A Feature-Based Bayesian Method for Content Popularity Prediction in Edge-Caching Networks

Sajad Mehrizi, Anestis Tsakmalis, Symeon Chatzinotas, Björn Ottersten
Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg
$\{sajad.mehrizi, anestis.tsakmalis, symeon.chatzinotas, bjorn.ottersten\}$@$uni.lu$

***Abstract*—Edge-caching is recognized as an efficient technique for future wireless cellular networks to improve network capacity and user-perceived quality of experience. Due to the random content requests and the limited cache memory, designing an efficient caching policy is a challenge. To enhance the performance of caching systems, an accurate content request prediction algorithm is essential. Here, we introduce a flexible model, a Poisson regressor based on a Gaussian process, for the content request distribution in stationary environments. Our proposed model can incorporate the content features as side information for prediction enhancement. In order to learn the model parameters, which yield the Poisson rates or alternatively content *popularities*, we invoke the Bayesian approach which is very robust against over-fitting. However, the posterior distribution in the Bayes formula is analytically intractable to compute. To tackle this issue, we apply a Monte Carlo Markov Chain (MCMC) method to approximate the posterior distribution. Two types of predictive distributions are formulated for the requests of existing contents and for the requests of a newly-added content. Finally, simulation results are provided to confirm the accuracy of the developed content popularity learning approach.**

***Index Terms*—Popularity prediction, Stationary environment, Content features, Poisson distribution, Gaussian process, Bayesian Learning**

I. Introduction

Mobile data traffic is forecast to increase at a 47% compound annual growth rate (CAGR) from 2016 to 2021, two times faster than the growth of global IP fixed traffic during the same period. [1]. This is largely due to the growth in both the number of mobile devices and the user interest towards high-rate multimedia applications. Nevertheless, supporting such a huge data traffic turns to be a big challenge which indicates the need for developing new architectures. To mitigate this issue, edge-caching is recognized as one of the leading technologies [2], [3]. It can bring the requested content from the core network close to the end mobile user, instead of downloading the same content multiple times through the backhaul links. Therefore, by serving the mobile users locally, edge-caching can jointly offload traffic burden on the backhaul links, reduce system costs and improve quality of service (QoS) of the mobile users.

Over the past few years, extensive research has been carried out on edge-caching networks, which has mainly focused on the performance analysis of caching, cache placement optimization and transmission strategies. A cache placement algorithm has been proposed to minimize the excepted downloading time for contents in [2]. In [4], physical layer features are used in the cache placement problem to minimize network cost while to satisfy users' QoS requirements. The authors in [5] investigated energy efficiency and time delivery of an edge-caching network. In addition, various coding schemes, intra and inter sessions, have been proposed to enhance caching performance [2], [6], [7].

The main assumption of the aforementioned papers is that the content popularity is known in advance. However, in practice, the popularity is unknown and has to be estimated and predicted. In this respect, the popularity learning problem can be categorized in two general approaches: model-free and model-based. In the model-free approach, there is no assumption on the content request distribution. The popularity learning is then performed within the process of optimizing a reward function (e.g cache hit ratio) by the so-called exploration-exploitation procedure. Multi-armed-bandit (MAB) and reinforcement learning algorithms are mostly based on this approach which also have been adapted to edge-caching applications [8]–[11]. On the other hand, in the model-based approach, it is assumed that the content requests are generated by a parametric distribution. The Poisson stochastic process is a popular model adopted in the content delivery networks [12] and also has been used in edge-caching [13]. Once the request is modeled, the next step is to estimate the popularity. A simple way is to take the average of instantaneous requests, which is equivalent to the maximum likelihood estimation (MLE) from the estimation theory perspective. However, the MLE suffers from overfitting especially in edge caching systems where only a few request observations are available. For example, as it is reported in [14], a base station cache typically may receive 0.1 requests/content/day which is too small in contrast with a typical content delivery network cache which normally receives 50 requests/content/day.

To improve the popularity estimation accuracy, side information (user profile and content features) can also be incorporated in learning algorithms. In [13], [15], user profiles are leveraged to speed up the learning convergence rate. One important issue with this kind of side information is that users may not be willing to share their personal profiles to the edge-cache entity. On the other hand, content features (e.g topic categories) can be easily and cheaply obtained from the content server without jeopardizing users' privacy. In addition, knowing the most important content features can be useful to design advanced cache-placement algorithms. For example, the authors of [16] observed that there is a traffic pattern under different topic categories of contents by doing experimental validation on the dataset of a real mobile network. Therefore, besides learning the popularities of the contents, in order to have a better understanding about the hidden request pattern, it is advantageous to also learn the importance of content features.

In this paper, we take the content features into account and introduce a new probabilistic model for the content requests. The learning process is performed in the Bayesian paradigm which is robust against overfitting and provides a way to quantify our uncertainty about the estimation. The model allows us to define different types of predictive distributions by which we can effectively model the uncertainty of future requests. The statistical information of these posterior predictive distributions can be used to design a sophisticated caching policy. Here, we should also mention that the central contribution of this paper is not to devise a caching policy but rather to propose a more accurate and reasonable probabilistic model for content requests. Overall, the main contributions of the paper are summarized as:

- We provide a probabilistic model, a Poisson regressor based on a Gaussian process, for stationary content requests which captures the similarity between contents. The Gaussian process is a very flexible and powerful statistical model that can model nonlinear relationships between the popularities and the features.
- The parameters of the model are learnt in the Bayesian framework. Due to few request samples in the local cache, Bayesian learning provides a powerful framework to mitigate overfitting.
- For prediction, two types of predictive distributions are specified. One is used to predict the future requests for the existing contents and the other to predict the popularity of a new content that may come to the system.

The rest of the paper is organized as follows: the system model and problem statement are described in Section II. In Section III, we apply the Bayesian approach for popularity learning. Finally, Section IV shows the simulation results and Section V concludes the paper.

II. System Model and problem formulation

In this paper, we consider a cellular network consisting of a base station (BS) serving its mobile users. Users can make random requests from a library of contents $\mathcal{C} = \{c_1, ..., c_M\}$, where M is the total number of contents. Each content is assumed to have a set of features. For instance, a video content may have a specific topic (e.g education, entertainment, science-technology,..) and some other features such as release year. We use $\mathbf{x}_m$ to be the feature vector of content c_m with Q dimensions whose values can be either binary or continuous.

The BS is equipped with a limited capacity cache memory, and is connected to the remote content server through the backhaul links. Additionally, the remote server has access to the whole content library $\mathcal{C}$. At each time slot[1], each user independently requests a content (or contents)[2] from the library $\mathcal{C}$. To alleviate the traffic burden on the backhaul links and increase the users' QoS, some contents are stored in the cache depending on the caching policy. The requested contents by the users will be served directly if they are already cached; otherwise they are fetched from the content server. We suppose that the cache module of the BS can only monitor the number of user requests towards contents of the library and cannot perform any user profiling. In addition, it is assumed that the content popularity is fixed (we can assume it does not change over short time intervals, e.g. a few days) and the requests are samples generated from a stationary distribution.

We define $\mathbf{d}_c\left[T_n\right] = \left[d_{c_1}\left[T_n\right], ..., d_{c_M}\left[T_n\right]\right]^T$ to be the request vector where $d_{c_m}\left[T_n\right]$ is the total number of requests for content m during time slot n with duration T_n. For simplicity, we assume that $T_n = T_{n'}, \forall n' \neq n$. Therefore, we can drop T and show the request vector by $\mathbf{d}_{c,n} = \left[d_{c_1,n}, ..., d_{c_M,n}\right]^T$. Also, the requests for $n' \neq n$ are presumed to be statistically independent random variables. A common parametric model for the requests is the Poison stochastic process and the MLE approach to estimate the rate request, or the popularity (we use the terms rate and popularity interchangeably) [13] as:

$$r_m = \frac{\sum\limits_{n=1}^{N} d_{c_m,n}}{N}, \quad \forall m = 1, ..., M \tag{1}$$

where r_m is the popularity of content c_m and N is the total number of request observations during the training period. Although this approach is simple, it is not very accurate for popularity estimation. Firstly, MLE suffers from severe overfitting especially when the training set has only a few request observations. Secondly, it cannot incorporate any kind of side information. For example, users commonly request contents based on their features. Therefore, we expect content popularities to be correlated in the feature space. By appropriately using this underlying prior knowledge about requests, the accuracy of popularity estimation can be significantly improved. In the next sections, we present our probabilistic model in order to deal with these issues. Before introducing the model, we summarize the basic concepts of Gaussian processes which are essential for the subsequent sections.

A. Gaussian Process in a Nutshell

A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribu-

[1] The time slots can be hours, days, etc.

[2] There is no limitation on the number of requests by a user at a time slot

tion. Using a Gaussian process, we can define a distribution over functions $f(\mathbf{x})$:

$$f(\mathbf{x}) \sim \mathcal{GP}\left(\mu(\mathbf{x}), K(\mathbf{x}, \mathbf{x}')\right) \tag{2}$$

where $\mathbf{x}$ is an arbitrary input[3] variable with Q dimensions, and the mean function, $\mu(\mathbf{x})$, and the Kernel function, $K(\mathbf{x}, \mathbf{x}')$, are respectively defined as:

$$\mu(\mathbf{x}) = E\left[f(\mathbf{x})\right] \tag{3}$$

$$K(\mathbf{x}, \mathbf{x}') = E\left[\left(f(\mathbf{x}) - \mu(\mathbf{x})\right)\left(f(\mathbf{x}') - \mu(\mathbf{x}')\right)\right]. \tag{4}$$

This means that any finite collection of function values has a joint Gaussian distribution:

$$\left[f(\mathbf{x}_1), ..., f(\mathbf{x}_M)\right]^T \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}) \tag{5}$$

where $\boldsymbol{\mu} = \left[\mu(\mathbf{x}_1), ..., \mu(\mathbf{x}_M)\right]^T$ and the covariance matrix $\mathbf{K}$ has the entities $[\mathbf{K}]_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$. The kernel function specifies the main characteristics of the function that we wish to model and the basic assumption is that variables with inputs $\mathbf{x}$ which are close are likely to be correlated. Choosing a good kernel function for a learning task depends on intuition and experience. A popular and simple kernel is the squared exponential kernel (SEK):

$$K(\mathbf{x}_i, \mathbf{x}_j) = \theta_1 e^{-\sum\limits_{q=2}^{Q+1} \theta_q \left\| x_i^{(q-1)} - x_j^{(q-1)} \right\|^2} \tag{6}$$

where θ_1 is the vertical scale variation and θ_{q+1} is the horizontal scale variation on dimension q of the function. By using different scales for each input dimension, we let them to have different importance. If θ_{q+1} is close to zero, dimension q will have little influence on the covariance of variables. Covariance function (6) is infinitely differentiable and is thus very smooth. More details about the Gaussian process and the kernel functions can be found in [17].

B. The proposed model

In this subsection, we introduce our probabilistic model for content requests. The following regression-based hierarchical (multilevel) probabilistic model is proposed:

$$d_{c_m,n} | \lambda_m(\mathbf{x}_m) \sim Poi\left(e^{\lambda_m(\mathbf{x}_m)}\right), \forall n = 1, ..., N \tag{7a}$$

$$\lambda_m(\mathbf{x}_m) | f(\mathbf{x}_m), \theta_0 \sim \mathcal{N}\left(f(\mathbf{x}_m), \theta_0\right) \tag{7b}$$

$$f(\mathbf{x}) | \mathbf{x}, \theta_1, ..., \theta_{Q+1} \sim \mathcal{GP}\left(0, K(\mathbf{x}, \mathbf{x}')\right). \tag{7c}$$

The first level of the model, (7a), is the Poisson observation distribution for content requests. At this level, the request for content c_m is assumed to follow a Poisson distribution with natural parameter $\lambda_m(\mathbf{x}_m)$ which is a function of its features. We note that the request rate is an exponential function of the natural parameter, $r_m(\mathbf{x}_m) = e^{\lambda_m(\mathbf{x}_m)}$. As we previously mentioned, it is expected that there is a similar request pattern between contents with similar features. This prior information is employed at the higher levels. In (7b), $\lambda_m(\mathbf{x}_m)$ follows a normal distribution with mean $f(\mathbf{x}_m)$ and variance θ_0. By this assumption, we allow contents with exactly the same features to have different popularities which is possible in practice. At the higher level of the model, (7c), we assume that $\{f(\mathbf{x}_m)\}_{m=1}^M$ are realizations of function $f(\mathbf{x})$ drawn from a Gaussian process with zero mean and kernel function K. By this assumption, contents with similar features are encouraged to be correlated in the feature space.

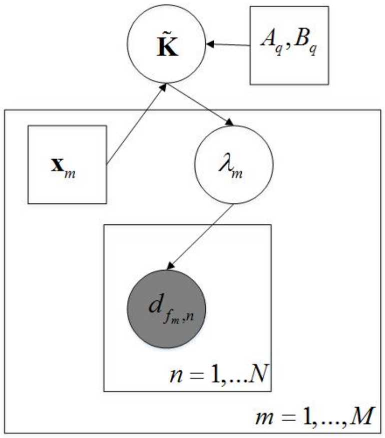

Fig. 1: The proposed probabilistic model for content requests

III. Bayesian Learning

A. Inference

In this section, we exploit the Bayesian framework to learn the probabilistic model in (7). In other words, given the content request observations $\mathcal{D} = \{\mathbf{d}_{c,n}, ..., \mathbf{d}_{c,n}\}_{n=1}^N$, we aim to update our belief about the model's parameters $\{\lambda_m(\mathbf{x}_m)\}_{m=1}^M, f(\mathbf{x})$. However, we cannot estimate the infinite-dimensional function $f(\mathbf{x})$ and hence the focus is only on the realizations at $\{f(\mathbf{x}_m)\}_{m=1}^M$. Moreover, to simplify the inference, we can integrate out $f(\mathbf{x}_m)$ from the model. By doing this, we have:

$$\boldsymbol{\lambda} = \left[\lambda_1(\mathbf{x}_1), ..., \lambda_M(\mathbf{x}_M)\right]^T \sim \mathcal{N}\left(\mathbf{0}, \tilde{\mathbf{K}}\right) \tag{8}$$

where $\tilde{\mathbf{K}} = \mathbf{K} + \theta_0 \mathbf{I}$. Additionally, in practice, the available prior knowledge may not be enough to fix the parameters $\{\theta_q\}_{q=0}^{Q+1}$. A common approach to estimate these parameters is cross validation. However, this trial and error experiment may be tedious and computationally extensive. A very systematic way to learn these parameters is to model their uncertainty by a prior distribution. Since the values of $\theta_0, .., \theta_Q$ must be positive, a natural choice would be Gamma priors:

$$\theta_q \sim Gam(A_q, B_q) \quad \forall q = 0, ..., Q+1 \tag{9}$$

where A_q and B_q are respectively the shape and the rate of each Gamma distribution.

Fig.1 shows the graphical representation of the Bayesian model. The shaded node represents the observed requests

[3] Here, input is a very general concept that can be time, location, ... In our problem, it is content features.

and the plates represent multiple samples of random variables. The unshaded circle nodes indicate unknown quantities and the squares show the deterministic parameters of the model.

The inference of all unknown variables of the model is given by the Bayes rule as:

$$p\left(\boldsymbol{\lambda}, \{\theta_q\}_{q=0}^{Q+1} | \mathcal{D}\right) = \frac{\prod_{n=1}^{N} \prod_{m=1}^{M} p\left(d_{c_m,n} | \lambda_m\right) p\left(\boldsymbol{\lambda} | \tilde{\mathbf{K}}\right) \prod_{q=0}^{Q+1} p\left(\theta_q\right)}{Z} \tag{10}$$

where $p\left(\boldsymbol{\lambda}, \{\theta_q\}_{q=0}^{Q+1} | \mathcal{D}\right)$ is the posterior distribution and the denominator Z is a normalization constant. Unfortunately, the normalization constant is intractable to compute and there is no closed-form expression for the posterior distribution. So, instead, we use a Monte Carlo Markov Chain (MCMC) method to approximate the posterior distribution. Specifically, we use the Hamiltonian Monte Carlo (HMC) method which has been one of the most successful MCMC methods to sample from an unnormalized distribution. Now, we give an overview of the HMC. The complete description can be found in [18].

HMC is based on the simulation of Hamiltonian dynamics as a method to generate a sequence of samples $\{\boldsymbol{\zeta}_s\}_{s=1}^{S}$ from a desired D-variate distribution $p(\boldsymbol{\zeta})$ by exploring its sample space. It combines gradient information of $p(\boldsymbol{\zeta})$ and auxiliary variables, $\mathbf{p} \in R^{D \times 1}$, with density $p(\mathbf{p}) = \mathcal{N}(\mathbf{0}, \mathbf{G})$. The Hamiltonian function is then defined as:

$$H(\boldsymbol{\zeta}, \mathbf{p}) = \psi(\boldsymbol{\zeta}) + \frac{1}{2}\log(2\pi)^D \mathbf{G} + \frac{1}{2}\mathbf{p}^T \mathbf{G} \mathbf{p} \tag{11}$$

where $\psi(\boldsymbol{\zeta})$ is the negative log of the unnormalized $p(\boldsymbol{\zeta})$ and $\mathbf{G}$ is usually assumed to be the identity matrix. The physical analogy of (11) is the Hamiltonian dynamics which describe the sum of the potential energy (the first term) and the kinetic energy (the last two terms).

Hamiltonian dynamics are simulated by discretizing their continuous analogue equations using the leapfrog method. This discretization has two parameters, number of leapfrog steps L and step-size ε. The full description of a movement in HMC which is from a current state (sample) to a new state is depicted in Alg.1. HMC is only applicable for differentiable and unconstrained variables. However, in (10), there are some variables, $\{\theta_q\}_{q=0}^{Q+1}$, that must be positive. To handle this issue, we exploit the exponential-transformation where instead of θ_q, we use $\phi_q = \log(\theta_q)$ with ϕ_q serving as an unconstrained auxiliary variable. Note that to use these transformations, we also need to compute the Jacobian determinant as a result of the change of random variables.

By defining $\boldsymbol{\zeta} = \left[\boldsymbol{\lambda}^T, \phi_0, ...\phi_{Q+1}\right]^T \in R^{(M+Q+2)\times 1}$ and $p(\boldsymbol{\zeta})$ as the posterior distribution (10), the negative log of unnormalized $p(\boldsymbol{\zeta})$ (after the exponential-transformation) is given by:

$$\psi(\boldsymbol{\zeta}) = -\log p\left(\boldsymbol{\lambda}, \{\theta_q\}_{q=0}^{Q+1} | \mathcal{D}\right) = \sum_{m=1}^{M} \sum_{n=1}^{N} -d_{c_m n}\lambda_m + e^{\lambda_m}$$
$$+ \frac{1}{2}\log\det\left(\tilde{\mathbf{K}}\right) + \frac{1}{2}\boldsymbol{\lambda}^T \tilde{\mathbf{K}}^{-1}\boldsymbol{\lambda} + \sum_{q=0}^{Q+1} -A_q\phi_q + B_q e^{\phi_q}. \tag{12}$$

Also, the gradient of (12), which is required in Alg.1, can be easily computed by using matrix derivatives [19]:

$$\frac{\psi(\boldsymbol{\zeta})}{\partial\lambda_m} = \sum_{n=1}^{N} -d_{c_m n} + Ne^{\lambda_m} + \left[\tilde{\mathbf{K}}^{-1}\boldsymbol{\lambda}\right]_m$$
$$\frac{\psi(\boldsymbol{\zeta})}{\partial\phi_q} = \frac{1}{2}tr\left(\tilde{\mathbf{K}}^{-1}\frac{\partial\tilde{\mathbf{K}}}{\partial\phi_q}\right) - \frac{1}{2}\boldsymbol{\lambda}^T\tilde{\mathbf{K}}^{-1}\frac{\partial\tilde{\mathbf{K}}}{\partial\phi_q}\tilde{\mathbf{K}}^{-1}\boldsymbol{\lambda} - A_q + B_q e^{\phi_q}.$$

Algorithm 1: The HMC sampling algorithm [18]

```
Input: $\boldsymbol{\zeta}_s, \varepsilon, L, \nabla_{\boldsymbol{\zeta}}\psi(\boldsymbol{\zeta},), \psi(\boldsymbol{\zeta},), \mathbf{G}$
Output: $\boldsymbol{\zeta}_{s+1}$
/* draw a sample from $p(\zeta)$ */
$\mathbf{q}_1 = \boldsymbol{\zeta}_s$, $\mathbf{p}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{G})$;
Compute $H(\boldsymbol{q}_1, \boldsymbol{p}_1)$;
for $l \leftarrow 1$ to $L$ do
    $\mathbf{p} \leftarrow \mathbf{p}_l - \varepsilon\nabla\psi(\boldsymbol{q}_l)$;
    $\boldsymbol{q}_{l+1} = \boldsymbol{q}_l + \varepsilon\mathbf{G}^{-1}\mathbf{p}$;
    $\mathbf{p}_{l+1} = \mathbf{p} - \varepsilon\nabla\psi(\boldsymbol{q}_{l+1})$;
end
compute $dH = H(\boldsymbol{q}_{L+1}, \mathbf{p}_{L+1}) - H(\boldsymbol{q}_1, \mathbf{p}_1)$;
if $rand() < e^{-dH}$ then
    $\boldsymbol{\zeta}_{s+1} = \boldsymbol{q}_{L+1}$;          /* accept */
else
    $\boldsymbol{\zeta}_{s+1} = \boldsymbol{q}_1$;          /* reject */
end
```

Once, we collect enough samples from the HMC, any function of the posterior distribution moments can be computed. The initial MCMC samples are usually discarded because they may be far away from the true distribution. These samples are called burn-in samples.

Nevertheless, our goal is not just to learn the parameters of the model based on the training set but is to make prediction about the possible content request values in future. The next subsection explains how this can be performed.

B. Prediction

Here, we aim to perform prediction in two ways. The first one is to predict the requests for the existing contents. This can be performed using the posterior predictive distribution (distribution of a new request) given in (13):

$$p\left(\mathbf{d}_c^{new} | \mathcal{D}\right) = \int p\left(\mathbf{d}_c^{new} | \boldsymbol{\lambda}\right) p\left(\boldsymbol{\lambda} | \mathcal{D}\right) d\boldsymbol{\lambda} \tag{13}$$

where $p(\mathbf{d}_c^{new} | \boldsymbol{\lambda})$ is a Poisson distribution and $p(\boldsymbol{\lambda} | \mathcal{D})$ is the marginal posterior distribution of $\boldsymbol{\lambda}$. However, we would like to make a point prediction rather than dealing with the whole predictive distribution. The best guess for a point estimation in the Bayesian context is based on risk (or loss) minimization [20, Chapter 2]. In other words, a loss function is defined which specifies the loss incurred by guessing the value $\mathbf{d}^{new}$ when the actual value is $\mathbf{d}^{*}$. The most common loss evaluation metric is the quadratic loss. The value of $\mathbf{d}^{new}$ that minimizes this risk function is the mean of the predictive distribution which can be approximated as:

$$E\left\{\mathbf{d}_c^{new}|\mathcal{D}\right\} \approx \frac{1}{S}\sum_{s=1}^{S} e^{\boldsymbol{\lambda}_s}. \qquad (14)$$

The second prediction task is to predict the popularity of a newly-added content that may enter the system. This can be calculated by a second type of posterior predictive distribution defined as:

$$p\left(\lambda_{M+1}|\mathbf{x}_{M+1}\right) = \int p\left(\lambda_{M+1}|\boldsymbol{\lambda},\boldsymbol{\theta},\mathbf{x}_{M+1}\right) p\left(\boldsymbol{\lambda},\boldsymbol{\theta}|\mathcal{D}\right) d\boldsymbol{\lambda} d\boldsymbol{\theta} \qquad (15)$$

where $\mathbf{x}_{M+1}$ is the feature vector of the new content. To compute $p\left(\lambda_{M+1}|\boldsymbol{\lambda},\mathbf{x}_{M+1}\right)$, we note that the joint distribution of $p\left(\lambda_1,...,\lambda_{M+1}\right)$ is a Normal distribution with zero mean and covariance matrix:

$$\begin{pmatrix} \tilde{\mathbf{K}} & \tilde{\mathbf{k}} \\ \tilde{\mathbf{k}}^T & K\left(\mathbf{x}_{M+1},\mathbf{x}_{M+1}\right)+\theta_0 \end{pmatrix}$$

where $\tilde{\mathbf{k}} = \left[K\left(\mathbf{x}_1,\mathbf{x}_{M+1}\right),...,K\left(\mathbf{x}_M,\mathbf{x}_{M+1}\right)\right]^T$. Based on the properties of Normal distributions, the conditional distribution $p\left(\lambda_{M+1}|\boldsymbol{\lambda},\mathbf{x}_{M+1}\right)$ is a Normal distribution with mean and variance:

$$\hat{\lambda}_{M+1} = \tilde{\mathbf{k}}^T\tilde{\mathbf{K}}^{-1}\boldsymbol{\lambda}$$
$$\hat{\sigma}_{M+1} = K\left(\mathbf{x}_{M+1},\mathbf{x}_{M+1}\right)+\theta_0 - \tilde{\mathbf{k}}^T\tilde{\mathbf{K}}^{-1}\tilde{\mathbf{k}}.$$

Again, the optimal predictive value for (15) considering the quadratic loss is its mean. It should be noted that (15) is the distribution of the natural parameter of a new content. The point estimation of the request rate can be approximated as:

$$E\left(r_{M+1}|\mathbf{x}_{M+1}\right) \approx \frac{1}{S}\sum_{s=1}^{S} e^{\tilde{\mathbf{k}}_s^T(\mathbf{x}_{M+1},\mathbf{x})\tilde{\mathbf{K}}_s^{-1}\boldsymbol{\lambda}_s}. \qquad (16)$$

IV. Simulation Results

In this section, we present our simulation results to show the performance of the proposed probabilistic content request model denoted by "*Bayesian Poisson-GP*". To compare our results, we use the independent Poisson model with MLE in (1) denoted by "*MLE Poisson*" as a benchmark. As far as the HMC technique is concerned, we set $\varepsilon = .015$ and $L = 20$ and ran it for 5000 samples where the first 2500 samples were considered as the burn-in samples. The number of features is $Q = 4$ and specifically features $x_m^{(1)}$, $x_m^{(2)}$, $x_m^{(3)}$ are binary whose values are randomly generated from Bernoulli distributions with parameters 0.5, 0.8 and 0.2 for all m, respectively. Feature $x_m^{(4)}$ is continuous and generated from a Normal distribution with zero mean and unit variance for all m. Moreover, the parameters of the Kernel function (6) are $\theta_0 = .0001, \theta_1 = 0.1, \theta_2 = 0.25, \theta_3 = 0, \theta_4 = 0.1$ and $\theta_5 = 0.5$.

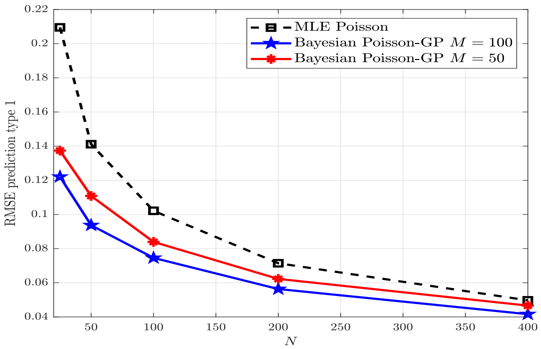

Fig. 2: RMSE prediction type 1 versus N request observations

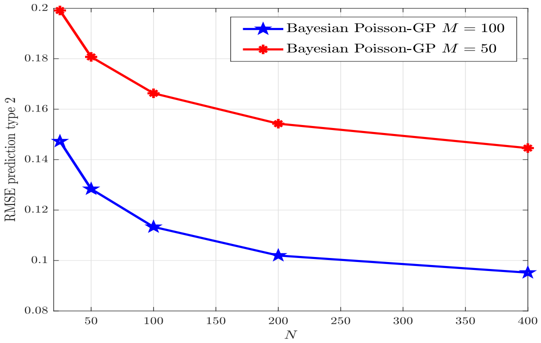

Fig. 3: RMSE prediction type 2 versus N request observations

Fig.2 shows the root mean square error (RMSE) of the popularity predictive type 1 in (14) versus the number of observations in the training set, N. It can be seen that the Bayesian Poisson-GP significantly performs better than the MLE Poisson. We can also observe that as the number of contents increases, the Bayesian Poisson-GP performance is improved. This is because as M increases, the Gaussian process can learn better the relationship between the popularities and the features.

Now, we investigate the performance of our model in terms of how well it can predict the popularity of a new content (popularity predictive type 2 in (16)). The feature of the new content is randomly generated with the same process as for the existing contents. Fig. 3 shows the RMSE of the predicted popularity of the new content versus N. As we see the performance of the model improves when the size of the training set or the number of contents increase. In this scenario, there is no explicit way to use the content features in the MLE Poisson to make prediction about the popularity of a new content, therefore we are unable to compare our model with it.

Next, we show the accuracy of Kernel parameter learning efficiency of our model. Tables I and II show the estimated mean values of the kernel function parameters. As we expected, it is observed that as the number of observations increases we get closer to the true values. However, from the tables, the accuracy improvement of the parameters is largely affected by the number of contents. For example, for feature $x_m^{(2)}$, which does not affect the outcome of the

	True value	N=25	N=50	N=100	N=200	N=400
θ_0	0.0001	0.0081	0.0032	0.0031	0.0016	0.0009
θ_1	0.1	0.1187	0.1443	0.1268	0.1313	0.1269
θ_2	0.25	0.1633	0.1553	0.1879	0.1848	0.2083
θ_3	0	0.0676	0.0484	0.0383	0.0146	0.0192
θ_4	0.1	0.0755	0.0535	0.0542	0.0871	0.0843
θ_5	0.5	0.3354	0.3441	0.3904	0.4180	0.4495

TABLE I: the value of estimated kernel function parameters for $M = 50$

	True value	N=25	N=50	N=100	N=200	N=400
θ_0	0.0001	0.0035	0.0014	0.0010	0.0006	0.0002
θ_1	0.1	0.1179	0.1225	0.1141	0.1117	0.1129
θ_2	0.25	0.2187	0.2296	0.2232	0.2451	0.2428
θ_3	0	0.0466	0.0179	0.0072	0.0077	0.0045
θ_4	0.1	0.0736	0.0762	0.0902	0.0969	0.1043
θ_5	0.5	0.3732	0.4504	0.4649	0.4536	0.4753

TABLE II: the value of estimated kernel function parameters for $M = 100$

model, the value of its scale variation, θ_3, has a better estimation at $N = 400$ for $M = 100$ in comparison with $M = 50$. These results confirm our previous simulations that as M increases the Gaussian process gets more accurate and consequently shows a better prediction performance. The reason for this behavior is that by increasing M the number of observations in the feature space increases which results in a better prediction accuracy.

V. Conclusions

In this paper, we proposed a flexible model for modeling the content requests and predicting their popularity. We proposed a multilevel probabilistic model, the Poisson regressor based on Gaussian process, that can capture the similarity between contents in terms of their features. We utilized Bayesian learning to obtain the parameters of the model because it is robust against overfitting and therefore efficient in edge-caching system where overfitting is a big challenge due to small number of request observations. Then, two posterior predictive distributions were specified for prediction purposes. In the simulation results, we showed that the Bayesian Poisson-Gaussian process structure significantly outperforms the MLE independent Poisson in terms of content popularity prediction.

Acknowledgment

This work was funded by the National Research Fund (FNR), Luxembourg under the projects ”LISTEN” and ”PROCAST”. This work was also supported by the European Research Council (ERC) under the project ”AGNOSTIC”.

References

[1] C. V. N. Index, “Global mobile data traffic forecast update, 2016–2021 white paper, accessed on may 2, 2017.”

[2] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, “Femtocaching: Wireless content delivery through distributed caching helpers,” *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 8402–8413, 2013.

[3] X. Ge, H. Cheng, M. Guizani, and T. Han, “5G wireless backhaul networks: challenges and research advances,” *IEEE Network*, vol. 28, no. 6, pp. 6–11, 2014.

[4] X. Peng, J.-C. Shen, J. Zhang, and K. B. Letaief, “Joint data assignment and beamforming for backhaul limited caching networks,” in *Personal, Indoor, and Mobile Radio Communication (PIMRC), 2014 IEEE 25th Annual International Symposium on*. IEEE, 2014, pp. 1370–1374.

[5] T. X. Vu, S. Chatzinotas, and B. Ottersten, “Edge-caching wireless networks: Performance analysis and optimization,” *IEEE Transactions on Wireless Communications*, 2018.

[6] M. A. Maddah-Ali and U. Niesen, “Fundamental limits of caching,” *IEEE Transactions on Information Theory*, vol. 60, no. 5, pp. 2856–2867, 2014.

[7] W. Han, A. Liu, and V. K. Lau, “PHY-caching in 5G wireless networks: Design and analysis,” *IEEE Communications Magazine*, vol. 54, no. 8, pp. 30–36, 2016.

[8] S. Müller, O. Atan, M. van der Schaar, and A. Klein, “Context-aware proactive content caching with service differentiation in wireless networks,” *IEEE Transactions on Wireless Communications*, vol. 16, no. 2, pp. 1024–1036, 2017.

[9] J. Song, M. Sheng, T. Q. Quek, C. Xu, and X. Wang, “Learning-based content caching and sharing for wireless networks,” *IEEE Transactions on Communications*, vol. 65, no. 10, pp. 4309–4324, 2017.

[10] A. Sadeghi, F. Sheikholeslami, and G. B. Giannakis, “Optimal and scalable caching for 5g using reinforcement learning of space-time popularities,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 180–190, 2018.

[11] S. O. Somuyiwa, A. György, and D. Gündüz, “A reinforcement-learning approach to proactive caching in wireless networks,” *IEEE Journal on Selected Areas in Communications*, 2018.

[12] M. Garetto, E. Leonardi, and V. Martina, “A unified approach to the performance analysis of caching systems,” *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, vol. 1, no. 3, p. 12, 2016.

[13] B. Bharath, K. Nagananda, and H. V. Poor, “A learning-based approach to caching in heterogenous small cell networks,” *IEEE Transactions on Communications*, vol. 64, no. 4, pp. 1674–1686, 2016.

[14] G. Paschos, E. Bastug, I. Land, G. Caire, and M. Debbah, “Wireless caching: Technical misconceptions and business barriers,” *IEEE Communications Magazine*, vol. 54, no. 8, pp. 16–22, 2016.

[15] E. Baştuğ, M. Bennis, and M. Debbah, “A transfer learning approach for cache-enabled wireless networks,” in *Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), 2015 13th International Symposium on*. IEEE, 2015, pp. 161–166.

[16] M. Zeng, T.-H. Lin, M. Chen, H. Yan, J. Huang, J. Wu, and Y. Li, “Temporal-spatial mobile application usage understanding and popularity prediction for edge caching,” *IEEE Wireless Communications*, vol. 25, no. 3, 2018.

[17] C. E. Rasmussen, “Gaussian processes in machine learning,” in *Advanced lectures on machine learning*. Springer, 2004, pp. 63–71.

[18] R. M. Neal *et al.*, “MCMC using hamiltonian dynamics,” *Handbook of Markov Chain Monte Carlo*, vol. 2, no. 11, 2011.

[19] K. B. Petersen, M. S. Pedersen *et al.*, “The matrix cookbook,” *Technical University of Denmark*, vol. 7, no. 15, p. 510, 2008.

[20] C. Robert, *The Bayesian choice: from decision-theoretic foundations to computational implementation*. Springer Science & Business Media, 2007.