THE UNIVERSITY of EDINBURGH

Edinburgh Research Explorer

Wireless Edge Caching and Content Popularity Prediction using Machine Learning

Citation for published version:
Krishnendu, S, Bharath, BN, Bhatia, V, Nebhen, J, Dobrovolny, M & Ratnarajah, T 2022, 'Wireless Edge Caching and Content Popularity Prediction using Machine Learning', *IEEE Consumer Electronics Magazine*, pp. 1-9. https://doi.org/10.1109/MCE.2022.3160585

Digital Object Identifier (DOI):
10.1109/MCE.2022.3160585

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
IEEE Consumer Electronics Magazine

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.

OPEN ACCESS

Download date: 11. Aug. 2022

Wireless Edge Caching and Content Popularity Prediction using Machine Learning

S. Krishnendu[1], B. N. Bharath[2], Vimal Bhatia[1,3], Jamel Nebhen[4], and Tharmalingam Ratnarajah[5]

[1]Department of Electrical Engineering, Indian Institute of Technology Indore, India.
[2]Department of Electrical Engineering, Indian Institute of Technology Dharwad, India
[3]Faculty of Information and Management, University of Hradec Kralove.
[4]College of Computer Engineering and Sciences, Prince Sattam bin Abdulaziz University.
[5]Institute for Digital Communications, University of Edinburgh, UK.

***Abstract*—The ever pervasive growth in information services and technology has resulted in the outbreak of demand for data in the wireless networks. This has made the network operators to ponder over the imminent difficulties such as computing capabilities and fronthaul-backhaul link capacities. Hence, to bridge the gap between the cloud capacity and the requirement of mobile services by the network edges, edge computing and caching techniques have been gaining more and more attention from researchers across the world. Further, motivated by the successful applications of machine learning (ML) in solving complex and dynamic problems, in this article, it has been used to advance edge caching capabilities. The proposed ML based algorithms have been evaluated and proved to have better performance compared with the existing conventional algorithms. The highest difference gain in the mean squared error (MSE) for the proposed deep learning (DL) algorithm is observed to be 0.425 and while comparing with the simple neural network, the gain in MSE for the proposed DL algorithm is observed around 27%. Similarly for the federated learning (FL) based caching algorithm the difference is around 10^4, hence demonstrating the benefit of the proposed algorithms. Additionally, opportunities and challenges for a promising upcoming future of ML in edge computing and content popularity prediction has also been discussed.**

I. Introduction

The surge in Internet usage through smart phones, social media platforms and online video streaming has heralded an explosive growth in the amount of data being created. Due to the fast development of communication based applications, it is expected that there will be 5.3 billion total Internet users (66 percent of global population) by 2023, up from 3.9 billion (51 percent of global population) in 2018 [1]. Thus, wireless edge caching which exploits the vast data, compensates for the shortage of local computing capacity and high transmission costs of individual cloud computing. While the unprecedented volume of wireless data traffic may suggest tougher and scalable communication system design, Machine Learning (ML) embraces unique opportunities that can be a game changer for breaking the wireless communication bottlenecks in the next generation 5G and beyond wireless networks [2], [3], [4].

The implementation of dense and Small Base Stations (SBSs) does reduce the latency and provides immense throughput in the 5G and beyond network, but the bottleneck issue still remains the same [5]. To strike a balance between increasing mobile traffic and user's quality-of-experience, edge caching and computing can be seen as invariable solution by bringing storage and computation close to the edge. Both edge caching and computation helps in storing and computation helps in the implementation of ML algorithms at the edge, hence making the edge intelligent and further reducing the burden on the backhaul (see [6], [7], [8], or [9]). The standard simplified algorithms such as least frequently used (LFU), least recently used

(LRU), least frequently/recently used (LRFU) and other variants, can be inefficient when it comes to dynamic environments, since they do not take into account the correlation and non-stationarity of the demand requests. Therefore, a gradual shift towards learning and optimizing the edge devices for content prediction is seen [10], [11]. The dynamic environment makes the modeling approach difficult, and accurate models are difficult to analyze, hence ML based approaches have become very popular.

A critical performance metrics of a caching algorithm include its ability to quickly and accurately learn a popularity distribution of requests. However, a majority of work on performance analysis focuses on cache hit probability after a huge duration has elapsed. Thus the caching problem reduces to how quickly the underlying unknown popularity distribution can be learned. Hence, in this article, the challenges and opportunities in designing scalable wireless network architectures to adapt to the pervasive big data is studied. In this article, first, a state-of-the-art learning based wireless edge caching schemes is reviewed. Building such a wireless network architecture will enable operators to optimize the 5G and beyond wireless networks, where large amount of data can be exploited by using ML tools for content popularity estimation and placement. Second, after an overview, (a) application of reinforcement learning jointly with recommendation in edge caching has been demonstrated, (b) content popularity prediction using Transfer Learning (TL) [12] (c) popularity prediction using Deep Learning (DL) and (d) Federated Learning (FL) based caching strategy is presented [9]. Finally, we present the future directions and challenges to conclude the article.

II. Wireless Caching Scenarios

In this section, wireless caching is classified into different subsections depending on the architecture of the wireless network.

A. *Small Base Station Caching*

Deploying edge servers on SBS allows services to be provided quickly and efficiently to the users. The system model is depicted in Fig. 1 and consists of N SBSs and U users. Designing a distributed content network reduces multiple transmissions at the downlink and hence redundancy is reduced. This way a user can fetch a file from local SBS or neighboring set of SBS. Caching of a file further involves an optimal cache placement algorithm in the caches of SBS which have limited storage capacity and also have to consider the ranking of files. The optimal cache content problem turns out to be a combinatorial problem and hence results in NP-hard problem. In [13], the authors proposed a epsilon close-to-optimal caching solution and also show that the proposed algorithm performs better with respect to average cache hit and average delay.

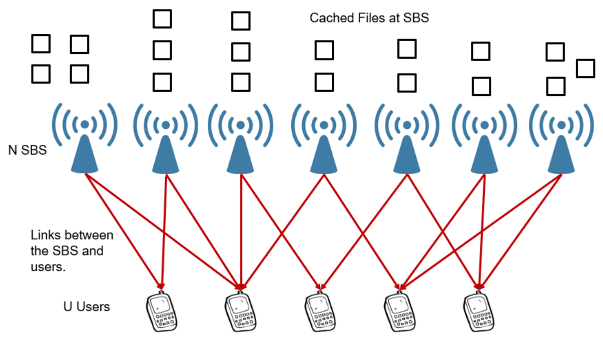

Fig. 1: System model of small base station caching.

B. *Device to Device Caching*

Device-to-Device (D2D) caching provides direct communication between devices, without the data being transferred from SBS. In a D2D communication the users are enabled to connect directly with each other, without or limited connectivity with the SBS.

With coordinated small-cell systems, considerable gain can be achieved. Typically, edge devices have less computing power, therefore the integration of ML with the edge devices will meet the needs of real-time services such as real-time data processing with low latency. In a mulit-agent setting, where the behavior of other users are unknown, the users can learn from each other using ML techniques. In [14], without assuming the knowledge of the rank vector of files in D2D networks, the content caching strategies using multi-agent reinforcement learning is designed. To maximize the expected total caching reward, D2D caching problem is formulated as a multi-agent multi-armed bandit learning problem in [14]. Then, the Q-learning algorithm is used to determine how to integrate the caching decisions in a multi-agent systems. Thus, a multi agent system in which the devices interact with each other and also learn from each other significantly improves system's performance.

III. Edge Caching using Reinforcement Learning

RL enables the agent to learn to behave in an environment by performing actions and then analyzing the results [15]. The task of Q-learning is usually described as Markov Decision Process (MDP), however, state space, transition probabilities and reward functions are not required. Therefore, it can be used in a situation when the model is known, but an analytical solution is not available. In particular, when the underlying transition probabilities are unknown, a caching policy is defined as a MDP and hence a Q-learning caching algorithm is formulated for estimating the average cache hit. The system model is illustrated in Fig. 1 which consists of $\mathcal{N}$ SBSs and $\mathcal{U}$ mobile users. Each SBS is assumed to have a storage capacity of $\mathcal{F}$. The scheduling event and the caching mechanism are considered independent of each other in this problem. The popularity of a file determines which files needs to be cached and hence the caching policy is affected by the demand profiles. Towards this, when the user demands are unknown, the problem of joint optimization of recommendation and caching to maximize the cache hit is considered in this section.

The variable $d_{f,n,t}$, represents the sum total of all the requests for file f at the SBS n at time t. The various factors like rank of a file, type of a file, genre of a file and other global factors are to be considered when estimating the demand requests. The nature/type of demand is captured in a attribute vector denoted by $x_{f,n,t} \in \mathcal{R}^{\mathcal{N}}$, $1 \leq f \leq F$. The attribute vector includes the details of the types of the files, genre of the files etc. The recommendation variable is denoted by $\beta_{f,n,t}$ for file f at time t and can take value in $\{0, 1\}$. Recommendation changes the demand rate of a file, and to capture this effect, variable $\delta_{f,n,t} \in \mathcal{R}$ is used. An illustration of the same would be, suppose we assume that a user demands for a file f at time t which is not cached at the SBS n, then based on the previous requests by the user, a similar file can be recommended instead of the original file and that is captured by the vector $\delta_{f,n,t}$. Another scenario could be, a file f is very relevant or popular at time t, then if that file is recommended to other users as well, the user requests may shoot up and hence the demand rate is altered. The variable $\beta_{f,n,t}$ takes the value 1 when the file is recommended in the nth SBS else it is 0.

All SBSs are assumed to have a total cache size of C. Also, all the files are assumed to be of equal size. Let $\lambda_{f,n,t} \in \{0, 1\}$ denote the absence/presence of a file f in the SBS n at time t. Then the average hit rate results in a maximization problem, and can be written as $\max \sum_t \sum_n \sum_{f \in F_{t,n}} \lambda_{f,n,t} \mathbb{E}[d_{f,n,t}|d_o]$, such that cache constraint at each SBS is satisfied (i.e. $\sum_n \lambda_{f,n,t} \leq C$).

The above problem cannot be solved since $d_{f,n,t}$ and its statistics are unknown at the central node. Here, we assume that the demand $d_{f,n,t}$ follows a Markov chain. This is a reasonable assumption since the current demand depends on the previous user requests as well as associated with the previous caching actions ($\lambda_{f,n,t-1}$) and previous recommendations ($\beta_{f,n,t-1}$). Hence, $d_{f,n,t}$ can be considered as following a Markov chain. The Markov decision process model can be defined by the tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{R}(s, a)\}$, where $\mathcal{S}$ is the set of all possible states a SBS can take and it describes the current situation of the SBS, $\mathcal{A}$ represents the action or the decisions taken by the SBS affecting the dynamics of the process and $\mathcal{R}(s, a)$ is the reward obtained for each transition between the states. The Q-learning algorithm based on greedy exploration is summarized in Algorithm 1. Similarly in [16] a deep reinforcement learning framework is used to study the content cache placement in wireless network, where no prior assumptions are made on the popularity of the files.

For the simulation setup, the number of SBS is assumed to be 30. For simplicity, in the simulation, the assumption is made that each user requests file of similar size. The learning rate η is taken as 0.8 and the discount factor γ is chosen to be 0.8. Fig. 2 shows performance of the Q-learning algorithm with recommendation which is compared with Q-learning without recommendation and other caching policies, such as known LRU (Least

Algorithm 1 Pseudo code for Q-learning

```
1:  procedure Q LEARNING
2:      iteration ← 0
3:      $Q(s,a)$ ← initial $Q$-value function
4:      $\mathcal{S}, \mathcal{A}$ ← initial action and state
5:      while (iteration < maximum iterations) do
6:          Execute action $\mathcal{A}$
7:          Reward $\mathcal{R}$ and $\mathcal{S}'$ is observed
8:          Joint optimization of caching and
            recommendation (i.e. Choose $\mathcal{A}$)
                                          ▷ greedy selec
9:          $Q(s,a) \leftarrow Q(s,a) + \eta[\mathcal{R} + \gamma Q(s',a')$
                    $-Q(s,a)]$
10:         $\mathcal{S} \leftarrow \mathcal{S}'$
11:         $\mathcal{A} \leftarrow \mathcal{A}'$
12:         iteration ← iteration + 1
13:     end while
14: end procedure
```

References	Scenario	Objective	Development Tools
[15]	Multi-cell massive-MIMO system	Maximize average success probability	Reinforcement learning based algorithm
[17]	D2D enabled cellular networks	Minimize energy consumption	ML multiclass classification models
[17]	Localized cellular networks	Maximize the average cache hit rate	Bayesian learning
[18]	In-edge Content delivery network server	Maximize the average cache hit rate	Model free reinforcement learning

TABLE I: Machine Learning Based Algorithms

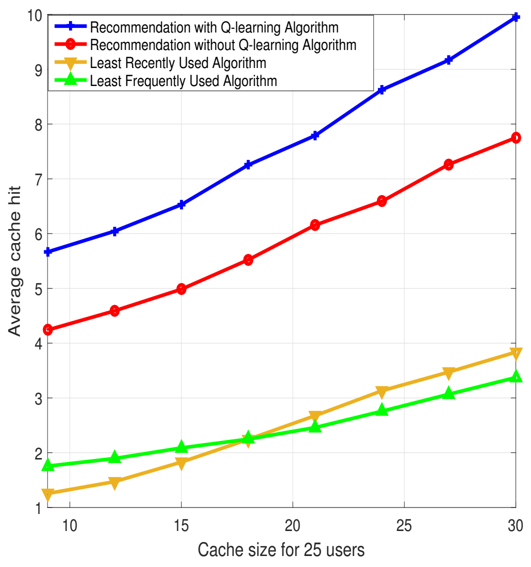

Fig. 2: Average cache hit versus cache size for 500 files .

Recently Used) and LFU (Least Frequently Used). It can be easily seen that the Q-learning algorithm outperforms all the other algorithms in terms of average cache hit. Thus, by intelligently exploiting user's information and popularity of files, a context aware caching system can be developed. Table I summarizes the other ML algorithms used for cache management.

IV. Content Popularity Prediction Using Transfer Learning

In this section, a TL method is used to estimate the rank vector of the cached contents. TL focuses on learning of the new tasks by transferring or reusing the information from previously learned tasks. Exploiting data available from other sources allows TL to predict the popularity with substantial accuracy. In [7], the authors consider a heterogeneous cellular network and propose a learning-theoretic analysis of content caching in heterogenous networks with non-stationary, statistically dependent and unknown popularity profiles. A heterogeneous cellular network is considered where the set $\Phi_u \subseteq \mathbb{R}^2$ of users, the set $\Phi_b \subseteq \mathbb{R}^2$ of base stations (BSs) and the set $\Phi_s \subseteq \mathbb{R}^2$ of SBSs are distributed as per the independent Poisson point processes (PPPs) with density λ_u, λ_b and λ_s for users, BSs and SBSs respectively. Using instantaneous demands from the users, the unknown popularity profile is estimated within a stated time frame. The file size is assumed to be B bits and the data file is requested independently by each user from the set $\mathcal{F} : \{f_1, f_2, \ldots, f_N\}$, where N is the sum total of files. The efficiency of the caching algorithm depends on the user's request rate, the cache size, density of the SBS nodes and the caching strategy. All SBSs are assumed to have a cache size of M files, where each file is B bits long. Each SBS interacts with the neighbouring user u using the following communication protocol.

Each SBS at $x_s \in \Phi_s$ communicates with a user u at $x_u \in \Phi_u$ if $||x_u - x_s|| < \gamma$;.

The main aim of the caching problem is to minimize the time taken to deliver a file due to the cache miss event when the requested file is not present. The BS collects all the demands from all the users in its coverage region and estimates the popularity profile in the time slot $[0, \tau]$. The demands are observed in the time duration $[0, \tau]$ and then the SBS calculates an estimate of the rank vector. For any $\epsilon > 0$, with a probability of at least $1 - \delta$, an offloading loss of $\hat{\tau}^* < \tau^* + \epsilon$ is obtained, with proof given in [7]. A lower bound on the waiting time is obtained. When the user density λ_u is greater than a threshold, an accuracy of $\epsilon > 0$ is achieved. So as to obtain an accuracy of $\epsilon > 0$, the waiting time can be reduced by using the knowledge gained from the users' interaction. The waiting time can further be reduced by combining samples from user's request pattern and the source domain.

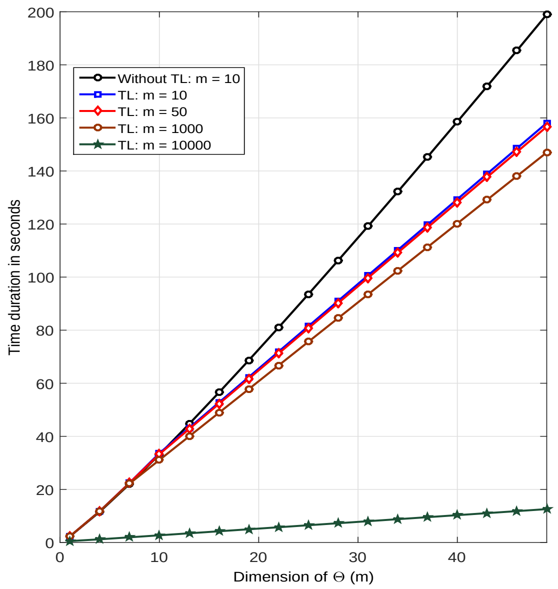

Fig. 3: Training duration versus Θ.

For the simulation the SBSs and users are distributed according to PPPs with densities $\lambda_b = 0.00001$ and $\lambda_u = 0.0001$, respectively. The number of files is $N = 100$, and the coverage of the BS and SBSs are 1000 m and 500 m, respectively. Fig. 3 shows the advantages of the TL-based approach for the parametric family of rank vectors. For values of m as low as 10, it can be observed that the TL method obtains significant better results in comparison with the source domain method. Also, the training duration increases as d increases, which is quite expected.

V. Popularity Prediction using Deep Learning

In this section, a DL approach has been used to predict the rank vector/popularity profile of the content. DL is basically stacked neural networks, wherein the network is composed of several layers. Computation takes place in the layers which are made of nodes. The node connects the input from the data with a set of coefficients, which either amplifies or dampen the input. Then, these weighted input products are added and are made to pass through an activation function to obtain the result. When the data has to be passed through multi-layers of node, then it no longer remains a single hidden layer neural network, and is referred as DL network. Due to its high accuracy in prediction, DL can be used appropriately in estimating the content popularity.

The system model consists of a single BS and multiple users spatially distributed according to a homogeneous Poisson point process. The frequency of contents (access pattern) follows Zipf-like distribution. To understand the type of user access styles, the analysis of the exploratory data is performed on the commonly used Movie Lens dataset. For testing and training the performance of prediction models, a dataset covering $100,000$ ratings from 1000 users on 1700 movies are used. To measure accuracy of the prediction model, mean squared error (MSE) is chosen. Keras and Tensorflow libraries are used to implement the prediction models. Keras has been used to define three models: (a) user embedder that learns to represent each user's preference as a vector, (b) movie embedder that learns to represent each movie as a vector, and (c) rating model that concatenates user and movie embedding networks and applies a dense neural network to predict the rating. Keras functional API model is used such that the inputs and embedding layers are shared. Rectified linear unit (ReLU) stimulates the convergence of stochastic gradient descent and hence is used as an output activation function. To reduce the prediction loss, MSE is chosen as an appropriate loss function. To calculate the gradients, multiple optimizers are available, and depending on the performance comparison of the different optimizers, adaptive moment estimation (ADAM) is chosen. The training iterations are then repeated over many epochs for loss minimalization. Performance of the DL method is compared with the existing online learning and online prediction methods [19]. Fig. 4 illustrates the averaged MSE prediction for the various models. It is noticed that the DL based techniques can outperform the recently proposed Grassmannian prediction model (GPM) [20], linear popularity prediction model (PPM) [19], auto regressive (AR) prediction model [21], weighted follow-the-leader (FTL) [19], weighted follow-the-regularized leader model (FoReL) [19], and mean guessing prediction model [22] for MSE. It is observed from Fig. 4 that DL has the high prediction accuracy, since it has the least MSE value i.e. the value of MSE for the proposed DL algorithm is 0.025, while while the MSE value for GPM, PPM, mean guessing, FTL, FoReL and AR are 0.06, 0.07, 0.09, 0.16, 0.36 and 0.45 respectively.

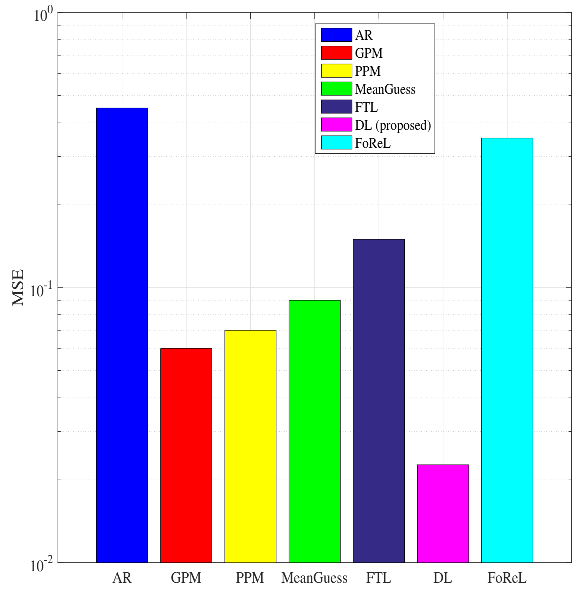

Fig. 4: Figure illustrates the average MSE prediction for various algorithms.

Fig. 5 shows the relationship between the hidden layers in DL model versus MSE. The total number of SBS and users are assumed to be 5

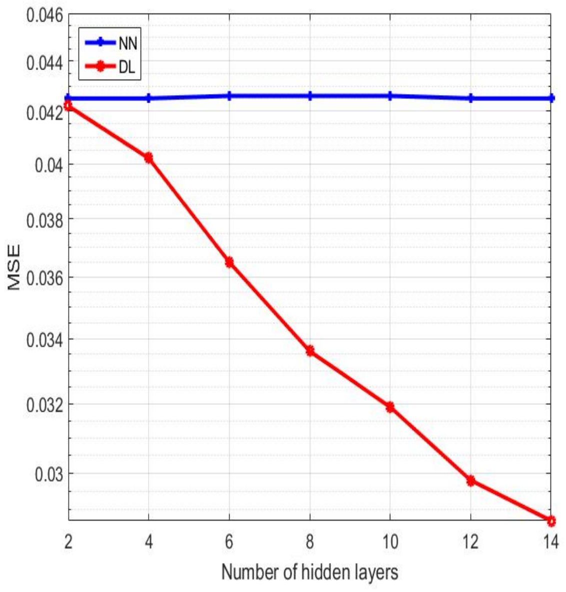

Fig. 5: MSE versus number of hidden layers with $N = 500$.

and 20 respectively. The DNN consists of an input layer, five hidden layers and an output layer. The five hidden layers have $1000, 1000, 3000, 3000$ and 1000 neurons. The content library size is fixed at 500 files (N). When the number of hidden layer is one, the model is referred to as a simple neural network (NN) and, it is observed that as we increase the number of layers in the DL model, the MSE reduces further and results in better performance.

VI. Federated Caching

One of the most promising distributed learning algorithms is the emerging FL framework. The data is collected and stored at multiple edge nodes, and a model is trained from the distributed data without sending the data from the nodes to a central node. This variant of distributed learning (model training) from a federation of edge nodes is known as FL [9]. In FL based caching the users are collaboratively trained to increase the overall speed. In the general setting, the caching strategy should be a function of the past requests and requests from neighboring SBSs. This leads to a complex online functional optimization problem, which is mathematically intractable and may lead to more complex algorithms. Therefore, before attempting to solve a general problem, it is natural to get insights on the general problem by solving a relatively simpler problem. This simplification is done by making suitable assumptions on the structure of the caching strategy. In addition, the motivation of using a linear combination of caching strategies come from online learning literature with iid (independent and identically distributed) data. A natural extension of the above to non-iid correlated requests is to use weighted average of the past strategies, and optimize the weights. Thus in the following section, we have assumed the structure mentioned in the following section for the caching strategy.

In this section, a FL based caching method is proposed which is assumed to be a weighted combination of past caching strategies of neighboring SBSs. The system model consists of M SBSs and U users and the library size is assumed to consist of F files of different sizes. The requests across SBSs and time is assumed to be correlated. The data available at the SBS b at time T is denoted by $Z_{b,1}^{T} \subseteq \mathcal{Z}_{b,1}^{T}$, which includes demands of SBS b until time slot T, and the data shared by the neighboring SBSs. The caching policy at the end of time slot $t-1$ for each file f, is given by $\pi_{b,f,t}$. The weighted average of a sequence of caching strategies $\boldsymbol{\pi}_{b,t}$ from time slot $t = T-\tau+1$ to T is given by

$$\bar{\boldsymbol{\pi}}_{b,T+1} := \sum_{t=T-\tau+1}^{T} \alpha_{b,t}\boldsymbol{\pi}_{b,t}, \tag{1}$$

where $\alpha_{b,t}$'s are the non-negative weights that satisfy $\sum_{t=T-\tau+1}^{T} \alpha_{b,t} = 1$. The caching strategy has been taken as a weighted linear combination of all the neighboring SBSs caching strategies and the past caching strategies. where the map $j_b : \mathcal{N}_b \to \{1,2,\ldots,|\mathcal{N}_b|\}$, and the weights are chosen to be non-negative with the constraint given by $\sum_{b' \in \mathcal{N}_b} w_{j_b(b')}^{T+1} + w_b^{T+1} = 1 \; \forall$ sBS b. The weights $w_{j_b(b')}^{T+1}$ as well as $\alpha_{b,t}$ are chosen in such a way the average cache hit is maximized. Thus assuming structured cache placement, a high probability bound on the conditional average cache hit is derived using martingale difference equation (see [9]). The insights provided by the bounds further helps in designing iterative FL based caching strategies. In the simulation, the demands are generated using the Movie Lens data set. The total number of files assumed is 800, i.e., the users can possibly request from only these catalog of Movie Lens data. Each

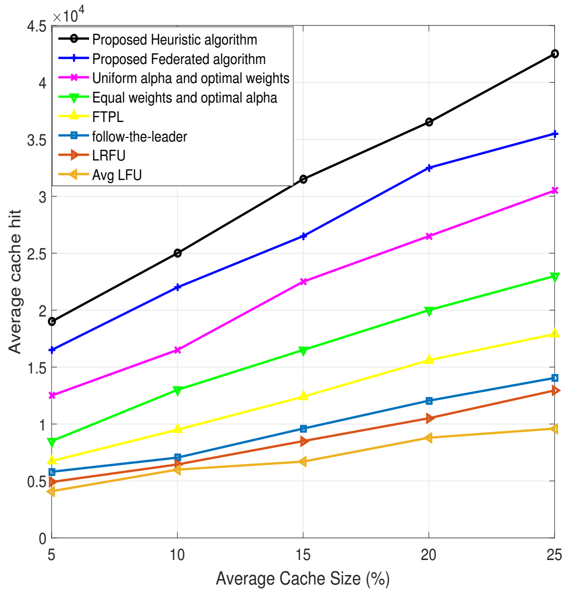

Fig. 6: Average sum cache hit versus cache size.

file is chosen uniformly random with a size of 10 to 100 units. Fig. 6 shows a plot of cache hit versus cache size for different caching algorithms. It is observed from the figure that both the proposed algorithms perform better than the LRFU, average LFU, follow-the-perturbed leader (FTPL), uniform α and optimal w, as well as uniform w with optimal values of α. The difference here is around 10^4 demonstrating the benefit of using the proposed scheme. The algorithms like LRFU, in which an average of the past demands of each file is listed in the decreasing order and thus the weights given across the time does not take into account the non-stationarity and correlation across time and hence for a correlated data it underperforms as compared to the proposed algorithms and results in 200 % gain.

VII. Future Directions & Challenges

To provide accurate caching strategies, huge amount of data needs to be collected and processed at the network edge. However, these data can be exposed to external attackers or eavesdroppers. Thus, storing user data at a centralized location poses security and privacy issues. Also, with the increasing amount of information collected through wireless edge devices, the centralized solutions are inherently becoming expensive. In such a scenario, FL plays an important role in protecting privacy-sensitive data. FL trains statistical models on the edge devices directly in a distributed fashion. Also at the edge networks, FL can play as permissive technology for collaborative model training, which further improves performance of the edge networks. Further, the multiple layers in DL can be trained jointly with information centric network such that the network performance is enhanced.

Since the time taken for the training is long, running ML algorithms with different hyperparameters is time-consuming. High computation resources are still required in DL to process the high-dimensional data to train the prediction model. In addition, the computing resources at the edge may be insufficient to process the high-dimensional data and thus computational efficiency and energy efficiency at the edge node should be appropriately considered when designing a ML scheme. The theoretical analysis of the size of the dataset required for training and evaluation on the convergence bounds of ML architectures are still open areas to explore. So as to make sure that DL and ML techniques work well in real-time systems, sound theoretical bounds and studies on all aspects are essential.

VIII. Conclusion

In this article ML techniques have been integrated with edge caching to advance the edge caching capabilities. This article highlights the ML techniques which have been used to demonstrate, model and analyze the mobile edge system and optimizing the mobile edge computing and caching with it. Different scenarios and the potential of integrating ML with the edge systems has been discussed with the help of ML methods. Results show that the edge caching and popularity prediction using ML achieves optimal performance when compared with other algorithms. Future plausible challenges have also been looked into and wireless edge computing techniques are expected to attract significant attention from both industry and academics. Though, it is difficult to design caching strategies and content popularity prediction algorithms, given the fact that a huge amount of data has to be handled under limited storage capacity, dynamic network topologies and other constraints. However, research on wireless edge caching and content popularity prediction using ML is a promising solution which opens up many research challenges and opportunities in this era of continuous data expansion.

Acknowledgement

This work is supported in part by the R&D project under the Visvesvarya PhD Scheme of Ministry of Electronics and Information Technology, Government of India, being implemented by Digital India Corporation (formerly Media Lab Asia), under Grant PhD-MLA/4(05)/2015-16 and by the Indian Institute of Technology Indore, India. This work is also supported by the Deanship of Scientific Research at Prince Sattam bin Abdulaziz University, Saudi Arabia.

References

[1] Cisco, "Cisco annual internet report (2018-2023)," 2020.

[2] S. Ali *et al.*, "6G White Paper on Machine Learning in Wireless Communication Networks," *arXiv e-prints*, p. arXiv:2004.13875, Apr. 2020.

[3] S. M. Nagarajan, G. G. Deverajan, P. Chatterjee, W. Alnumay, and U. Ghosh, "Effective task scheduling algorithm with deep learning for internet of health things (IoHT) in sustainable smart cities," *Sustainable Cities and Society*, vol. 71, 2021.

[4] S. K. Punia, M. Kumar, T. Stephan, G. G. Deverajan, and R. Patan, "Performance analysis of machine learning algorithms for big data classification: ML and AI-based algorithms for big data analysis," *International Journal of E-Health and Medical Communications*, vol. 12, 2021.

[5] K. Poularakis *et al.*, "Exploiting Caching and Multicast for 5G Wireless Networks," *IEEE Transactions on Wireless Communications*, vol. 15, pp. 2995 – 3007, 2016.

[6] C. Vallati, A. Virdis, E. Mingozzi, and G. Stea, "Mobile-edge computing come home connecting things in future smart homes using LTE device-to-device communications," *IEEE Consumer Electronics Magazine*, vol. 5, no. 4, pp. 77–83, 2016.

[7] B. N. Bharath, K. G. 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, Apr 2016.

[8] S. Krishnendu, B. N. Bharath, and V. Bhatia, "Joint edge content cache placement and recommendation: Bayesian approach," in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, 2021, pp. 1–5.

[9] S. Krishnendu, B. N. Bharath, N. Garg, V. Bhatia, and T. Ratnarajah, "Learning to cache: Federated caching in a cellular network with correlated demands," *IEEE Transactions on Communications*, pp. 1–1, 2021.

[10] P. Blasco and D. Gündüz, "Learning-based optimization of cache content in a small cell base station," in *2014 IEEE International Conference on Communications (ICC)*, 2014, pp. 1897–1903.

[11] A. Sadeghi, G. Wang, and G. B. Giannakis, "Deep reinforcement learning for adaptive caching in hierarchical content delivery networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 4, pp. 1024–1033, 2019.

[12] B. N. Bharath, K. G. Nagananda, D. Gündüz, and H. V. Poor, "Caching with time-varying popularity profiles: A learning-theoretic perspective," *IEEE Transactions on Communications*, vol. 66, no. 9, pp. 3837 – 3847, May 2018.

[13] S. Krishnendu, B. N. Bharath, and V. Bhatia, "Cache enabled cellular network: Algorithm for cache placement and guarantees," *IEEE Wireless Communincation Letters*, vol. 8, no. 6, pp. 1550–1554, Dec. 2019.

[14] W. Jiang, G. Feng, S. Qin, T. S. P. Yum, and G. Cao, "Multi-agent reinforcement learning for efficient content caching in mobile D2D networks," *IEEE Transactions on Wireless Communications*, vol. 3, pp. 1610 – 1622, Mar. 2019.

[15] N. Garg, M. Sellathurai, V. Bhatia, and T. Ratnarajah, "Function approximation based reinforcement learning for edge caching in massive MIMO networks," *IEEE Transactions on Communications*, pp. 1–1, 2020.

[16] C. Zhong, M. C. Gursoy, and S. Velipasalar, "Deep reinforcement learning-based edge caching in wireless networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 1, pp. 48–61, 2020.

[17] D. Prerna, R. Tekchandani, N. Kumar, and S. Tanwar, "An energy-efficient cache localization technique for D2D communication in iot environment," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4816–4829, 2021.

[18] V. Kirilin, A. Sundarrajan, S. Gorinsky, and R. K. Sitaraman, "RL-cache: Learning-based cache admission for content delivery," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 10, pp. 2372–2385, 2020.

[19] N. Garg, M. Sellathurai, V. Bhatia, B. N. Bharath, and T. Ratnarajah, "Online content popularity prediction and learning in wireless edge caching," *IEEE Transactions on Communications*, vol. 68, no. 2, pp. 1087–1100, Feb. 2020.

[20] Y. Yang and T. M. Hospedales, "Multivariate regression on the grassmannian for predicting novel domains," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5071–5080, 2012.

[21] H. Nakayama, S. Ata, and I. Oka, "Caching algorithm for content-oriented networks using prediction of popularity of contents," in *2015 IEEE International Symposium on Integrated Network Management*, 2015, pp. 1171–1176.

[22] Y. Jiang, M. Ma, M. Bennis, F.-C. Zheng, and X. You, "User preference learning-based edge caching for fog radio access network," *IEEE Transactions on Communications*, vol. 67, no. 2, pp. 1268–1283, Feb. 2019.

S. Krishnendu received the B.Tech. degree in Electronics & Communications Engineering from the National Institute of Technology Calicut, India, in 2016. She is currently pursuing the Ph.D. degree with the Department of Electrical Engineering, Indian Institute of Technology Indore, India. From 2016 to 2017, she worked as an Engineer with Tata Elxsi, Thiruvananthapuram, India. Her research interests include wireless communications, edge caching, optimization, and machine learning. She is currently a recipient of the Visvesvaraya Ph.D. Fellowship from MeitY, Government of India. Contact her at phd1701102001@iiti.ac.in.

B. N. Bharath is currently working as an Assistant Professor in the Department of Electrical Engineering at Indian Institute of Technology Dharwad, Karnataka. His research interests include signal processing and machine learning for communications, information theory, stochastic optimization, and wireless networks. He received his PhD from the Electrical Communications Engineering department of the Indian Institute of Science (IISc), Bangalore, India in 2013. He worked at Qualcomm Inc. from 2013 to 2014. Contact him at bharathbn@iitdh.ac.in.

Vimal Bhatia is currently working as a Professor with the Indian Institute of Technology Indore, India, and is an adjunct faculty at IIT Delhi and IIIT Delhi, India and UHK, Czech Republic. He received Ph.D. degree from Institute for Digital Communications with The University of Edinburgh, Edinburgh, U.K., in 2005. During Ph.D. he also received the IEE fellowship for collaborative research at the Department of Systems and Computer Engineering, Carleton University, Canada, and is Young Faculty Research Fellow from MeitY, Government of India. He is also a recepient of Prof SVC Aiya Memorial Award (2019). He has worked with various IT companies for over 11 years both in India and the UK. He is a PI/co-PI/coordinator for external projects with funding of over USD 17 million from MeitY, DST, UKIERI, MoE, AKA, IUSSTF and KPMG. He has more than 300 peer reviewed publications, and has filed 13 patents (with two granted). He has supervised 15 defended/submitted PhD thesis. His research interests are in the broader areas of communications, non-Gaussian non-parametric signal processing, machine/deep learning with applications to communications and photonics. He is a reviewer for IEEE, Elsevier, Wiley, Springer, and IET. He is currently Senior Member of IEEE, Fellow IETE and certified SCRUM Master. He was also the General Co-Chair for IEEE ANTS 2018, and General Vice-Chair for IEEE ANTS 2017. He has served as founder head of Center for Innovation and Entrepreneurship, Associate Dean R&D and Dean, Academic Affairs. He has delivered many talks, tutorials and conducted faculty development programs for the World Bank's NPIU TEQIP-III programs. He is currently Associate Editor for IETE Technical Review, Frontiers in Communications and Networks, Frontiers in Signal Processing, and IEEE Wireless Communications Letters. Contact him at vbhatia@iiti.ac.in.

Jamel Nebhen received the M.Sc. in Microelectronics from the National Engineering School of Sfax, Tunisia in 2007, and the Ph.D. degrees from the Aix-Marseille University, France, in 2012, all in Microelectronics. From 2012 to 2018, he worked as a Postdoctoral Researcher in France in LIRMM-Lab Montpellier, IM2NP-Lab Marseille, ISEP Paris, LE2I-Lab Dijon, Lab-Sticc Telecom Bretagne Brest, and IEMN-Lab Lille. Since 2019, he joined the Prince Sattam bin Abdulaziz University in Alkharj, Saudi Arabia, as an Assistant Professor. His research interests are Wireless Communication Systems, Microwave Electronics, Internet of Things, Low power design, design of analog integrated circuits, analog and RF circuits for wireless communications. Contact him at j.nebhen@psau.edu.sa.

Tharmalingam Ratnarajah is currently with the Institute for Digital Communications, The University of Edinburgh, Edinburgh, UK, as a Professor in Digital Communications and Signal Processing. He was a Head of the Institute for Digital Communications during 2016 – 2018. His research interests include signal processing and information theoretic aspects of 6G wireless networks, full-duplex radio, mmWave communications, random matrices theory, interference alignment, statistical and array signal processing and quantum information theory. He has published over 400 publications in these areas and holds four U.S. patents. He has supervised 16 PhD students and 21 post-doctoral research fellows and raised $ 11+ million USD of research funding. He was the coordinator of the EU projects ADEL (3.7M€) in the area of licensed shared access for 5G wireless networks, HARP (4.6M€) in the area of highly distributed MIMO, as well as EU Future and Emerging Technologies projects HIATUS (3.6M€) in the area of interference alignment and CROWN (3.4M€) in the area of cognitive radio networks. Dr Ratnarajah was an associate editor IEEE Transactions on Signal Processing, 2015 – 2017 and Technical co-chair, The 17th IEEE International workshop on Signal Processing advances in Wireless Communications, Edinburgh, UK, 3 – 6, July 2016. Dr Ratnarajah is a Fellow of Higher Education Academy (FHEA). Contact him at T.Ratnarajah@ed.ac.uk.