Content Popularity Estimation in Edge-Caching Networks from Bayesian Inference Perspective

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*—The efficiency of cache-placement algorithms in edge-caching networks depends on the accuracy of the content request's statistical model and the estimation method based on the postulated model. This paper studies these two important issues. First, we introduce a new model for content requests in stationary environments. The common approach to model the requests is through the Poisson stochastic process. However, the Poisson stochastic process is not a very flexible model since it cannot capture the correlations between contents. To resolve this limitation, we instead introduce the Poisson Factor Analysis (PFA) model for this purpose. In PFA, the correlations are modeled through additional random variables embedded in a low dimensional latent space. The correlations provide rich information about the underlying statistical properties of content requests which can be used for advanced cache-placement algorithms. Secondly, to learn the model, we use Bayesian Learning, an efficient framework which does not overfit. This is crucial in edge-caching systems since only partial view of the entire request set is available at the local cache and the learning method should be able to estimate the content popularities without overfitting. In the simulation results, we compare the performance of our approach with the existing popularity estimation method.**

***Index Terms*—Cache-placement, Stationary environment, Poisson Factor Analysis, Bayesian Learning**

I. Introduction

The growth in mobile data traffic over past years is forecast to continue excessively, reaching 47 percent from 2016 to 2021 [1]. This is mainly due to the increase in both the number of devices and the demands for high-rate multimedia applications. The traditional network architectures cannot accommodate such rapid data traffic growth which indicates the need to develop new architectures. One of the most promising approaches to solve the problem is edge-caching which brings popular contents close to the end users [2], [3]. Since an important portion of data traffic in due to only a small number of popular contents [2], caching these contents can significantly reduce the traffic load on the backhaul links and improve the users' quality of service (QoS). Edge-caching is also a promising technology to densify data traffic in Ultra-Dense Networks (UDNs) [4].

In the last few years, extensive research has been conducted on edge-caching networks with its majority focusing on studying performance gain of content placement and transmission strategies. An optimization problem for content placement over multiple caches to reduce latency has been proposed in [2]. In [5], physical layer features are considered in the content placement problem to minimize network cost subject to user's QoS requirements. Energy efficiency and time delivery of edge-caching network have been analyzed and optimized in [6]. Different coding strategies, intra and inter sessions, have been proposed to improve caching performance [2], [7], [8].

The main assumption of the aforementioned papers is that the content popularity is known. However, in practice, the popularity is unknown and has to be estimated. Several works have investigated this issue. In [9], [10], the authors designed a multi-armed-bandit (MAB) algorithm to estimate the popularities. In [11], the content requests are modeled by Poisson distributions and the popularities are estimated by the Maximum Likelihood (ML) approach. They also studied the required training time for obtaining a good estimate of the popularities.

A fundamental challenge in content popularity estimation in edge-caching networks is that a comparatively small number of users request contents [12]. For example, as it is reported in the paper, a base station cache typically may receive 0.1 requests/content/day. This together with the small size of cells makes the task of content popularity estimation challenging. Therefore, the issue is how to efficiently estimate the content popularities from a very small number of request samples (incomplete observation) to achieve a satisfactory cache hit ratio. To deal with this problem, transfer-learning (TL) algorithms are proposed in [11], [13] where social network knowledge has been used as side-information to enhance the popularity estimation accuracy.

However, there is one important issue that to the best of our knowledge has been ignored in edge-caching literature. All previous works assumed contents are uncorrelated. This assumption is too unrealistic and in fact contents can be highly correlated. Some of them may contain the same features, for example in movie files, they may have the same genre. As a result of this correlation, their requests follow a similar pattern in temporal domain. Modeling the correlation among contents can improve the accuracy of popularity estimation and provide rich information about the underlying requests' pattern which can also be exploited in the cache-placement optimization problem.

In this paper, we study the content popularity estimation problem in edge-caching wireless networks. The main contributions of the paper include:

- We provide a probabilistic model, the PFA, for stationary content requests which captures rich correlations between

contents. Our proposed model is general and can be applied for any popularity law. In addition, PFA is an efficient method for dimensionality reduction which allows us to model the requests of a large number of correlated contents.
- We learn the parameters of the PFA in a Bayesian manner. Due to few request samples in the local caches, Bayesian learning provides a powerful framework to avoid overfitting.

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

II. System Model and problem formulation

Consider a cellular network consisting of a base station (BS) serving U mobile users. The BS is equipped with a finite cache memory of size C, and is connected to the content server containing M files in total. At each time slot[1], each user independently requests a content (or contents)[2] from the set $\mathcal{F} = \{f_1, ..., f_M\}$. To alleviate the traffic burden on backhaul links, the most popular contents are cached during off-peak periods (e.g. nights). The requested content will be served directly depending on the availability of the content in the local cache. Moreover, we assume that these requests are generated from a stationary distribution whose statistical properties do not change over time (we can assume it is stationary over short time intervals, e.g. a few days).

We define $\mathbf{d}_f[T_n] = [d_{f_1}[T_n], ..., d_{f_M}[T_n]]$ to be the request vector where $d_{f_m}[T_n]$ is the total number of requests for file m during time slot n with duration T_n. For simplicity, we assume that $T_n = T_{n'}$. Therefore, we can drop T and show the request vector by $\mathbf{d}_{f,n} = [d_{f_1,n}, ..., d_{f_M,n}]$. Also, we assume for $n \neq n'$ the requests are statistically independent random variables. In the literature, the most common approach to model requests is based on the Poison stochastic process and the ML approach to estimate the means (popularities) [11] as:

$$r_m = \frac{\sum_{n=1}^{N} d_{f_m,n}}{N}, \quad \forall m = 1, ..., M \tag{1}$$

where N in the total number of observations during the training period. This approach has two important defects. Firstly, it cannot model the correlations between contents. Secondly, ML suffers from severe overfitting especially when the training set has a few samples. In the next sections, we present our approach to deal with these issues.

III. Probabilistic model for content requests

In order to capture the correlation between contents, a multivariate distribution whose support is the natural number set is needed. The Poisson-Normal (PN) distribution is a mixture distribution that was proposed in [14] for count data to capture their correlations. In the PN distribution, it is assumed that the natural parameter of Poisson data is a random variable following a normal distribution. In this context, the request generation process based on the PN model is given by:

$$\begin{aligned} &\mathbf{y}_n \sim \mathcal{N}(\mathbf{m}, \mathbf{\Omega}), \quad \forall n = 1, ..., N \\ &d_{f_m n} \sim Poi(y_{nm}), \forall n = 1, ..., N, \forall m = 1, ..., M \end{aligned} \tag{2}$$

where $\mathcal{N}(\mathbf{m}, \mathbf{\Omega})$ indicates an M-dimensional multivariate Normal (MVN) distribution with mean $\mathbf{m}$ and covariance matrix $\mathbf{\Omega}$ and $Poi(y_{nm})$ represents a Poisson distribution with natural parameter y_{nm}. Note that the rate (mean) of Poisson is obtained by exponentiating its natural parameter. In model (2), the correlation among contents is captured by a Normal random variable $\mathbf{y}_n$ which is also known as latent variable.

However, modeling requests by the PN distribution in high dimensions (when the number of contents is too large) can be quite challenging. This issue can be remedied by the exponential family factor analysis framework [15]. The assumption made is that the natural parameters of the exponential family distribution of the data can be described by linear combination of a lower dimensional, i.e. K, latent random variables. By adapting exponential family factor analysis to Poisson data, we obtain the PFA as a special case, where the request vector at time slot n is modeled as:

$$d_{f_m,n} \sim Poi\left(\mathbf{w}_m^T \mathbf{x}_n\right), \forall m = 1, ..., M \tag{3}$$

where $\mathbf{w}_m \in R^{M \times 1}$ is a fixed parameter and it is called a factor loading vector and $\mathbf{x}_n \in R^{K \times 1}$ is a latent random variable and it is assumed to follow an MVN distribution:

$$\mathbf{x}_n \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma}). \tag{4}$$

Note that PFA can be considered as a means of reducing the dimensionality of the PN distribution. For simplicity, we can also assume that latent variables $\mathbf{x}_n$ are uncorrelated. Therefore, $\mathbf{\Sigma}$ is a diagonal matrix with elements σ_k^2. Modeling the latent variables like this allows us to capture both positive and negative correlations. The mean and the correlation can be easily derived as:

$$\begin{aligned} &r_m = E\{d_{f_m}\} = e^{[\mathbf{W}\boldsymbol{\mu}]_m + \frac{1}{2}\left[\mathbf{W}\mathbf{\Sigma}\mathbf{W}^T\right]_{mm}} \\ &Cov\left(d_{f_m}, d_{f_{m'}}\right) = r_m r_{m'}\left[e^{\left[\mathbf{W}\mathbf{\Sigma}\mathbf{W}^T\right]_{mm'}} - 1\right] \end{aligned} \tag{5}$$

where $\mathbf{W} = \left[\mathbf{w}_1^T, ..., \mathbf{w}_M^T\right]$ and it is called the factor loading matrix. Eq. (3) and (4) build a probabilistic model for request generation. Now, we aim to learn this model which is described in the next section.

IV. Bayesian Learning

A common method to estimate the parameters of a probabilistic model is based on ML. However, as we mentioned, ML overfits and has a poor performance. In this section, we use Bayesian techniques to estimate the parameters of PFA.

[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

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

$$p\left(\mathbf{W},\boldsymbol{\mu},\boldsymbol{\Sigma},\{\mathbf{x}_n\}_{n=1}^N|\mathcal{D}\right)= \frac{p\left(\mathcal{D}|\mathbf{W},\{\mathbf{x}_n\}_{n=1}^N\right)p\left(\{\mathbf{x}_n\}_{n=1}^N|\boldsymbol{\mu},\boldsymbol{\Sigma}\right)p\left(\mathbf{W}\right)p\left(\boldsymbol{\mu}\right)p\left(\boldsymbol{\Sigma}\right)}{\int p\left(\mathcal{D}|\mathbf{W},\{\mathbf{x}_n\}_{n=1}^N\right)p\left(\{\mathbf{x}_n\}_{n=1}^N|\boldsymbol{\mu},\boldsymbol{\Sigma}\right)p\left(\mathbf{W}\right)p\left(\boldsymbol{\mu}\right)p\left(\boldsymbol{\Sigma}\right)d\boldsymbol{\mu}d\boldsymbol{\Sigma}d\mathbf{x}_n} \tag{6}$$

where $\mathcal{D}=\{\mathbf{d}_{f,n}\}_{n=1}^N$ contains the request observations during N training time slots. $p\left(\mathbf{W}\right)$, $p\left(\boldsymbol{\mu}\right)$ and $p\left(\boldsymbol{\Sigma}\right)$ are prior distributions that summarize the initial knowledge about the parameters, $p\left(\mathcal{D}|\mathbf{W},\{\mathbf{x}_n\}_{n=1}^N\right)$ and $p\left(\{\mathbf{x}_n\}_{n=1}^N|\boldsymbol{\mu},\boldsymbol{\Sigma}\right)$ consist the likelihood function and $p\left(\mathbf{W},\boldsymbol{\mu},\boldsymbol{\Sigma},\{\mathbf{x}_n\}_{n=1}^N|\mathcal{D}\right)$ is the posterior distribution which shows the updated belief about all unknown variables in the model after receiving the observation set. The denominator is a normalization constant.

Common choices for priors of $\boldsymbol{\mu}$ and $\mathbf{W}$ are:

$$\begin{aligned}\boldsymbol{\mu}&\sim\mathcal{N}\left(\boldsymbol{\mu}_0,\boldsymbol{\Sigma}_0\right)\\ \mathbf{w}_m&\sim\mathcal{N}\left(\mathbf{0},\boldsymbol{\Lambda}\right),\forall m=1,...M\end{aligned} \tag{7}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix with elements λ_k. We use Half-Cauchy for the prior of variances σ_k^2. As it is discussed in [16], Half-Cauchy distribution performs well in a training set with small number of samples as a non-informative prior. Therefore, we assume that:

$$\sigma_k^2\sim C^+\left(0,A\right),\forall k=1,...,K \tag{8}$$

where A is the scale parameter of Half-Cauchy distribution. The form of the prior of the factor load matrix is very important and needs special attention. One important issue is the choice of a suitable K which affects the flexibility of the model and the computational complexity of the inference part. As discussed in [17], to specify a proper dimension for the latent space without discrete choice model e.g. cross validation, another layer of prior should be used for the variance of each column of the factor loading matrix. The magnitude of these variances reveal important characteristics about the columns. A small value of λ_k shows that the corresponding column is irrelevant and vice versa. Therefore, the value of λ_k performs as a sparsity promoting term that acts independently on each column. Hence, similar with (8), we use Half-Cauchy distribution with scale parameter B as a prior for the variance of each column,

$$\lambda_k\sim C^+\left(0,B\right),\forall k=1,...,K \tag{9}$$

Half-Cauchy distribution has also good properties near zero which make it suitable as a sparsity encouraging distribution. This issue is investigated in [18].

Fig.1 shows the graphical representation of the Bayesian model. The shaded node represents the observed requests and the plates represent multiple samples of random variables. Additionally, the unshaded circle nodes indicate unknown quantities and the squares show the deterministic parameters of the model.

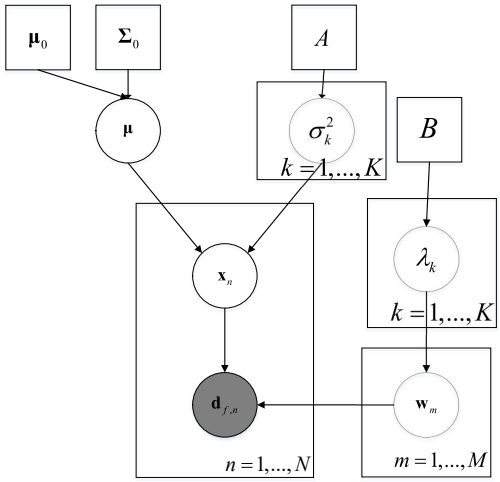

Fig. 1: Graphical representation of our Bayesian model

A. Inference

The complete Bayesian inference problem is given by

$$\begin{aligned}&p\left(\mathbf{W},\{\mathbf{x}_n\}_{n=1}^N,\mu,\boldsymbol{\Sigma},\boldsymbol{\Lambda}|D\right)\\ &=\frac{1}{Z}\prod_{n=1}^N p\left(\mathbf{d}_{fn}|\mathbf{W},\mathbf{x}_n\right)p\left(\mathbf{x}_n|\mu,\boldsymbol{\Sigma}\right)p\left(\mu\right)\\ &\times\prod_{k=1}^K p\left(\sigma_k^2\right)\times\prod_{m=1}^M p\left(\mathbf{w}_m|\boldsymbol{\Lambda}\right)\times\prod_{k=1}^K p\left(\lambda_k\right)\end{aligned} \tag{10}$$

where 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 a brief description of the HMC. The complete description can be found in [19].

HMC is based on the simulation of Hamiltonian dynamics as a method to probe the sample space of a distribution. It combines gradient information of the desired distribution $p\left(\boldsymbol{\theta}\right)$, with $\boldsymbol{\theta}\in R^{D\times1}$ and auxiliary variables, $\mathbf{p}\in R^{D\times1}$, with density $p\left(\mathbf{p}\right)=\mathcal{N}\left(\mathbf{0},\mathbf{G}\right)$. The negative joint log-likelihood is given by:

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

where $\psi\left(\boldsymbol{\theta}\right)$ is the negative log of the unnormalized $p\left(\boldsymbol{\theta}\right)$. $\mathbf{G}$ is a mass matrix and usually it is assumed to be identity. 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, $\mathbf{\Sigma}, \mathbf{\Lambda}$ that must be positive. To handle this issue, we exploit the exponential-transformation where instead of σ_k^2, we use $\tau_k = \log(\sigma_k^2)$ with τ_k serving as an unconstrained auxiliary variable. The same approach is used for λ_k where $\rho_k = \log(\lambda_k)$ is considered. Note that to use these transformations, we also need to compute the determinant Jacobian as a result of the change of random variables.

As a result of some simple mathematical operations, the negative log of the unnormalized posterior distribution is given by:

$$-\log p\left(\mathbf{W}, \{\mathbf{x}_n\}_{n=1}^{N}, \mu, \mathbf{\Sigma}, \mathbf{\Lambda}|\mathcal{D}\right) = \sum_{i=1}^{6} U_i \tag{12}$$

where:

$$U_1 = \sum_{m=1}^{M}\sum_{n=1}^{N} -d_{nm}\left(\mathbf{w}_m^T\mathbf{x}_n\right) + e^{\mathbf{w}_m^T\mathbf{x}_n}$$

$$U_2 = \frac{N}{2}\sum_{k=1}^{K}\tau_k + \sum_{n=1}^{N}\sum_{k=1}^{K}\frac{(x_{nk}-\mu_k)\,e^{-\tau_k}}{2}$$

$$U_3 = -\log p\left(\boldsymbol{\mu}\right) = \frac{\left(\boldsymbol{\mu}-\boldsymbol{\mu}_0\right)\mathbf{\Sigma}_0^{-1}\left(\boldsymbol{\mu}-\boldsymbol{\mu}_0\right)}{2}$$

$$U_4 = \sum_{k=1}^{K} -\tau_k + \log\left(1+\frac{e^{2\tau_k}}{A^2}\right)$$

$$U_5 = \frac{M}{2}\sum_{k=1}^{K}\rho_k + \sum_{m=1}^{M}\sum_{k=1}^{K}\frac{(w_{mk})^2 e^{-\rho_k}}{2}$$

$$U_6 = \sum_{k} -\rho_k + \log\left(1+\frac{e^{2\rho_k}}{B^2}\right)$$

The inputs of HMC are the current sample $\boldsymbol{\theta} = [\mathbf{w}_1, ..., \mathbf{w}_M, \mathbf{x}_1, ..., \mathbf{x}_N, \boldsymbol{\mu}, \tau_1, ..., \tau_K, \rho_1, ..., \rho_K] \in R^{(MK+NK+3K)\times 1}$, the number of leap frog L, the step size ε and the gradient of (12) which can be easily computed. The output is a sample from the unnormalized posterior distribution.

To get independent MCMC samples, the first portion of the generated samples is usually thrown away because they are correlated. This can also be checked by their autocorrelation. After collecting enough of them, we can compute the marginal distributions of all quantities and also any function of them. For example, the mean of content popularities can be calculated as:

$$E\left\{r_m\right\} = \frac{1}{S'}\sum_{s=1}^{S'} e^{[\mathbf{W}_s\boldsymbol{\mu}_s]_m + \frac{1}{2}\left[\mathbf{W}_s\mathbf{\Sigma}_s{\mathbf{W}_s}^T\right]_{m,m}},\ \forall m = 1, ..., M \tag{13}$$

where S' is the effective number of MCMC samples.

Algorithm 1: The HMC sampling algorithm [19]

Input: $\boldsymbol{\theta}_1, \varepsilon, L, \nabla_{\boldsymbol{\theta}}\psi\left(\boldsymbol{\theta},\right), \mathbf{G}$
Output: $\boldsymbol{\theta}$

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

V. Simulation Results

In this section, we present our simulation results to show the performance of the proposed probabilistic content request model. To compare our results, we used the ML independent Poisson approach [11] as a benchmark. Furthermore, we used the MovieLens 10M DataSet [20] which contains 10 million ratings for 10,000 movies by 72,000 users. We assumed that each rating corresponds to one request and also we scrambled the whole data set to satisfy our requirement for stationarity. Each vector request sample corresponds to the number of requests observed in one hour. Additionally, we set $\varepsilon = .001$ and $L = 100$ for the HMC technique and the MCMC was run for 15000 samples where the last 5000 samples were considered as the effective samples. For the dimensions of the latent space, we set $K = 3$ which was found to be sufficient for the simulation parameters.

First, we compare the PFA based model with the ML independent Poisson approach in terms of model accuracy where the log predictive density or the log-likelihood is used as a metric [21] which shows how well a probabilistic model fits to the data. The size of the training set, N, is 10 request vector samples and the movie contents were randomly selected from the data set. Fig.2 represents the log-likelihood values of the Bayesian PFA and the ML independent Poisson models versus the number of movie contents. As we can see, there is a huge gap between the two models especially as the number of contents increases. This shows that the ML independent Poisson is a poor model for modeling the content requests.

In the next part, we compare the performance of the proposed PFA model with respect to the ML independent Poisson case in terms of Cache Hit Ratio (CHR). Here, we randomly chose $M = 25$ movie contents with vector request sample size of 100 from the data set. From these samples, we choose 10 for the training set and the rest for testing. Because the dataset does not provide information about the size of the movies, we randomly and uniformly assign a size for each

movie from the interval $(0, 100)$. As a cache placement rule, we use the traditional CHR maximization policy:

$$\begin{aligned} \max_{\mathbf{x}} \quad & \mathbf{x}^T \mathbf{E}\{\mathbf{r}\} \\ s.t: \quad & \mathbf{x}^T \mathbf{s} \leq C \\ & \mathbf{x} \in \{0,1\}^{M\times 1} \end{aligned} \tag{14}$$

where $\mathbf{x}$ is a binary vector that determines which contents should be cached and $\mathbf{s}$ is a vector which contains the size of the contents. As commonly utilized in the literature, the objective function is the expected value of the CHR and the constraint denotes the cache memory size limit. This optimization problem is combinatorial and computationally intense to solve. However, we can efficiently solve it by relaxing the constraint $\mathbf{x} \in \{0,1\}^{M\times 1}$ with $\mathbf{x} \in [0,1]^{M\times 1}$ which turns the problem into a linear programming and as a result easy to solve. Then, we cache the contents based on the rounded optimized cache policy vector $\mathbf{x}$. The CHR is computed based on the observed requests of the test set. Fig.3 shows the CHR versus different cache size values C. As it can be seen, the proposed model performs better than the ML independent Poisson one for the entire of cache capacity. This is also a strong indicator that the ML independent Poisson model severely overfits.

VI. Conclusions

In this paper, we investigated an alternative model for modeling the content requests and estimating their popularity. We proposed a flexible PFA based model that can capture the correlation between contents. Then, we utilized Bayesian learning to obtain the parameters of the PFA model and as a result the content popularities. Bayesian learning does not overfit and can be considered as an efficient learning method in edge-caching system where overfitting is a big challenge due to small size of request samples. In the simulation results, we showed that the Bayesian PFA structure significantly outperforms the ML independent Poisson one in terms of CHR. Moreover, we assessed the accuracy of the PFA and the ML independent Poisson models in terms of how well they fit to the data set. For this purpose, we used the log predictive density or log-likelihood as a metric and it was observed that the Bayesian PFA model fits significantly better to the data than the ML independent Poisson one.

Acknowledgment

This work was partially supported by the National Research Fund, Luxembourg under the project "LISTEN".

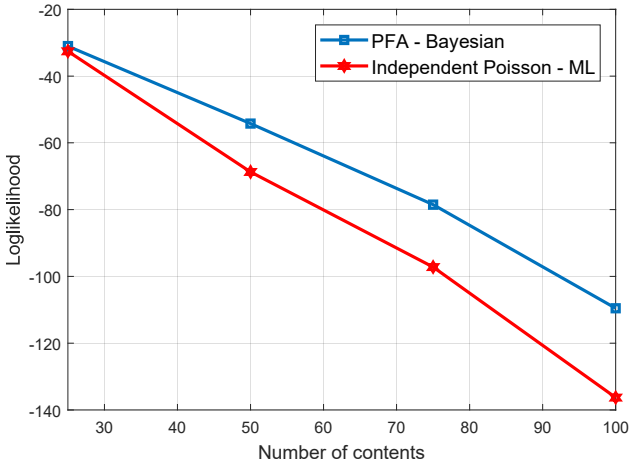

Fig. 2: Loglikelihood versus the number of contents

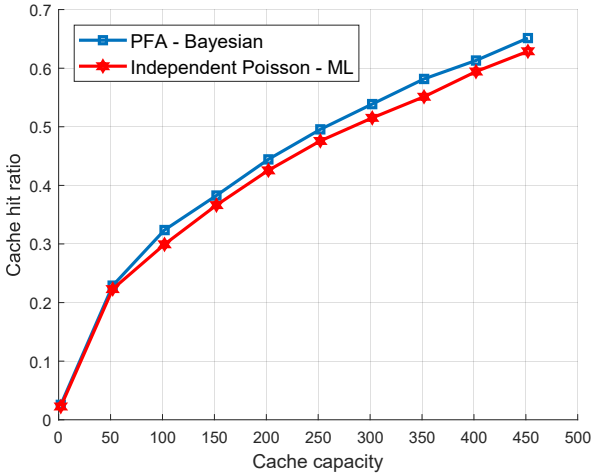

Fig. 3: Cache capacity versus CHR

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] M. Kamel, W. Hamouda, and A. Youssef, "Ultra-dense networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2522–2545, 2016.

[5] 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.

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

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

[8] 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.

[9] 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.

[10] 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.

[11] 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.
[12] 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.
[13] 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.
[14] J. Aitchison and C. Ho, "The multivariate poisson-log normal distribution," *Biometrika*, vol. 76, no. 4, pp. 643–653, 1989.
[15] S. Mohamed, Z. Ghahramani, and K. A. Heller, "Bayesian exponential family PCA," in *Advances in neural information processing systems*, 2009, pp. 1089–1096.
[16] A. Gelman *et al.*, "Prior distributions for variance parameters in hierarchical models (comment on article by browne and draper)," *Bayesian analysis*, vol. 1, no. 3, pp. 515–534, 2006.
[17] C. M. Bishop, "Variational principal components," 1999.
[18] N. G. Polson, J. G. Scott *et al.*, "On the half-Cauchy prior for a global scale parameter," *Bayesian Analysis*, vol. 7, no. 4, pp. 887–902, 2012.
[19] R. M. Neal *et al.*, "MCMC using hamiltonian dynamics," *Handbook of Markov Chain Monte Carlo*, vol. 2, no. 11, 2011.
[20] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 5, no. 4, p. 19, 2016.
[21] A. Gelman, J. Hwang, and A. Vehtari, "Understanding predictive information criteria for bayesian models," *Statistics and computing*, vol. 24, no. 6, pp. 997–1016, 2014.