mathcal{B}$, trajectory replay buffer $\boldsymbol{\tau}$.
2: **for** iteration $i=1:I$ **do**
3: Randomly selects a subset of devices to generate the global FL model that are quantized into α_t bits.
4: Records $F\left(\boldsymbol{g}_t\right)$, $F\left(\boldsymbol{g}_{t+1}\right)$, α_t, and device selection scheme $\boldsymbol{u}_t$ in $\mathcal{B}$.
5: **end for**
6: Estimate $1/L$, ζ_1, and ζ_2 to construct $P\left(s_{t+1}|s_t,\boldsymbol{a}_t\right)$ using (17) based on the real transition in $\mathcal{B}$.
7: **for** iteration $i=1:H$ **do**
8: Sample initial state from $\mathcal{S}$, then use policy $\boldsymbol{\pi_\theta}$ and learned $P\left(s_{t+1}|s_t,\boldsymbol{a}_t\right)$ to perform T trajectories and update $\boldsymbol{\tau}$.
9: Sample from $\boldsymbol{\tau}$, and update the current policy evaluation by solving Equation $\boldsymbol{\theta}=\boldsymbol{\theta}+\iota\nabla_{\boldsymbol{\theta}}\mathcal{L}\left(\boldsymbol{\theta}\right)$.
10: **end for**

Given the values of L, ζ_1, and ζ_2, the gap between $\mathbb{E}(F(\boldsymbol{g}_{t+1}))$ and $\mathbb{E}(F(\boldsymbol{g}_t))$ can be determined. Based on the definition of the state, the state transition probability $P(s_t+1|s_t,\boldsymbol{a}_t)$ is given by

$$P\left(s_{t+1}|s_t,a_t\right)=\begin{cases}1,\text{if } s_{t+1}=s_t+K\left(L,\zeta_1,\zeta_2|F(\boldsymbol{g}_t)^{(i)},\boldsymbol{a}_t^{(i)},F(\boldsymbol{g}_{t+1})^{(i)}\right)\\0,\text{ otherwise.}\end{cases}\tag{18}$$

C. Optimization of Device Selection and Quantization Scheme

With the state transition probability $P\left(s_{t+1}|s_t,\boldsymbol{a}_t\right)$ in hand, we proceed to optimize $\boldsymbol{\pi_\theta}$ so as to find the optimal device selection scheme $\boldsymbol{u}_t$ and quantization scheme α_t. Optimizing $\boldsymbol{\pi_\theta}$ for minimizing the FL training loss is done based on the following likelihood [16]:

$$\begin{aligned}\mathcal{L}\left(\boldsymbol{\theta}\right)=&\sum_{\left(s_t,\boldsymbol{a}_t\right)\in\boldsymbol{\tau}}P\left(s_0\right)\prod_{t=1}^{T}\boldsymbol{\pi}_\theta\left(s_{t-1},\boldsymbol{a}_t\right)\\&\times P\left(s_t|s_{t-1},\boldsymbol{a}_t\right)\sum_{t=1}^{T}r\left(\boldsymbol{s}_t,\boldsymbol{a}_t\right),\end{aligned}\tag{19}$$

where $\boldsymbol{\tau}=\{s_0,\boldsymbol{a}_0,\ldots,s_T,\boldsymbol{a}_T\}$ is the trajectory replay buffer.

Given (19), the optimization of policy network $\boldsymbol{\theta}$ is $\max_{\boldsymbol{\theta}}\mathcal{L}\left(\boldsymbol{\theta}\right)$. We update $\boldsymbol{\pi_\theta}$ using a standard gradient descent method. The entire process of training the proposed model-based RL algorithm is shown in Algorithm 1.

IV. Simulation Results

For our simulations, we consider a circular network area having a radius $r=1500$ m with one server at its center serving $M=12$ uniformly distributed devices. The other parameters used in simulations are listed in Table I. We consider a handwritten digit identification ML task based on the MNIST dataset [17]. We compare the proposed algorithm with an FL algorithm that is optimized by model-free RL.

TABLE I
SIMULATION PARAMETERS [18]

Parameters	Values	Parameters	Values
W	15 kHz	P	0.5 W
σ_N^2	-174 dB	ρ	2.8×10^6
T	1000	f	3.3 GHz

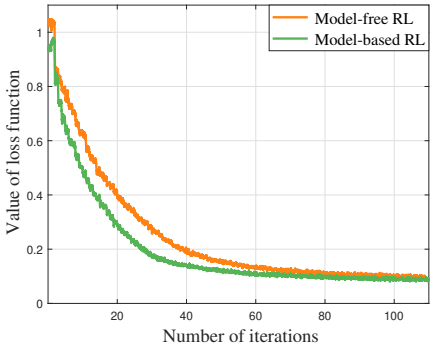

Fig. 2. Training loss vs. the number of iterations.

In Fig. 2, we show how the FL training loss changes as the number of iterations varies. We see that the proposed model-based RL method can improve convergence speed by 21% compared to a model-free RL-based approach. This is due to the fact that the proposed method enables the server to estimate the FL training parameters in the first few iterations so as to model the FL training process mathematically, speeding up the convergence.

In Fig. 3, we show how the number of iterations required to converge in FL varies as the number of devices changes. This shows that the proposed model-based RL method can reduce the iterations required to converge by 20% compared to the proposed model-based RL when the network has 6 RBs and 15 devices. The 20% gain stems from the fact that the model-based RL method enables the server to optimize the FL training process through minimal interaction with each device, resulting in quickly finding the optimal policy and not encumbering FL convergence speed.

V. CONCLUSION

In this paper, we developed a novel quantized FL framework in which distributed wireless devices train and transmit their locally trained FL models to a coordinating server based on variable bitwidths. We formulated an optimization problem that jointly considers the device selection and quantization scheme to minimize FL training loss while accounting for communication and computation heterogeneity across the devices. To solve this problem, we analytically derived the expected training convergence rate of our quantized FL framework. We then showed how the expected improvement of FL training loss between two adjacent iterations depends on the device selection scheme, the quantization scheme, and inherent properties of the model being trained. Based on estimators for model property evoluation, the improvement of FL performance at adjacent iterations was modeled as an MDP. We then proposed a model-based RL method to learn the relationship between FL performance and the choice of device selection and quantization scheme so as to converge on the

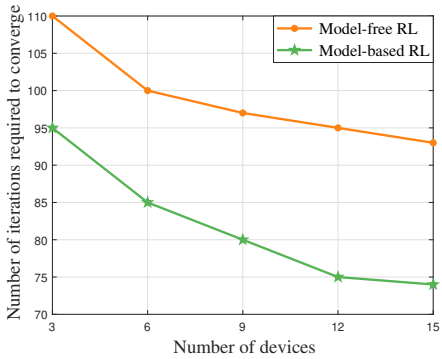

Fig. 3. Number of iterations required to converge vs. the number of devices.

policy minimizing FL loss. Numerical evaluation demonstrated that the proposed method can significantly improve the FL convergence speed compared to model-free RL-based FL.

REFERENCES

[1] M. Chen, D. Gündüz, K. Huang, W. Saad, M. Bennis, A. V. Feljan, and H. V. Poor, "Distributed learning in wireless networks: Recent progress and future challenges," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3579–3605, Dec. 2021.
[2] S. Chen, C. Shen, L. Zhang, and Y. Tang, "Dynamic aggregation for heterogeneous quantization in federated learning," *IEEE Transactions on Wireless Communications*, vol. 20, no. 10, pp. 6804–6819, May 2021.
[3] Y. Wang, Y. Xu, Q. Shi, and T. H. Chang, "Quantized federated learning under transmission delay and outage constraints," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 1, pp. 323–341, Jan. 2022.
[4] Y. Du, S. Yang, and K. Huang, "High-dimensional stochastic gradient quantization for communication-efficient edge learning," *IEEE Transactions on Signal Processing*, vol. 68, pp. 2128–2142, March 2020.
[5] M. Chen, N. Shlezinger, H. V. Poor, Y. C. Eldar, and S. Cui, "Communication-efficient federated learning," *Proceedings of the National Academy of Sciences*, vol. 118, no. 17, 2021.
[6] J. Sun, T. Chen, G. B. Giannakis, Q. Yang, and Z. Yang, "Lazily aggregated quantized gradient innovation for communication-efficient federated learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 4, pp. 2031–2044, Oct. 2022.
[7] W. Zhang, D. Yang, W. Wu, H. Peng, N. Zhang, H. Zhang, and X. Shen, "Optimizing federated learning in distributed industrial IoT: A multi-agent approach," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3688–3703, Oct. 2021.
[8] H. T. Nguyen, N. Cong Luong, J. Zhao, C. Yuen, and D. Niyato, "Resource allocation in mobility-aware federated learning networks: A deep reinforcement learning approach," in *Proc. of IEEE World Forum on Internet of Things (WF-IoT)*, LA, USA, June 2020.
[9] Y. Zhan, P. Li, and S. Guo, "Experience-driven computational resource allocation of federated learning by deep reinforcement learning," in *Proc. of IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, LA, USA, May 2020.
[10] L. U. Khan, W. Saad, Z. Han, E. Hossain, and C. S. Hong, "Federated learning for Internet of Things: Recent advances, taxonomy, and open challenges," *IEEE Communications Surveys & Tutorials*, vol. 3, no. 23, pp. 1759–1799, June 2021.
[11] H. Qin, R. Gong, X. Liu, X. Bai, J. Song, and N. Sebe, "Binary neural networks: A survey," *Pattern Recognition*, vol. 105, pp. 1–14, July 2020.
[12] Y. Cai, T. Tang, L. Xia, M. Cheng, Z. Zhu, Y. Wang, and H. Yang, "Training low bitwidth convolutional neural network on RRAM," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jeju, Korea (South), Jan. 2018.
[13] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 3, pp. 1935–1949, March 2021.
[14] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 269–283, Oct. 2021.
[15] N. Megiddo and A. Tamir, "Finding least-distances lines," *Journal on Algebraic and Discrete Methods*, vol. 4, no. 2, pp. 207–211, June 1983.
[16] Y. Wang, M. Chen, Z. Yang, W. Saad, T. Luo, S. Cui, and H. V. Poor, "Meta-reinforcement learning for reliable communication in THz/VLC wireless VR networks," *IEEE Transactions on Wireless Communications*, to appear, 2022.
[17] Y. LeCun, "The MNIST database of handwritten digits," Available Online*: http://yann.lecun.com/exdb/mnist/*, Sep. 2020.
[18] S. Wang, M. Chen, X. Liu, C. Yin, S. Cui, and H. V. Poor, "A machine learning approach for task and resource allocation in mobile-edge computing-based networks," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1358–1372, Feb. 2021.