Content Popularity Prediction Based on Quantized Federated Bayesian Learning in Fog Radio Access Networks

Yunwei Tao®, Yanxiang Jiang®, Senior Member, IEEE, Fu-Chun Zheng®, Senior Member, IEEE, Zhiheng Wang, Student Member, IEEE, Pengcheng Zhu®, Member, IEEE, Meixia Tao®, Fellow, IEEE, Dusit Niyato®, Fellow, IEEE, and Xiaohu You®, Fellow, IEEE

Abstract—In this paper, we investigate the content popularity prediction problem in cache-enabled fog radio access networks (F-RANs). In order to predict the content popularity with high accuracy and low complexity, we propose a Gaussian process based regressor to model the content request pattern. Firstly, the relationship between content features and popularity is captured by our proposed model. Then, we utilize Bayesian learning to train the model parameters, which is robust to overfitting. However, Bayesian methods are usually unable to find a closed-form expression of the posterior distribution. To tackle this issue, we apply a stochastic variance reduced gradient Hamiltonian Monte Carlo (SVRG-HMC) method to approximate the posterior distribution. To utilize the computing resource of fog access points (F-APs) and also reduce the communication overhead, we propose a quantized federated learning (FL) framework combining with Bayesian learning. The proposed quantized federated Bayesian learning framework allows each F-AP to send gradients to the cloud server after quantizing and encoding. It can achieve a tradeoff between prediction accuracy and communication overhead effectively. Simulation results show that the performance of our proposed policy outperforms the considered baseline policies.

Index Terms—F-RANs, Bayesian learning, federated learning, content popularity, content feature.

I. INTRODUCTION

T he 6th-Generation mobile communications system aims to provide higher quality services and larger coverage areas [1]. It is also expected to stimulate new developments in various industry applications, such as healthcare, energy, agriculture, etc. Rapid growth increases the pressure on traffic, and fog radio access network (F-RAN) equipped with cache storage is considered as a promising solution [2]. In F-RANs, fog access points (F-APs) with limited caching and computing resource are deployed at the edge of the network, where popular contents can be cached to satisfy most of the users’ requests [3]. In view of the caching capacity constraints, the issue of predicting content popularity has attracted more and more attention, since it plays an important role in improving caching efficiency.

Traditional caching update policies, such as first in first out (FIFO), least recently used (LRU) and least frequently out (FIFO), least recently used (LRU) and least frequently used (LFU), are widely used in wired networks with sufficient storage and computing resource [4], [5], [6]. However, they are inefficient because they are all reactive and have ignored content popularity. When considering content popularity, most existing proactive edge caching policies assume that content popularity is already known [7], [8], which is nonrealistic, especially when the contents are specific to the domain, e.g., IoT applications. It is therefore important to predict future content popularity based on the information collected by the communications network. In [9], a novel deep learning-based proactive caching framework, i.e., DeepCachNet, was proposed to estimate content popularity at the core network by extracting the features of users and contents. In [10], the authors proposed a user preference model to predict content popularity with model parameters learned by online gradient descent (OGD), which was proved to be superior to traditional caching schemes in [4], [5], and [6]. In [11], the authors proposed two online prediction models for content popularity, namely the popularity prediction model and the Grassman prediction model, where the unconstrained coefficients for linear prediction were obtained by solving a constrained nonnegative least squared method. In [12], the authors investigated the effect of time on the prediction of popularity by dynamically weighting the feature sequences, mainly using dynamic weighting techniques to model the changes in the importance of factors over time.

Digital Object Identifier 10.1109/TCOMM.2022.3229679
of the feature sequences over time. A simplified bi-directional long short-term memory (Bi-LSTM) network based content popularity prediction scheme was proposed in [13] and [14], which tracked popularity trend by the number of requests. In [15] and [16], federated learning (FL) was utilized to obtain the context-aware popularity prediction model. In [17], popular contents were learned through a Possion regressor model whereby the Monte Carlo method was utilized to optimize the model parameters. In [18], to maximize the long-term Quality of Experience (QoE) of users, a Mobile Edge Computing (MEC) assisted predictive adaptive video streaming scheme was proposed for on-road driving scenarios based on deep reinforcement learning (DRL). In [19], a neural network was deployed at each user to predict file popularity, and FL was utilized to address the privacy issue. In [20], a stacked autoencoder-based scheme was proposed to predict the content popularity by considering both the local and global user request status within a specified time interval, and a DRL-based algorithm was then applied to solve the edge content placement problem based on the predicted popularity. In [21], the delay-sensitive augmented reality (AR) application was deployed in a fog computing framework where unmanned aerial vehicles (UAVs) served as cache-enabled edge nodes, and the performance of transmission delay and energy consumption during AR data acquisition and AR content downloading was then optimized by solving two nonconvex programming problems. These existing works except [9], [15], [17] fail to predict the popularity of newly-added contents with few statistical data. In other words, the content popularity cannot be predicted with a small number of training samples [22]. For example, the performance was limited by lack of statistical data [23], which caused serious overfitting problems. Furthermore, the stacked autoencoder-based scheme proposed in [20] also requires a great number of training samples and statistical data to refactor the input data set by extracting the foremost characteristic. In F-RANs, it is indeed difficult to obtain adequate data about numerous contents through F-APs with limited storage. Moreover, most of the prediction policies employ centralized learning approach that cannot efficiently utilize the computing resource of edge devices.

Most current research works focus on prediction over a single F-AP or high power node (HPN). Since the coverage area and statistical data of a single F-AP are limited, multiple F-APs need to collaborate in prediction [15]. However, collaborative prediction models suffer from a critical issue of data transfer. In [24] and [25], the authors claimed that data transfer from data owners to the centralized server would lead to privacy leakage, and it was also impractical to transfer all data to the centralized server due to limited network bandwidth. Correspondingly, FL, a novel distributed learning approach that does not need to exchange raw user data, was proposed to solve these problems [15]. FL enables multiple participating devices to jointly train a machine learning model. Each participant computes the local model based on the local training data and then sends the local training model to the server for aggregation, which certainly prevents the privacy leakage of the participants. Nowadays, FL has become an effective framework to perform distributed optimization.

Despite the benefits brought by FL, communication is still the challenge to confront with. In order to deal with the problem of heavy communication overhead in FL training process, many researches about communication in FL have been conducted. In [26], the authors proposed an adaptive federated averaging method using distributed Adam optimization and compression techniques. In [27], a sparse ternary compression method was proposed to meet the requirements of FL, which extended the existing compression of Top-K gradient sparsification with optimal Golomb encoding. In [28], a quantization method was proposed to reduce the volume of the model parameters exchanged among edge devices, and an efficient wireless resource allocation scheme was developed. In [29], the authors used periodic averaging, partial device participation and quantized message passing to achieve a trade-off between communication and computation. In conclusion, communication efficiency is an important challenging issue that has to be considered in FL.

Motivated by the aforementioned discussions, we propose a popularity prediction policy via content request probabilistic model based on content features. The model parameters are learned by Bayesian learning, which is a robust method against overfitting caused by lack of statistical data [30]. The training inputs of the prediction model are the features of contents requested by users and the number of requests in different time periods. Due to the computing resource constraint in F-APs, a modified stochastic gradient based Bayesian learning method is adopted to learn the popularity prediction model. Considering the collaboration of multiple F-APs, we combine the advantages of Bayesian learning and FL to establish a distributed learning framework. Besides, we propose a quantization and encoding scheme to improve the communication efficiency in FL. Our main contributions are summarized as follows.

1) We build a content request model and propose a content popularity prediction policy based on content features and the number of content requests. The proposed policy does not require user preferences and therefore needs less access to user privacy. Then, the content request model is trained by Bayesian learning to obtain the predicted popularity. Specifically, we innovatively introduce stochastic variance reduced gradient (SVRG) into the traditional Bayesian learning method, which can greatly reduce the computational complexity and training time. Finally, our proposed popularity prediction policy can predict future content popularity of the existing contents as well as newly-added contents.

2) We construct a global content popularity prediction model based on an FL framework. The method integrates Bayesian learning into the FL framework. In this way, the local models can be generated by Bayesian learning and finally aggregated effectively to obtain the global model, which can greatly reduce the computational burden and communication overhead of the cloud server. Besides, our
proposed model is robust to overfitting due to the local Bayesian learning procedure.

3) We propose a quantization and encoding scheme for federated Bayesian learning to further reduce communication overhead. The gradients uploaded by participants are quantized. After that, a variable-length encoding method is used to encode the quantization results for transmission with improved communication efficiency. We analyze the performance of the quantization and encoding scheme and derive the upper bound on the number of bits required for transmission after encoding. The theoretical analysis shows that our proposed scheme can greatly reduce the communication overhead and achieve an efficient tradeoff between prediction accuracy and communication overhead.

4) We validate our theoretical results with real-world dataset. The simulation results show that our proposed quantized federated Bayesian learning based content popularity prediction policy can predict future content popularity with high accuracy and timely refresh the cache storage based on the prediction results. Compared with the considered baseline policies, a higher cache hit rate and lower communication overhead can be achieved.

The reminder of this paper is organized as follows. The system model is presented in Section II. Section III describes the proposed popularity prediction policy based on quantized federated Bayesian learning. Simulation results are shown in Section IV and final conclusions are drawn in Section V.

II. SYSTEM MODEL

As shown in Fig. 1, we consider an F-RAN architecture in a specific region with $M$ F-APs, where the users are served by these F-APs and the cloud server [31]. Let $Q = \{q_1, q_2, \ldots, q_n, \ldots, q_M\}$ denote the set of F-APs and $C = \{c_1, c_2, \ldots, c_f, \ldots, c_F\}$ the content library located in the cloud server [32]. Let $x_f$ denote the feature vector of content $c_f$ and $s_f$ the size of content $c_f$. Take the movie category as an example, and a file may include genre (e.g., action, comedy, and sci-fi) and some other features such as release year and production company. If content $c_{f}$ requested by a user is stored in its associated F-AP or the neighboring F-APs in the region, then a cache hit occurs, and the content can be downloaded directly from that F-AP or the neighboring F-APs. Otherwise, the content needs to be fetched from the cloud server [33]. Without loss of generality, we assume that all the F-APs have the same storage capacity and communication ability [34].

An F-AP can collect information about the considered user’s requested contents, which mainly includes the features of the requested contents and the number of requests. Let $r_c[n] = [r_{c_1}[n], r_{c_2}[n], \ldots, r_{c_F}[n]]^T$ denote the vector representing the number of requests from all the users in the considered region for each content in the content library during the $n$-th time period. Considering that the number of requests shows the level of preference for certain movie, we take the number of requests as the popularity indicator for the considered contents. We consider global content popularity in the given coverage region and hence assume the content popularity distributions are the same among all the F-APs in the region [15]. Assume that $N$ time periods have been observed. Let $r_{cf}[N]$ and $r_{cf}[N+1]$ denote the predicted and real numbers of requests of $c_f$ during the $(N+1)$-th time period, respectively. There exists deviation between the predicted and real numbers of requests. Correspondingly, the root mean squared error (RMSE) is utilized to measure the prediction accuracy as follows:

$$\text{RMSE} = \sqrt{\frac{1}{F} \sum_{f=1}^{F} \left| r_{cf}[N] - r_{cf}[N+1] \right|^2}. \quad (1)$$

The cache hit rate is defined as the ratio of the number of cache hits to the number of total requests. The higher the number of cache hits is, the less contents are fetched from the cloud server, which results in less communication pressure over the backhaul links. By caching contents with higher popularity, the cache hit rate can be significantly increased.

The goal of this paper is to find a content popularity prediction policy that can minimize the prediction RMSE with lower computational complexity and less communication overhead. For convenience, a summary of major notations is presented in Table I.

III. PROPOSED CONTENT POPULARITY PREDICTION POLICY BASED ON QUANTIZED FEDERATED BAYESIAN LEARNING

In order to minimize the prediction RMSE, we propose a quantized federated Bayesian learning based content popularity prediction policy in this section. The policy is composed of a probabilistic model construction phase, a local model learning phase, a global model integration phase, and a predicting phase. The proposed policy utilizes the number of requests and content features to predict content popularity accurately.

A. Policy Outline

1) Probabilistic Model Construction Phase: The relationship between content popularity and content features is hard to generalize. Given the nonlinear relationship between content features and content popularity, traditional regression models are not able to capture it. Gaussian process, on the other hand, can model nonlinear relationship flexibly and effectively. Therefore, the construction of a probabilistic regression model based on Gaussian process is the first step in our proposed policy. If the feature vectors of two contents are similar, the popularity of the two contents will be close to each other.
The request arrivals are modeled using the typical Poisson distribution, which is the most commonly used intuitive model.

2) Local Model Learning Phase: In the considered edge caching scenario in F-RANs, F-APs may obtain only a few request observations, so overfitting is a challenging problem [30]. We propose to learn the probabilistic model via Bayesian learning due to its robustness to overfitting [22]. However, Bayesian learning cannot yield a closed-form expression in most cases. Therefore, Markov Chain Monte Carlo (MCMC) is utilized to approximate the model parameters. Concretely, we apply an SVRG based variant of MCMC to content popularity prediction for the first time. First, we obtain the unnormalized posterior approximation of the parameters based on their prior knowledge. Second, we compute the gradient of the posterior distribution on the basis of the existing request observations accordingly. Finally, we apply a discretization method to approximate the updated parameters via multiple sampling.

3) Global Model Integration Phase: The coverage area and training data of a single F-AP are limited. Consequently, we propose a training approach that can combine FL and Bayesian learning. First, each F-AP still takes the Bayesian approach which is proposed in the second step in the local model learning phase. Second, different weights are assigned to F-APs according to the local dataset which participate in FL to learn the global model. Finally, a quantization and encoding scheme is proposed for federated Bayesian learning to trade off communication overhead and prediction accuracy.

4) Predicting Phase: After the local model learning and global model integration phases, the approximate model parameters can be estimated. Accordingly, each F-AP is capable of predicting the popularity of existing contents or newly-added contents in the next time period with lower computational complexity and less communication overhead.

B. Probabilistic Model Construction Phase

1) Review of Gaussian Process: Gaussian process is a very powerful nonparametric Bayesian tool for modeling nonlinear problems. A Gaussian process is a set of random variables of a certain class, and any number of random variables in the set has a joint Gaussian distribution [35]. Using a Gaussian process, we can define the distribution with a non-parametric function

\[
\mu(x) = \mathbb{E}[f(x)], \quad K(x, x') = \mathbb{E}[(f(x) - \mu(x)) (f(x') - \mu(x'))]. \tag{2}
\]

This means that any \(M\) samples follow a joint Gaussian distribution

\[
[f(x_1), f(x_2), \ldots, f(x_M)]^T \sim \mathcal{N}(\mu, K),
\]

where \(\mu = [\mu(x_1), \mu(x_2), \ldots, \mu(x_M)]^T\) denotes the mean vector and \(K\) represents the covariance matrix \(K_{ij} = K(x_i, x_j)\). The kernel function \(K(\cdot)\) specifies the features of the function that we wish to model. It depends on intuition and experience to choose a proper kernel function for a learning task, and the squared exponential kernel functions, the Marton kernel functions and the radial basis kernel functions are the most commonly used.
Gaussian process is often used in prediction tasks. By giving a set of training datasets $\mathcal{D} \equiv \{X, Y\}$, where $X = [x_1, x_2, \ldots, x_n]$ is the training input and $Y = [y_1, y_2, \ldots, y_n]^T$ is the training output, the aim is to predict the output $Y^* = [y_1^*, y_2^*, \ldots, y_n^*]^T$ given a test input dataset $X^* = [x_1^*, x_2^*, \ldots, x_n^*]$ and the posterior distribution $p(Y^* | \mathcal{D}, X^*; \theta)$, where $\theta$ is the hyper-parameter vector in the kernel function. According to the definition of Gaussian process, the training output $Y$ and the test output $Y^*$ follow the joint distribution:

$$
\begin{bmatrix}
Y \\
Y^*
\end{bmatrix} \sim \mathcal{N}
\begin{pmatrix}
0 \\
[K + \sigma_e^2 I_n] k^*
\end{pmatrix},
$$

where $\sigma_e$ is the hyper-parameter of the Gaussian process, $I_n$ represents the $n$-dimensional identity matrix, $K = K(X, X)$ represents the $n \times n$ correlation matrix between the training inputs, $k^* = K(X, X^*)$ represents the $n \times n^*$ correlation matrix between the training and test inputs, and $k^{**}$ represents the $n^* \times n^*$ correlation matrix between the test inputs. By applying the properties of Gaussian distribution, the mean and variance of the posterior distribution can be obtained as follows:

$$
\begin{align*}
\mathbb{E}[Y^*] &= (k^*)^T (K + \sigma_e^2 I_n)^{-1} Y, \\
\text{Var}[Y^*] &= k^{**} - (k^*)^T (K + \sigma_e^2 I_n)^{-1} k^*.
\end{align*}
$$

Accordingly, the statistical properties of the predicted output can be obtained.

2) Probabilistic Model: In this subsection, we propose a probabilistic model to show the users’ request pattern. In order to introduce the proposed model, first, we assume that each content has a set of characteristics. For example, a video may belong to a specific category (e.g., education, arts, entertainment, and technology), and have some other features such as year of release, language, etc. Let $x_f$ denote a $Q$-dimensional feature vector of content $c_f$ whose values can be binary or continuous, which allows the proposed probabilistic model to handle diverse combinations of features. Then, the proposed regression-based hierarchical multilevel probabilistic model can be expressed as follows:

$$
r_{c_f}[n] | \lambda_f (x_f) \sim \text{Poisson} \left( \frac{e^{\lambda_f (x_f)}}{\beta_0} \right), \quad \forall n = 1, 2, \ldots, N,
$$

$$
\lambda_f (x_f) | g(x_f), \beta_0 \sim \mathcal{N} (g(x_f), \beta_0),
$$

$$
g(x_f) \mid x_i, \beta_1, \beta_2, \ldots, \beta_{Q+1} \sim \mathcal{GP} (0, K(x, x')),
$$

$$
\beta_q \mid \text{Gamma} \left( A_q, B_q \right), \quad \forall q = 0, 1, \ldots, Q + 1.
$$

The first layer of the model is the distribution of content request observations. In this layer, the number of requests for content $c_f$ is assumed to obey a Poisson distribution with natural parameter $\lambda_f (x_f)$ and the request rate $e^{\lambda_f (x_f)}$ is an exponential function of the natural parameter. Poisson distribution is a simple and widely used distribution to describe the occurrence number of a random event during unit time (or space) [36], [37]. It is utilized to describe the occurrence number of random content requests, where the parameter $\lambda_f$ is the average occurrence rate of the request for content $c_f$ per unit time. Moreover, $\lambda_f$ is modeled as a function of content features, indicating that the request probability of the considered content is directly related to its features.

As mentioned in the previous discussions, the model recognizes that those contents with similar features tend to have similar request patterns or popularity, and this prior information needs to be applied at a higher level of the model. In (5b), $\lambda_f (x_f)$ follows a normal distribution with mean $g(x_f)$ and variance $\beta_0$ [38]. With this assumption, the model defines popularity as a function of content features and also allows contents with the same features to have different popularity through the variance $\beta_0$, which is consistent with prior knowledge.

In the third layer (5c) of the model, the Gaussian process is applied and $g(x_f)$ is assumed to be the realization of a Gaussian process with zero mean and a covariance matrix with kernel function $K(x_i, x_j)$, which captures the relationship between the different features and reflects them on the subsequent samples. The covariance matrix $[K]_{i,j} = K(x_i, x_j)$ is the kernel function that determines the interpretation of Gaussian process and the correlation or similarity between the random variables. Our proposed model utilizes the scaled exponential kernel (SEK), which is defined as $K(x_i, x_j) = \beta_1 e^{-\sum_{q=0}^{Q} \beta_q \| x_i^{q+1} - x_j^{q+1} \|_2^2}$, where $\beta_1$ determines the vertical scale and $\beta_q$ determines the horizontal scale on the $q$-th dimensional feature. By using a different scale for each input feature dimension, the proposed model can change their importance. For example, if $\beta_q$ tends to 0, the features in the $q$-th dimension have little effect on the covariance. Besides, the kernel function $K(x_i, x_j)$ is infinitely differentiable due to the nature of the exponential function.

Finally, the model introduces uncertainty in the values of the kernel function parameters $\{\beta_q\}_{q=1}^{Q}$ and the variance of the Gaussian distribution $\beta_0$, since the existing prior knowledge cannot completely determine their values. The gamma distribution $\text{Gamma} \left( A_q, B_q \right)$ is used as the prior for each of $\{\beta_q\}_{q=0}^{Q}$, where $A_q$ is called the shape parameter and $B_q$ is the inverse scale parameter.

The schematic of our proposed probabilistic model is shown in Fig. 2, which indicates the mapping relationship among the parameters at each level of the probabilistic model. In our proposed probabilistic model, the number of requests for each content is assumed to obey a Poisson distribution in (5a), and its request rate in (5b) is determined by a realization of Gaussian process in (5c). The realization is related to content features and kernel function parameters of the Gaussian process in (5d). Finally, the model uncertainty is measured by the Gamma distribution with $A_q$ and $B_q$.

C. Local Model Learning Phase

1) Bayesian Learning in a Nutshell: Bayesian learning is an important branch in machine learning, which is based
on Bayesian theory. Bayesian theory states that the model parameters of Bayesian learning follow a potential distribution, and the optimal decisions of parameters can be made based on this distribution and the observed data. The most important advantage is that it integrates the prior knowledge of model parameters, which can make the model more robust to overfitting. Given a set of training examples \( D \), we need to infer the model \( h \) that generates these data. The model \( h \) is considered to be jointly determined by the objective function and the parameters \( \omega \) of the distribution. Since the objective function is usually specified before training, we are thus more interested in its realizations instead of its distribution and estimate the parameters.

Multiple random samples to approximate the true posterior distribution, indicating that the optimal parameters of Bayesian learning follow a potential distribution, which can make the model more robust to overfitting. By applying the Hamiltonian Monte Carlo (HMC) method, we can obtain a set of independent samples from the normalized posterior distribution and then calculate over an efficient sampling method is required to sample from the posterior distribution, which contains enough samples to perform inference. However, the traditional MCMC method has some limitations, such as requiring long time to reach the stationary distribution. Therefore, Hamiltonian Monte Carlo (HMC), which has been one of the most successful methods in MCMC for sampling from unnormalized distributions, and its variants have been chosen for sampling [17], [39].

HMC is a popular method based on the simulation of Hamiltonian dynamics and can generate a sequence of samples \( \{\xi_s\}_{s=1}^S \) from a \( U \)-dimensional distribution \( p(\xi) \). In order to generate the sample \( \xi_{s+1} \), the sample \( \xi_s \) is considered as the position of a particle and their directions of motion are determined by the Hamiltonian dynamics. The position of \( \xi_{s+1} \) is obtained by simulating the motion of the particle. HMC uses Hamiltonian dynamics to construct Markov chains and introduces a \( U \)-dimensional auxiliary momentum variable \( \theta \). HMC is only applicable to differentiable functions and real-valued unconstrained variables. However, \( \beta_\theta \) in the kernel function must be positive so that the exponential transformation \( \rho_\theta = \log(\beta_\theta) \) can be used to make \( \rho_\theta \) an unconstrained auxiliary variable. Let \( \tau = \left[ \lambda^T, \rho_0, \rho_1, \ldots, \rho_{Q+1} \right]^T \in \mathbb{R}^{(F+Q+2)} \) denote the vector of unknown parameters and \( r_{cf} = [r_{c1}[1], r_{c2}[2], \ldots, r_{cF}[N]]^T \) the request observation vector for content \( c_f \) during \( N \) time periods. Accordingly, by substituting the prior knowledge and the established model, the negative logarithm of the posterior distribution can be derived from Eq. (8) as follows:

\[
\phi(\tau) = -\log p(\lambda, \beta | R) = -\log p(R | \lambda) - \log p(\lambda | \beta) - \sum_{q=0}^{Q+1} \log(\beta_q) + \log H = \sum_{n=1}^{N} \sum_{f=1}^{F} \left( -r_{cf}[n] \lambda_f + e^{\lambda_f} + \frac{1}{2} \log det(K') \right) + \frac{1}{2} \lambda^T K'^{-1} \lambda + \sum_{q=0}^{Q+1} A_q \rho_q + B_q e^{\rho_q} + \log H.
\]

The gradients of Eq. (9) are required to perform HMC sampling, which can be calculated as follows:

\[
\nabla \phi(\tau) = -\sum_{n=1}^{N} \sum_{f=1}^{F} \nabla \log p(r_{cf}[n] | \lambda_f) - \nabla \log p(\lambda | \beta),
\]

\[
\frac{\partial \phi(\tau)}{\partial \rho_q} = \sum_{n=1}^{N} -r_{cf}[n] + Ne^{\lambda_f} + [K'^{-1} \lambda]_f,
\]

\[
\frac{\partial \phi(\tau)}{\partial \lambda_f} = \frac{1}{2} tr \left( K'^{-1} \frac{\partial K'}{\partial \rho_q} \right) - \frac{1}{2} \lambda^T K'^{-1} \frac{\partial K'}{\partial \rho_q} K'^{-1} \lambda - A_q + B_q e^{\rho_q}.
\]
follows):

The SVRG-HMC based sample update can be expressed as which can also reduce the convergence speed of HMC algo-

ration can be divided into inner and outer loops, where the outer loop performs $S/L - 1$ rounds of the global gradient update. In the inner loop, $L$ rounds of minibatch stochastic gradient calculations are performed. When $b = 1$, the inner loop degrades into simple stochastic gradient descent. In the $l$-th iteration of the inner loop, the stochastic gradient is computed as follows:

$$g_l = \frac{1}{L} \sum_{i \in I} (\nabla \delta_i (\xi_{sL+l}) - \nabla \delta_i (\omega)) + g,$$  

where $g$ is the gradient value obtained by traversing the entire dataset in the outer loop, and $\frac{1}{L} \sum_{i \in I} (\nabla \delta_i (\xi_{sL+l}) - \nabla \delta_i (\omega))$ is used to correct the gradient. Clearly, at the beginning of the inner loop, the variance of the estimate $\frac{1}{L} \sum_{i \in I} (\nabla \delta_i (\xi_{sL+l}) - \nabla \delta_i (\omega))$ is small, and thus the variance in estimating the entire gradient is also small. As $l$ increases, the iterations of the inner loop continue and the variance increases, and then the inner loop is jumped to reset $\omega$ to estimate the new gradient value for training. The number of inner loops can be flexibly adjusted according to different convergence rate and computational complexity constraint.

Algorithm 1 can reduce computational complexity by using stochastic gradient rather than traversing the entire dataset in each iteration. In the HMC sampling method, it requires $S \cdot (N^2F^2)$ gradient computations to get $S$ samples. However, our proposed SVRG-HMC sampling method only needs to traverse the entire dataset in the outer loop and use stochastic gradient in the inner loop, which only requires $S \cdot (N^2F + bL)$ gradient computations to get $S$ samples. Therefore, our proposed sampling method reduces $S \cdot (N^2F^2 - N^2F - bL)$ gradient computations, achieving lower computational complexity.

D. Global Model Integration Phase

In this subsection, we propose a communication-efficient FL based global model integration method. Firstly, we introduce a sampling method that simply combines Bayesian learning and FL. Secondly, we employ gradient quantization to achieve tradeoff between prediction accuracy and communication overhead. Thirdly, we adopt an encoding scheme for transmitting quantized gradient efficiently. Finally, we propose a communication-efficient global model integration for the federated Bayesian learning sampling method.

1) Native Federated Bayesian Learning Sampling Method: The content popularity prediction method based on Bayesian learning can carry out model training and prediction on a single F-AP. However, the limitation of this method is the small coverage of a single F-AP and the constraints of computing and caching resource. To solve the above problems and take advantage of the huge amount of data in F-RANs, cooperative prediction participated by neighboring F-APs needs to be considered [15].

From Eq. (9), we can infer that $K'$ and $\lambda$ are related to the dimension of the local input dataset and their computational costs in Eq. (10) are fixed. Let $R_m$ denote the local dataset of the $m$-th F-AP. Note that $R_m$ is disjoint for different $m$. Then, the global posterior can be written as follows:

$$p(\lambda, \beta | R) \propto p(\lambda, \beta) \prod_{m=1}^{M} p(R_m | \lambda, \beta).$$  

\begin{algorithm}
\caption{The Proposed SVRG-HMC Sampling Method.} \label{alg1}
\begin{algorithmic}[1]
\State $S, L, b, h > 0, D_h < 1, D \geq 1$;
\State \textbf{Output:} $\{\tau_s\}_{L \leq S}$;
\State \textbf{Initialize} $\theta_0, \xi_0$;
\For {$s = 0, 1, \ldots, S/L - 1$}
\State $g = -\frac{\eta}{N} \sum_{i=1}^{NF} \nabla \log p(r_{c_i} | n) \lambda_f$;
\State $\omega = \xi_{sL}$;
\EndFor
\For {$l = 0, 1, \ldots, L - 1$}
\State Randomly draw a subset $I \subseteq R, |I| = b$ with equal probability;
\State $\nabla \phi (\tau) = -|R| \sum_{r \in R} \nabla \log p(r_{c_i} | n) \lambda_f$;
\State $-\nabla \log p(\lambda | \beta) - \sum_{q=0}^{Q+1} \nabla \log p(\beta_q)$. \hspace{1cm} (11)
\State $\theta_{sL+l} = (1 - Dh) \theta_{sL+l} - h \nabla \phi (\tau) + \sqrt{2D_h} \cdot \eta_l$;
\State $\xi_{sL+l+1} = \xi_{sL+l} + h \theta_{sL+l+1}$;
\EndFor
\State $\tau_s = \xi_{sL+L}$;
\end{algorithmic}
\end{algorithm}
According to the previous definitions: \( \rho_q = \log (\beta_q) \), \( \tau = [\tau^T, \rho_0, \ldots, \rho_{Q+1}]^T \in R^{(F+Q+2)} \), Eq. (14) can be re-written as follows:

\[
p(\tau | R) \propto p(\tau) \prod_{m=1}^{M} p(R_m | \tau).
\] (15)

Therefore, sampling from the global posterior can be naturally considered as a federated Bayesian learning sampling problem [42]. By utilizing our proposed SVRG-HMC sampling method employed in each F-AP, the sampling updates at the cloud server can be expressed as follows:

\[
\theta_{k+1} = (1 - Dh) \theta_{k+1} - \frac{h}{M} \sum_{m=1}^{M} H_{k+1}^m (\tau_k) + \sqrt{2Dh} \cdot \eta_{k+1}, \\
\tau_{k+1} = \tau_k + h\theta_{k+1}.
\] (16)

where \( H_{k+1}^m (\tau_k) \) represents the unbiased estimate of the gradient \( \nabla \log p(\tau | R) \).

In the FL framework, the \( m \)-th F-AP obtains the gradient estimate \( H_{k+1}^m \) at the \( k \)-th iteration based on \( R_i \), which is a subset of the local dataset \( R_m \). Then, \( H_{k+1}^m = \sum_{i=1}^{b} H_{k+1}^i \) calculates the gradient estimate of the whole dataset. Also, we assume that \( H_{k+1}^m \) is the unbiased estimate of the global gradient while \( H_{k+1}^i \) is not an unbiased estimator, which helps to use a biased local stochastic gradient with better convergence guarantee [42]. Correspondingly, Eq. (16) can be re-written as follows:

\[
\theta_{k+1} = (1 - Dh) \theta_{k+1} - \frac{h}{M} \sum_{m=1}^{M} H_{k+1}^m (\tau_k) + \sqrt{2Dh} \cdot \eta_{k+1}, \\
\tau_{k+1} = \tau_k + h\theta_{k+1}.
\] (17)

By applying the model training method of a single F-AP, the native federated Bayesian learning sampling method is presented in Algorithm 2, where each F-AP computes gradients based on its local dataset and sends them to the cloud server, and the cloud server then performs federated averaging and updates the samples. Note that by exchanging the computed gradients instead of datasets, the privacy preservation of local data at each F-AP is also achieved. This method integrates FL and Bayesian learning effectively. The flexibility of FL allows each F-AP to generate local model through Bayesian learning while FL further exploits the computing resource of each F-AP and the data in RANs together.

2) Gradient Quantization: If the gradient vectors are dense, each participant needs to send and receive multiple floating point numbers to each other in each iteration to update the parameter vectors. In practical applications, transmitting gradients in each iteration is a significant performance bottleneck.

In this phase, we adopt a gradient quantization scheme. Let \( v \in \mathbb{R}^d \) denote the gradient vector passed in FL, \( \|v\|_2 \) the number of nonzero elements in \( v \) and \( \|v\|_2 = \sqrt{v_1^2 + v_2^2 + \ldots + v_d^2} \). Let \( F \) denote the number of bits required to encode a floating number (typically 32 or 64 bits, considered to be a constant in the following). For any binary code string \( w \), let \( |w| \) denote its length. For any scalar, let \( \text{sgn}(x) \in \{-1, +1\} \) denote its sign and \( \text{sgn}(0) = 1 \).

Define a random quantization function such that for any \( d \)-dimensional vector \( v \in \mathbb{R}^d \) with \( v \neq 0 \), the \( i \)-th element of its quantized vector \( Q(v) \) can be expressed as \( Q_i(v, s) = \|v\|_2 \cdot \text{sgn}(v_i) \xi_i(v, s) \) [43], where \( s \) is a quantization level, and \( \xi_i(v, s) \) is an independent random variable. Let \( l_i \) denote a positive integer such that \( 0 \leq l_i < s \) and \( \frac{|v_i|}{\|v\|_2} \in \left[ \frac{l_i}{s}, \frac{l_i+1}{s} \right] \). Define \( \xi_i(v, s) \) as follows:

\[
\xi_i(v) = \begin{cases} 
\frac{l_i}{s} & p = 1 - q \left( \frac{|v_i|}{\|v\|_2}, s \right) \\
\frac{l_i+1}{s} & p = q \left( \frac{|v_i|}{\|v\|_2}, s \right),
\end{cases}
\] (18)

where \( q \left( \frac{|v_i|}{\|v\|_2}, s \right) = \frac{s |v_i|}{\|v\|_2} - l_i \). According to the relationship of \( \frac{|v_i|}{\|v\|_2} \in \left[ \frac{l_i}{s}, \frac{l_i+1}{s} \right] \), \( q \left( \frac{|v_i|}{\|v\|_2}, s \right) \in [0, 1] \) can be established, which satisfies the definition of probability distribution. Then, we can establish the following lemma.

**Lemma 1:** For any \( v \in \mathbb{R}^d \), the following relationships hold:

\[
\mathbb{E}[\|Q(v, s)\|_2] \leq \sqrt{d} + s^2, \\
\mathbb{E}[\|Q(v, s)\|^2] \leq \left( 1 + \min \left( \frac{d}{s^2}, \frac{\sqrt{d}}{s} \right) \right) \|v\|_2^2.
\] (19) (20) (21)

**Proof:** Please see Appendix A.

**Lemma 1** demonstrates the sparsity, unbiasedness and the upper bound on the estimated variance of \( Q(v, s) \).

It can be observed that the properties of \( Q(v, s) \) is determined by \( s \). As shown in Fig. 3, the purpose of \( s \) is to
provide a higher quantization accuracy, which is equivalent to dividing the quantization result into discrete $s$ points from the coordinate. For each $v_i$, the corresponding $l_i$ is selected so that $\frac{|v_i|}{\|v\|_2}$ falls within the interval $\left[\frac{l_i}{s}, \frac{l_i+1}{s}\right]$. The left and right endpoints of the interval are two quantization levels. The larger $s$ is, the smaller the corresponding quantization accuracy $\frac{1}{s}$ is and the more accurate the result is. A larger $\frac{|v_i|}{\|v\|_2}$ leads to a greater probability of being quantized to the right endpoint of the interval, i.e., $\frac{l_i+1}{s}$. Similarly, a smaller $\frac{|v_i|}{\|v\|_2}$ leads to a greater probability of being quantized to the left endpoint of the interval, i.e., $\frac{l_i}{s}$, which is consistent with the intuitive perception of quantization.

From Lemma 1, we can infer that the upper bound of $\mathbb{E}\|Q(v, s)\|_2^2$ is affected by the vector dimension $d$ and the quantization level $s$. When $s = 1$, the upper bound of $\mathbb{E}\|Q(v, s)\|_2^2$ is $(1 + \sqrt{d}) \|v\|_2^2$. When $s$ is adjusted from 1 to $\sqrt{d}$, the upper bound of $\mathbb{E}\|Q(v, s)\|_2^2$ varies between $O(\sqrt{d})$ and $O(1)$. It can be observed that the sparsity of the quantized gradient decreases as $s$ increases. Therefore, for a given gradient vector dimension, a suitable quantization level $s$ needs to be specified for quantization to obtain a tradeoff between prediction accuracy and communication overhead. Then, we can establish the following theorem.

**Theorem 1:** For any $v \in \mathbb{R}^d$ and $s^2 + \sqrt{d} \leq \frac{d}{2}$, there exists a coding scheme which allows the upper bound on the number of bits required to transmit $Q(v, s)$ to be expressed as:

$$F + 3 + \frac{3}{2} (1 + O(1)) \log_2 \left(\frac{2(s^2 + d)}{s^2 + \sqrt{d}}\right) \left(s^2 + \sqrt{d}\right).$$

(22)

**Proof:** Please see Appendix B.

Theorem 1 shows the upper bound of communication overhead for transmitting $Q(v, s)$. In summary, for any gradient $v$, its quantizer output $Q(v, s)$ can be represented by a tuple $([|v|_2], \gamma, \varepsilon)$, where $\gamma$ is the vector representing the sign of each element $v_i$ of $v$, and $\varepsilon$ is the vector representing the integer $s \cdot \xi_i(v, s)$. By encoding these messages, communication overhead can be greatly reduced.

3) **Coding Scheme:** In practical applications, the value of $s$ should be small enough to maintain sparsity in Lemma 1, so the element $s \cdot \xi_i(v, s)$ in $\varepsilon$ is usually not a big integer [42]. If a fixed-length encoding scheme is used to represent these values, each of them requires a full 32-bit integer, but the number of their effective bits is usually only 10 or even less, resulting in a great waste of communication resource. Therefore, when the quantization results are transmitted, a suitable variable-length encoding scheme is required.

**Algorithm 3** Elias Coding Method.

**Input:** Integer $k$;

**Output:** Coding string Elias $(k)$;

1: Add a zero to the right of the coding string;
2: while $|\log_2 k| > 0$ do
3: Write the binary encoding result of $k$ on the left of the coding string;
4: $k \leftarrow |\log_2 k|$;
5: end while

(23)

Let $Q(v, s)$ denote the function that maps from $\mathbb{R}^d$ to $B_s$, where $B_s$ is defined as follows:

$$B_s = \left\{(A, \gamma, y) : A \in \mathbb{R}, \gamma_i \in \{-1, +1\}, y_i \in \left\{0, \frac{1}{s}, \ldots, 1\right\}\right\}.$$  

Note that the tuple $B_s$ contains all the information of the gradient $v$. Therefore, we can encode $B_s$ instead of $v$. There are some classic variable-length encoding schemes such as Shannon coding [44] and Huffman coding [45]. However, both of them need to know the appearing probability of each integer in advance, which is not applicable to our proposed quantization scheme. Instead, Elias coding is an efficient method to encode integer without the prior knowledge of the appearing probability of each integer [46]. The procedure of Elias coding is presented in Algorithm 3. For input $k$, Elias $(k)$ is the output Elias coding string. For example, Elias $(1) = 0$, Elias $(8) = 1110000$. The decoding step simply involves an inverse transformation from left to right of Elias $(k)$ to determine the original value. Then, we have the following lemma which describes a well-known property of Elias coding [47].

**Lemma 2:** For any positive integer $k$, we have:

$$|\text{Elias}(k)| \leq \log_2 k + \log_2 (\log_2 k) + \cdots + 1 = (1 + O(1)) \log_2 k + 1.$$

(24)

Lemma 2 shows the upper bound of the number of bits required to encode a positive integer $k$ by Elias coding.

Given a tuple $(A, \gamma, y) \in B_s$, let $\text{Code}(A, \gamma, y)$ denote the number of bits required for transmitting the tuple, where $A$ is a floating number which requires $F$ bits to encode. Then, the Elias coding scheme can be described as follows.

First, the location of the first non-zero coordinate in $y$ is encoded through Elias coding. After that, a bit representing $\gamma_i$ is appended and followed by Elias $(s|y_{i-\text{next}}|)$. Let $c(y_i)$ denote the distance from the current coordinate of $y$ to the next non-zero coordinate $y_{i-\text{next}}$. Iteratively, encode $c(y_i)$ through Elias coding. $\gamma_i$-next and $y_{i-\text{next}}$ of that coordinate are encoded through Elias coding in the same way. The decoding steps are similar: first read the $F$ bits to reconstruct $A$, and then iteratively use the decoding scheme for Elias coding to read off the positions and values of the non-zero elements of $\gamma$ and $y$.

By using the Elias encoding scheme, bit waste can be greatly reduced. Therefore, introducing Elias coding into our federated Bayesian learning sampling framework is another...
According to our proposed probabilistic model, we can obtain the mean of the predictive distribution and approximate it as follows:

\[ \mathbb{E}(r_c | N + 1) = \frac{1}{S} \sum_{s=1}^{S} e^{\lambda(s)}. \] (26)

### Algorithm 4 Communication-Efficient Federated Bayesian Learning Sampling Method

**Input:** \( S, L, b, h > 0, D_h < 1, D \geq 1 \)

**Output:** \( \{\tau_s\}_{1 \leq s \leq S} \)

1. Initialize \( \{\theta_0^{(1)}, \ldots, \theta_0^{(b)}\}, \{\tau_0^{(1)}, \ldots, \tau_0^{(b)}\} \);
2. for \( s = 0, 1, \ldots, S \) do
3. Choose participants \( A_{s+1} \);
4. for \( i \in A_{s+1} \) do
5. (in parallel over nodes)
6. \( g = -\frac{1}{NF} \sum_{i=1}^{NF} \nabla \log p(r_c \mid \lambda) = \frac{1}{NF} \sum_{i=1}^{NF} \nabla\delta_i (\xi) \);
7. \( \omega = \tau_s^{(b)} \);
8. for \( l = 0, 1, \ldots, L - 1 \) do
9. Randomly draw a subset \( I \subseteq \mathcal{R}, |I| = b \) with equal probability:
10. \( \tilde{\nabla}^{(i)}_s = -\log p(\tau_s^{(i)}) - \sum_{q=0}^{Q} \log p(\tau_s^{(i)}) + \frac{NF}{b} \sum_{i \in I} \left( \nabla\delta_i (\tau_s^{(i)}) - \nabla\delta_i (u) \right) + g \);
11. Quantize and encode to obtain \( \mathcal{L}(\tilde{\nabla}^{(i)}_s) \);
12. Send \( \mathcal{L}(\tilde{\nabla}^{(i)}_s) \) to the cloud server;
13. end for
14. end for
15. The cloud server decodes \( \mathcal{L}(\tilde{\nabla}^{(i)}_s) \) to obtain \( \tilde{\nabla}^{(i)}_s^* \);
16. Federated averaging \( \tilde{\nabla}_s = \frac{1}{|A_{s+1}|} \sum_{i \in A_{s+1}} \tilde{\nabla}^{(i)}_s^* \);
17. Draw \( \eta_{s+1} \sim \mathcal{N}(0, \mathcal{I}) \);
18. \( \theta_{s+1} = (1 - Dh) \theta_s - h \tilde{\nabla}_s + \sqrt{2Dh} \cdot \eta_{s+1} \);
19. \( \tau_{s+1} = \tau_s + h \theta_{s+1} \);
20. Send \( \tau_{s+1} \) to \( M \) F-APs;
21. end for

For the second case, the posterior distribution can be derived as follows:

\[ p(\lambda_{F+1} | x_{F+1}) = \int p(\lambda_{F+1} | \lambda, \beta, x_{F+1}) \cdot p(\lambda, \beta | \mathcal{R}) \, d\beta d\lambda, \] (27)

where \( x_{F+1} \) represents the feature vector of a newly-added content. Note that the joint distribution of \( p(\lambda_1, \lambda_2, \ldots, \lambda_{F+1}) \) is a normal distribution with zero mean and covariance matrix as follows:

\[ \begin{bmatrix} K' \\ k'^T \end{bmatrix} = K(x_{F+1}, x_{F+1}) \],

(28)where \( k' = \{K(x_1, x_{F+1}), K(x_2, x_{F+1}), \ldots, K(x_F, x_{F+1})\}^T \). According to the property of Gaussian distribution, the conditional distribution \( p(\lambda_{F+1} | \lambda, x_{F+1}, \beta) \) should be a normal distribution with mean and variance as follows:

\[ \lambda_{F+1} = k'^T K'^{-1} \lambda, \quad \sigma_{F+1} = K(x_{F+1}, x_{F+1}) - k'^T K'^{-1} k'. \] (29)

Similarly, the point estimation of the request rate for a newly-added content can be approximated as follows:

\[ \mathbb{E}(r_{F+1} | N + 1) = \frac{1}{S} \sum_{s=1}^{S} e^{\lambda_{F+1}(s)} + \frac{1}{2} \sigma_{F+1}(s). \] (30)
According to the above descriptions, the procedure of our proposed popularity prediction policy is shown in Fig. 4. In the model learning phase, the model parameters are learned based on the number of requests and the features of contents stored in the cloud server (referred to as seen contents). In the predicting phase, the proposed policy enables to predict the popularities of both seen and unseen contents (i.e., newly-added contents). Besides, considering that content popularity can be a temporal variant, we can shorten the time period to track content popularity in peak hours.

IV. SIMULATION RESULTS

To evaluate the performance of our proposed popularity prediction policy, we take movie contents as an example and perform simulations using the data extracted from the MovieLens 100K Dataset [50]. The dataset consists of 100,000 ratings (1-5) from 943 users on 1,682 movies, and is collected during the seven-month period from September 19th, 1997 to April 22nd, 1998. Each item contains the information about the movies including the features of rating, category and timestamp. We take the rating as the number of requests for the movies. Besides, we set the number of the considered F-APs $M$ to 5, the preset time period to 12 hours and the dimension of content feature vector to 10, respectively. Therefore, the whole time interval of the MovieLens 100K Dataset can be split into more than 400 time slots. For convenience, a summary of major simulation parameters is presented in Table II.

In order to validate our theoretical results, our simulation results include two parts. First, we compare our proposed Bayesian learning based prediction policy in local training phase with traditional Bayesian learning based prediction policy in Fig. 5 and Fig. 6. These policies are performed on a single F-AP. The aim of this part is to validate the advantages of introducing SVRG to the traditional sampling method in Bayesian learning. The reminder of simulation results is to show the superiority of our proposed quantized federated Bayesian learning based content popularity prediction policy. The auto regressive (AR) process based policy, LFU, LRU and random caching (RC) are chosen as the baseline policies.

In Fig. 5, we show the RMSE of HMC, stochastic gradient HMC (SGHMC) and SVRG-HMC based local content popularity prediction policy versus training epoch. These policies are performed over a single F-AP without aggregation. It can be observed that SVRG-HMC and SGHMC take less training epoches to converge to the same accuracy, and reach higher accuracies in shorter epoches, which can satisfy the requirements of real-time prediction of content popularity. The reason is that SVRG-HMC and SGHMC utilize stochastic gradients to iterate only a small portion of the whole dataset. Besides, The introduction of variance reduction also accelerates the convergence due to the simple use of stochastic gradient in SGHMC.

In Fig. 6, we show the RMSE of HMC, SGHMC and SVRG-HMC based local content popularity prediction policy versus the number of request observations $N$. It can be observed that SVRG-HMC and SGHMC take less training epoches to converge to the same accuracy, and reach higher accuracies in shorter epoches, which can satisfy the requirements of real-time prediction of content popularity. The reason is that SVRG-HMC and SGHMC utilize stochastic gradients to iterate only a small portion of the whole dataset. Besides, The introduction of variance reduction also accelerates the convergence due to the simple use of stochastic gradient in SGHMC.

In Fig. 6, we show the RMSE of HMC, SGHMC and SVRG-HMC based local content popularity prediction policy versus the number of request observations $N$. It can be observed that the RMSE of our proposed SVRG-HMC based local content popularity prediction policy and the other HMC based policies decreases as the number of request observations increases. The reason is that Gaussian process in the probabilistic model allows for a better insight into the

| $M$ (number of F-APs) | 5 |
|----------------------|---|
| $F$ (number of contents) | \{100, 150\} |
| $Q$ (dimension of the feature vector) | 10 |
| $N$ (number of the observed time periods) | [50, 200] |
| $s$ (quantization level) | \{1024, 4096\} |
relationship between the number of requests and the content features by obtaining more observation samples. It can also be observed that the RMSE of SVRG-HMC is significantly smaller than that of SGHMC and close to that of HMC. The reason is that stochastic gradient introduces prediction error, and SVRG-HMC utilizes variance reduction to alleviate this impact. Specifically, it can be observed that the RMSE decreases as the number of contents in the library $F$ increases.

In Fig. 7, we compare our proposed quantized federated Bayesian learning (QFBL) based predicting policy with the unquantized federated Bayesian learning (FBL) based policy and the AR process based policy [51]. The AR process based policy is a state-of-the-art content popularity prediction algorithm which derives regression parameters based on least-squares estimates and can achieve a high prediction accuracy. It can be seen that similar to what is shown in Fig. 6, the RMSE gradually decreases as the observation increases. It can also be seen that when the quantization level $s$ increases, the prediction error is significantly reduced. When $s = 1024$, the prediction result is already much better than the AR process based policy. When $s = 4096$, the performance of the proposed policy is nearly the same as that of the unquantized case. This is because when $s$ increases, the quantization interval becomes larger, which increases the accuracy of the prediction after quantization.

In Fig. 8, we compare the performance of our proposed policy with centralized learning based policy when the quantization level $s = 1024$ and $s = 4096$. It can be seen that our proposed policy achieves a higher accuracy after a few iterations. This is because FL framework can iterate and then aggregate each existing local model obtained from an iteration of local training in each F-AP separately to generate the global model. While in centralized learning, only a single iteration of training is performed with a fixed amount of data sampled from all datasets. Furthermore, compared with random generation in centralized learning, the global model can be initialized by the local models in QFBL, which can also accelerate the convergence of QFBL in early training. After reaching convergence, our proposed policy can approach the performance of centralized learning.

In Fig. 9, we show the effects of quantization on communications. Firstly, we set 0.05 as the predefined objective prediction accuracy. It can be seen that QFBL can achieve better accuracy with less communication overhead. When $s = 2048$ and $s = 4096$, it costs nearly 30% and 40% less bits to converge to the predefined accuracy than the unquantized FBL based policy, respectively. When $s = 1024$, the accuracy is higher at the beginning of training. It is because a smaller $s$ allows to perform more iterations using the same number of communications bits, but the convergence speed is slower than those of the others due to the larger quantization error brought by a smaller $s$. From the communications perspective, QFBL provides a new method to reduce communication overhead, which can adjust the value of $s$ flexibly according to the predefined objective prediction accuracy.
In Fig. 10, we show the cache hit rates of our proposed QFBL policy and four baseline policies versus normalized cache size, which is defined by the ratio of cache size to the size of the content library. Assume that the AR process based policy and our proposed policy cache the most popular contents according to the prediction results. It can be observed that the cache hit rates of all policies increase significantly with the normalized cache size. It can also be observed that the proposed policy and the AR process based policy have better performance than LRU, LFU and RC due to their utilization of content popularity. Besides, our proposed policy outperforms the AR process based policy and is very close to the optimal policy.1 The reason is that our proposed policy leverages content features to predict the popularity of newly-added contents.

V. CONCLUSION

In this paper, we have proposed the content popularity prediction policy based on quantized federated Bayesian learning in F-RANs. By using the number of requests and content features, our proposed policy enables prediction of content popularity with lower computational complexity and communication overhead by utilizing the relationship between content features and popularity in our proposed probabilistic model. At the same time, we have employed FL to leverage the data and computing resource of multiple F-APs to accelerate the convergence. We have also proposed a quantized and encoded federated Bayesian learning sampling method that can efficiently trade off between prediction accuracy and communication overhead. We have theoretically analyzed the upper bound of the number of bits required by our proposed policy. The quantization level determines the communication overhead of our proposed policy where a larger quantization level leads to a higher prediction accuracy but more communication overhead. Simulation results have shown that our proposed policy can significantly reduce communication overhead and accelerate the convergence while maintaining the prediction accuracy. Our future work will explore the possibility of extending QFBL to the clustering problem in multi-region-covering scenario.

APPENDIX A

Proof of Lemma 1

Set \( u = \frac{v}{\|v\|_2} \) and let \( I(u) \) denote the set of \( u_i \leq \frac{1}{s} \). Then, we have:

\[
(n - |I(u)|)/s \leq \sum_{i \notin I(u)} u_i^2 \leq 1.
\]

Since \( s \) is positive, it is obvious that \( n - |I(u)| \leq s^2 \). Besides, for any \( i \in I(u) \), \( Q_i(v, s) \) takes a non-zero value with probability of \( u_i \). Therefore, we have:

\[
\mathbb{E}[\|Q(v, s)\|_0] \leq d - |I(u)| + \sum_{i \in I(u)} u_i \leq s^2 + \|u\|_1 \leq s^2 + \sqrt{d}.
\]

The unbiasedness can be verified as follows:

\[
\mathbb{E}[\xi_i(v, s)] = \left(1 + s \frac{|v_i|}{\|v\|_2^2}\right) - \left(\frac{s |v_i|}{\|v\|_2^2} - 1\right) \frac{l + 1}{s} = \frac{|v_i|}{\|v\|_2^2}.
\]

From the definition of \( \xi_i(v, s) \), we have:

\[
\mathbb{E}[\xi_i(v, s)]^2 = \mathbb{E}[\xi_i(v, s)]^2 + \mathbb{E}[(\xi_i(v, s) - \mathbb{E}[\xi_i(v, s)])^2] = \frac{v_i^2}{\|v\|_2^2} \left(1 + \frac{1}{s^2} q \left(\frac{|v_i|}{\|v\|_2^2}, s\right)\right) \left(1 - q \left(\frac{|v_i|}{\|v\|_2^2}, s\right)\right)
\]

\[
\leq \frac{v_i^2}{\|v\|_2^2} + \frac{1}{s^2} q \left(\frac{|v_i|}{\|v\|_2^2}, s\right).
\]

\[
\leq \frac{v_i^2}{\|v\|_2^2} + \frac{1}{s^2} q \left(\frac{|v_i|}{\|v\|_2^2}, s\right).
\]

Besides, according to \( q(a, s) \leq 1 \) and \( q(a, s) \leq a s \), the upper bound of the estimate error can be verified as follows:

\[
\mathbb{E}[\|Q(v, s)\|_0^2] = \sum_{i=1}^{d} \mathbb{E}[\|Q(v, s)\|_0] = \sum_{i=1}^{d} \left(\frac{|v_i|^2}{\|v\|_2^2} + s^2 q \left(\frac{|v_i|}{\|v\|_2^2}, s\right)\right) \leq \left(1 + \frac{1}{s^2}\right) \sum_{i=1}^{d} q \left(\frac{|v_i|}{s \|v\|_2^2}, s\right) \|v\|_2^2 \leq \left(1 + \min \left(\frac{d}{s^2}, \frac{\sqrt{d}}{s}\right)\right) \|v\|_2^2.
\]

This completes the proof.

APPENDIX B

Proof of Theorem 1

We need two auxiliary lemmas to prove Theorem 1.

Lemma 3: For any vector \( q \in \mathbb{R}^d \), \( q_i > 0 \) and \( \|q\|_p \leq \rho \), we have:

\[
\sum_{i=1}^{d} \text{Elias}(q_i) \leq \left(1 + O\left(\frac{1}{p}\right) \log_2 \left(\frac{\rho}{d}\right) + 1\right) d.
\]

Proof: For any positive integer \( k \), the length of Elias(k) is less than \( (1 + O\left(\frac{1}{p}\right)) \log_2 k + 1 \) [47]. Then, we have:

\[
\sum_{i=1}^{d} \text{Elias}(q_i) \leq (1 + O\left(\frac{1}{p}\right)) \sum_{i=1}^{d} (\log_2 q_i^p) + d
\]

\[
\leq (1 + O\left(\frac{1}{p}\right)) \sum_{i=1}^{d} (\log_2 (q_i^p)) + d.
\]

According to Jensen inequality, we have:

\[
\sum_{i=1}^{d} \text{Elias}(q_i) \leq (1 + O\left(\frac{1}{p}\right)) d \log_2 \left(\frac{1 + O\left(\frac{1}{p}\right)}{d} \sum_{i=1}^{d} q_i^p\right) + d
\]

\[
\leq (1 + O\left(\frac{1}{p}\right)) d \log_2 \left(\frac{\rho}{d}\right) + d
\]

\[
= (1 + O\left(\frac{1}{p}\right)) \log_2 \left(\frac{\rho}{d}\right) + 1\right) d.
\]

The optimal policy will cache the most popular contents according to the real content popularity.
Therefore, the number of bits required for encoding can be reduced by limiting the number of non-zero elements.

Lemma 4: For any tuple \((A, \gamma, y) \in B_s\), the maximum number of bits that Code\(\{A, \gamma, y\}\) contains can be expressed as follows:

\[
F + \left(1 + O(1)\right) \cdot \log_2 \left(\frac{d}{\|y\|_0} \right) + 3 \cdot \|y\|_0 - 1,
\]

Proof: The float number \(A\) requires \(F\) bits for communications (without Elias coding). \(\gamma\) and \(y\) can be divided into two parts. First, there is a subsequence \(S_1\) defining the non-zero coordinate of \(y\). Secondly, there is a subsequence \(S_2\) dedicated to representing the sign bit and \(c(y_i)\) for each non-zero coordinate \(i\). Although these two subsequences are not consecutive in the string, they clearly divide the remaining bits of the string. The lengths of these two substrings are calculated separately. The information of \((A, \gamma, y) \in B_s\) can be expressed by \(F\), subsequence \(S_1\) and \(S_2\).

For \(S_1\), let \(i_1, \ldots, i_{\|y\|_0}\) denote the positions of the non-zero coordinate in \(y\). Then, \(S_1\) consists of the encoding of the vector \(q^{(1)} = [i_1, i_2 - i_1, \ldots, i_{\|y\|_0} - i_{\|y\|_0 - 1}]^T\). Each element of \(q^{(1)}\) is encoded with a distance function \(c\). According to Lemma B.1, since the vector \(q^{(1)}\) has length \(\|y\|_0\) and \(\|q^{(1)}\|_1 \leq d\), we have:

\[
|S_1| \leq \left(1 + O(1)\right) \log_2 \left(\frac{d}{\|y\|_0} \right) + 1 \cdot \|y\|_0.
\]

For \(S_2\), each non-zero coordinate is required to pass the sign and the encoding result of \(s_j y_i\), so the length of \(S_2\) can be determined by the two parts. According to Lemma B.1, we have:

\[
|S_2| = \sum_{j=1}^{\|y\|_0} (1 + |Elias| (s_j y_i)) \leq \|y\|_0 + \frac{1 + O(1)}{2} \log_2 \left(\frac{s^2 \|y\|_0^2}{\|y\|_0} \right) + 1 \cdot \|y\|_0. \tag{41}
\]

According to Eq. (40) and Eq. (41), Lemma B.2 can then be proved.

Based on Lemma B.2 and the sparsity of \(Q (v, s)\), we have:

\[
\mathbb{E}[\text{Code}(Q (v, s))] \leq F + \left(1 + O(1)\right) \mathbb{E}[\|\varepsilon\|_0 \log_2 \left(\frac{d}{\|\varepsilon\|_0} \right)] + \frac{1 + O(1)}{2} \mathbb{E}[\|\varepsilon\|_0 \log_2 \left(\frac{s^2 \|y\|_0^2}{\|\varepsilon\|_0} \right)] + 3 \mathbb{E}[\|\varepsilon\|_0] \leq F + \left(1 + O(1)\right) \mathbb{E}[\|\varepsilon\|_0 \log_2 \left(\frac{d}{\|\varepsilon\|_0} \right)] + \frac{1 + O(1)}{2} \mathbb{E}[\|\varepsilon\|_0 \log_2 \left(\frac{2(s^2 + d)}{\|\varepsilon\|_0} \right)] + 3 \left(s^2 + \sqrt{d}\right). \tag{42}
\]

Clearly, \(f(x) = x \log_2 \left(\frac{x}{2}\right)\) is a concave function at \(C > 0\), which is an increasing function until \(x = \frac{C}{2}\) and then decreases gradually. According to Jensen’s inequality, the sparsity of \(Q (v, s)\) and the assumption \(s^2 + \sqrt{d} \leq \frac{2}{4}\), we have:

\[
\mathbb{E}\left[\|\varepsilon\|_0 \log_2 \left(\frac{d}{\|\varepsilon\|_0} \right)\right] \leq s^2 + \sqrt{d} \log_2 \left(\frac{d}{s^2 + \sqrt{d}}\right), \tag{43}
\]

\[
\mathbb{E}\left[\|\varepsilon\|_0 \log_2 \left(\frac{2(s^2 + d)}{\|\varepsilon\|_0} \right)\right] \leq (s^2 + \sqrt{d}) \log_2 \left(\frac{2(s^2 + d)}{s^2 + \sqrt{d}}\right). \tag{44}
\]

Then, Theorem 1 can be proved by combining Eq. (42), Eq. (43) and Eq. (44). This completes the proof.

REFERENCES

[1] X.-H. You et al., “Towards 6G wireless communication networks: Vision, enabling technologies, and new paradigm shifts,” Sci. China Inf. Sci., vol. 64, no. 1, pp. 1–74, Jan. 2021.

[2] Y. Jiang, B. Wang, F.-C. Zheng, M. Bennis, and X. You, “Joint MDS codes and weighted graph-based coded caching in fog radio access networks,” IEEE Trans. Wireless Commun., vol. 21, no. 9, pp. 6789–6802, Sep. 2022.

[3] J. Yan, Y. Jiang, F. Zheng, F. R. Yu, X. Gao, and X. You, “Distributed edge caching with content recommendation in fog-RANs via deep reinforcement learning,” in Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops), Jun. 2020, pp. 1–6.

[4] C. Aggarwal, J. L. Wolf, and P. S. Yu, “Caching on the world wide web,” IEEE Trans. Knowl. Data Eng., vol. 11, no. 1, pp. 94–107, Jan. 1999.

[5] H. Ailehagh and S. Dey, “Video caching in radio access network: Impact on delay and capacity,” in Proc. IEEE Wireless Commun. Netw. Conf. (WCNC), Apr. 2012, pp. 2276–2281.

[6] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. C. M. Leung, “Cache in the air: Exploiting content caching and delivery techniques for 5G systems,” IEEE Commun. Mag., vol. 52, no. 2, pp. 131–139, Feb. 2014.

[7] Y. Jiang et al., “Analysis and optimization of cache-enabled fog radio access networks: Successful transmission probability, fractional offloaded traffic and delay,” IEEE Trans. Veh. Technol., vol. 69, no. 5, pp. 5219–5231, May 2020.

[8] Y. Jiang et al., “Analysis and optimization of fog radio access networks with hybrid caching: Delay, energy efficiency,” IEEE Trans. Wireless Commun., vol. 20, no. 1, pp. 69–82, Jan. 2021.

[9] S. Rathore, J. Ryu, P. Sharma, and J. Park, “DeepCachNet: A proactive caching framework based on deep learning in cellular networks,” IEEE Netw., vol. 33, no. 3, pp. 130–138, May 2019.

[10] Y. Jiang, M. Ma, M. Bennis, F.-C. Zheng, and X. You, “User preference learning-based edge caching for fog radio access network,” IEEE Trans. Commun., vol. 67, no. 2, pp. 1268–1283, Feb. 2019.

[11] X. Liu, M. Derakhshani, and S. Lamborghini, “Contextual learning for content caching with unknown time-varying popularity profiles via incremental clustering,” IEEE Trans. Commun., vol. 69, no. 5, pp. 3011–3024, May 2021.

[12] V. Reddy G et al., “DWP-WP: Dynamic feature weighting based popularity prediction for social media content,” 2021, arXiv:2110.08510.

[13] Y. Jiang, H. Feng, F.-C. Zheng, D. Niyato, and X. You, “Deep learning-based edge caching in fog radio access networks,” IEEE Trans. Wireless Commun., vol. 19, no. 12, pp. 8442–8454, Dec. 2020.

[14] H. Feng, Y. Jiang, D. Niyato, F.-C. Zheng, and X. You, “Content popularity prediction via deep learning in cache-enabled fog radio access networks,” in Proc. IEEE Global Commun. Conf. (GLOBECOM), Dec. 2019, pp. 1–6.

[15] Y. Jiang, Y. Wu, F.-C. Zheng, M. Bennis, and X. You, “Federated learning-based content popularity prediction in fog radio access networks,” IEEE Trans. Wireless Commun., vol. 21, no. 6, pp. 3836–3849, Jun. 2022.

[16] Y. Wu, Y. Jiang, M. Bennis, F. Zheng, X. Gao, and X. You, “Content popularity prediction in fog radio access networks: A federated learning based approach,” in Proc. IEEE Int. Conf. Commun. (ICC), Jun. 2020, pp. 1–6.

[17] S. Mehrzai, A. Tskakalidis, S. Chatzinotas, and B. Ottersten, “A feature-based Bayesian method for content popularity prediction in edge-caching networks,” in Proc. IEEE Wireless Commun. Netw. Conf. (WCNC), May 2019, pp. 1–6.
