Machine Learning-based Cache Optimization on MEC Platform

Waleed Akbar, Afaq Muhammad, Javier Jose Diaz Rivera, Wang-Cheol Song *
Department of Computer Engineering, Jeju National University, Jeju-si, South Korea
waleedwali786@gmail.com, philo@jejunu.ac.kr * Corresponding Author

***Abstract*—The amount of data generation is exponentially increasing over the past decade due to the widespread use of multimedia applications and social media platforms. Advanced real-time applications such as virtual reality, augmented reality, automated vehicles, smart homes, and intelligent traffic control systems have increased the demand for low latency. Many of these applications are delay-sensitive and put enormous stress on the core network to respond in real-time. CDN (Content Delivery Network) brings storage service to end-users proximity to provide low latency, high data throughput, and low traffic pressure to handle the problems mentioned above. Due to the limited storage capacity of the edge, only in-demand content should cache. Therefore, to optimally utilized the cache space, an efficient content caching and replacement policy is needed. To this end, in this paper, we propose an optimal content replacement algorithm. In this algorithm, a video request pattern is first generated based on a publicly available dataset. After that, a machine learning model is trained on cache logs data. As a result, the predicted video is deleted from the edge to make space for new videos. A real-time testbed is built on KOREN to check the performance of our model. The results based on MAE, MSE, and R-2 show that our model performs well in real-time scenarios.**

I. Introduction

The current era of Internet of Things (IoT) devices such as mobile devices, sensor devices, and wearable devices has exponentially increased the volume, velocity, and variety of data. According to Seagate, in the 'Digitization of World' white paper, the generation of big data will be increased from 45 zettabytes (2019) to 175 zettabytes (2025) [1]. Cisco reported that by the end of 2023, the number of Internet-connected devices would be three times the total earth population [2]. Cisco also published in 'cisco visual networking index', the total traffic generated by social media platform is the second largest contributor in mobile data traffic, with over 15% of traffic load [3]. The emergence of real-time applications, such as augmented reality and virtual reality, also added to the complexity and speed of data generation. This tremendous increase in speed of data generation imposed a high traffic load on the backhaul link and core network. Some studies [4], [5], [6] also mentioned the traffic increase due to the duplicate downloading of popular content.

The latest advancements have guided us to new technologies such as Mobile Edge Computing (MEC), Information-Centric Network (ICN), and Content Delivery Network (CDN). The MEC architecture receives considerable attention from researchers and industrial experts, and therefore, it is standardized by (European Telecommunications Standards Institute) ETSI [7]. MEC architecture provides caching capacity, compute service, and big data analytics in the proximity of end-user. Utilizing the MEC caching capabilities, the popular content can be stored on the cache to reduce the backhaul traffic congestion and latency.

MEC caching has been widely considered, and many techniques have been proposed to take advantage of this emerging technology. MEC has been implemented in many technologies such as V2V communication [8], IoT systems [9], [10], smart homes. Many researcher predicted that MEC caching has a major role in 5G networks [11], [12], [13], [14]. The technologies bring many benefits, but deployment remains a big challenge. [15], [16], [17], [18] proposed an efficient deep learning model for cache optimization. All these models are trained on a publicly available dataset, and the results are measured on the test dataset. The authors in [19] proposed a reinforcement learning model to understand user behavior based on historical data. The simulation is done on the MovieLens dataset to evaluate the results. [20] proposed a knapsack (dynamic programming algorithm) to find the optimal replacement policy. The simulation using MATLAB is done to evaluate the proposed algorithm. [18] used a hybrid approach of deep learning and collaborative filtering. The proposed model is evaluated on a test dataset, and no simulation work is done. [21] consider cache optimization problem as a multi-objective optimization problem; a GA (Genetic Algorithm) with ant colony optimization is used to solve this problem. The proposed model is numerically evaluated.

II. System Model

A. Problem Statement

By now, many research works have been published in MEC cache optimization [15], [16], [18], [19], [20], [22] and content replacement policy optimization [17], [21], [23]. Many popular papers proposed a cache optimization model based on machine learning [17], deep learning [15], reinforcement learning [19], dynamic programming [22], [20] and traditional weighted LRU [23] (Least Recently Used) models. According to the best of our knowledge, all published papers either simulated the testbed environment [15], [16], [17], [19], [20] or evaluates their model numerically [18], [21]. A simulated environment has many constraints and does not represent the real network environment. Without testing these models in a physical testbed, it is impossible to tell how the model

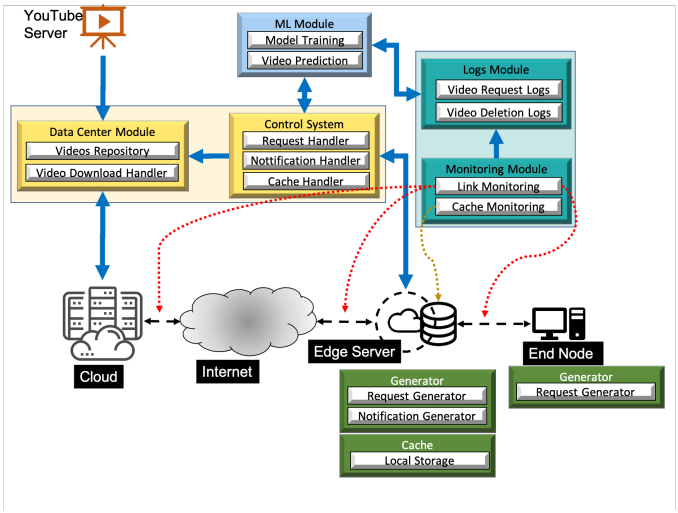

Fig. 1. System overview and components

will perform in real-world scenarios. Therefore, we have proposed a novel testbed environment to test and evaluate the cache optimization model. Our testbed environment is built on KOREN [24], and it consists of a datacenter node, edge node, and end/user device. Initially, we construct a cache usage dataset and train our DNN (Deep Neural Network) model. We predicted content to be deleted from a cache and evaluate the results on MAE, MSE, and R-2.

B. Architecture Overview

Figure 1 shows the overview of the whole system. The system consists of three main components, data center (DC), edge server, and end node. The data center stores all the available contents and provide content to edge once requested. The storage capacity of a data center is assumed to be unlimited. The edge server has limited resources, and it only caches popular content. It provides the content to the user from the cache if available; otherwise, it requests DC for content. The user generated the content request and received the content. Each component has sub-modules. The intelligence cache control system (ICCS) is deployed at the data center to monitor the working of the overall system. The ICCS is consists of six modules. In DC, the request handler module is responsible for receiving the video content request from the edge server. The data center module stores information about the requested video and deleted video contents. The machine learning (ML) module first finds the requested pattern and then predicts the content deleted from the cache. Cache handlers provide requested content to the edge server. The edge server is consisting of three modules. Request generator initiates content requests towards the data center. The notification generator initiates notification for the data center at cache max state. Local storage to cache the content temporarily. Edge server has limited storage space to cache popular content only.

Layer (type)	Output Shape	Param #	Connected to
user_input (InputLayer)	[(None, 1)]	0	
rating_input (InputLayer)	[(None, 1)]	0	
user_embedding (Embedding)	(None, 1, 50)	2650	user_input[0][0]
rating_embedding (Embedding)	(None, 1, 50)	21050	rating_input[0][0]
FlattenUsers (Flatten)	(None, 50)	0	user_embedding[0][0]
FlattenRating (Flatten)	(None, 50)	0	rating_embedding[0][0]
dropout_29 (Dropout)	(None, 50)	0	FlattenUsers[0][0]
dropout_30 (Dropout)	(None, 50)	0	FlattenRating[0][0]
Simalarity-Dot-Product (Dot)	(None, 1)	0	dropout_29[0][0] dropout_30[0][0]
dense_26 (Dense)	(None, 96)	192	Simalarity-Dot-Product[0][0]
dropout_31 (Dropout)	(None, 96)	0	dense_26[0][0]
dense_27 (Dense)	(None, 1)	97	dropout_31[0][0]

Fig. 2. DNN Model - Layer and Parameters Configuration

C. System Flow

There are two basic flows, control flow and data flow. Data flows consist of request and response messages between the data center, edge server, and end node. Two system entities generate the request messages: video requests by the end device and the edge node. Likewise, two system entities generate the response messages: video requests by the end device and the data center. Control signals are responsible for overall system management, such as notification messages, notification responses, and other messages.

D. Machine Learning Model

We use DNN (Deep Neural Network) machine learning model; the model configuration is depicted in figure 2. In pre-possessing, after sorting the dataset with time, the timestamp and video title is removed from the dataset. The video genre feature is encoded into a unique float id from the text. There are no NULL values, and Nan is found in the dataset. User Id, genre, and rating are given as input to the model, and video Id is predicted output. As shown in the figure, the user vector and rating vector are also used as embedding layers. A three 50% dropout layers are added. A dense layer with 96 neurons is added, and the output layer with one neuron is added. The RELU function is used as activation in all layers. The batch size for model training is two.

E. Mathematical Formulation

Let there are N number of videos available in data center, represented as equation 1. Each video v_n is belong to a genre G and their are K generics, depicted in equation 2. let T be set of time intervals represented in equation 3 and J is total number of time intervals. S_c represents the size of cache storage. Equation 4 represents the binary decision variable of video availability in a cache. The size of video v_n is represented as S_{v_n}.

$$V = \{v_1, v_2, v_3, \ldots, v_N\} \qquad N = Total\ videos \tag{1}$$

$$G = \{g_1, g_2, g_3, \ldots, g_K\} \qquad K = Total\ generics \tag{2}$$

$$T = \{t_1, t_2, t_3, \ldots, t_J\} \qquad J = Total\ time\ slots \tag{3}$$

TABLE I
DATA CENTER DESCRIPTION

S. No.	Attributes	Description
1	User count	104
2	Video count	1095
3	Rating count	10,517
4	Rating range	1 (worse) to 5 (best)
5	Video genre	Interview, Education, Sports, Films, Animation, NEWS, Music, Travel

$$X_{v_n} = \begin{cases} 1, & \text{if video } (v_n) \text{ is available in cache} \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

$$\sum_{v_n \in V} [v_n(t) * X_{v_n}], \qquad t \in T \quad (5)$$

$$\pounds_{v_n,cc}, \qquad v_n \in V \quad (6)$$

$$\pounds_{v_n,dc}, \qquad v_n \in V \quad (7)$$

$$\sum_{v_n \in V} r_{v_n}(t), \qquad t \in T \quad (8)$$

$$\sum_{v_n \in V} [S_{v_n}(t) * X_{v_n}], \qquad t \in T \quad (9)$$

Equation 5 represents available videos on cache at time t. Latency to fetch a video from cache and data center is represented by equations 6 and 7, respectively. The number of received video requests is represented by equation 8. Cache memory utilization a time t is represented in equation 9. Equation 10 represents the objective function. The objective is to minimize the total download latency.

$$min \sum_{v_n \in V} [X_{v_n} * \pounds_{v_n,cc} + (1 - X_{v_n}) * \pounds_{v_n,dc}] \quad (10)$$

III. TRAINING AND EXPERIMENTAL SETUP

A. Dataset

Firstly, we downloaded 1095 videos from YouTube; these video belongs from eight different genres, listed in table I. Secondly, the hidden user request pattern is captured from the MovieLens dataset [25]. Lastly, We generate user requests and collect cache utilization logs with the timestamp. Whenever there is a new update in the cache, the deleted video log is captured and stored in the central log files. The dataset detail is mentioned in table I. It takes 19 hours to complete this experiment.

B. Test-bed Setup

Our tested environment is built on KOREN (Korea Advance Research Network). A KOREN provides a testbed for research and development purposes. Our experiments are conducted on physical PCs deployed in different cities. Datacenter PC is placed in Gwangju city, and the other two PCs are placed in Jeju city. The edge server is deployed in JNU (Jeju National University) network operation center, and the end node is deployed in our lab. The PC specifications are mentioned in table III.

TABLE II
DELETED VIDEOS LOG DESCRIPTION

S. No.	Attributes	Description
1	User Id	User that requested the video
2	Video Id	Video to be requested
3	Rating	Rating attached with video (1 to 5)
4	Genre	Attached with video
5	Time stamp	Time stamp of deleted video

TABLE III
NODES SPECIFICATIONS

Specifications	DC Node	Edge Server	End Node
OS	Ubuntu 18.04 LTS		
Environment	Conda Virtual Environment		
Libraries	Pandas, NumPy, Threading libraries		
NIC	10 GB/sec		
CPU Cores	40	4	16
RAM	132 GB	16 GB	16 GB
HDD	1 T.B	200 GB	500 GB
Network Link	100 GB link		Over the Internet

IV. RESULTS AND DISCUSSION

Figure 3 show the training and validation loss over the 50 epochs. The training loss decreases with the increase in epochs; after 30 epochs, the training loss becomes stable. The validation loss continues to be stable after few epochs. Both the validation and training loss are almost the same and less than 0.05. These results show that our model neither over-fit nor under-fit. The figure 4 shows the comparison between predicted video ids and true values. In the figure, most of the predictions are near to higher video ids because the higher video ids are popular videos. Table IV show the MSE (Mean Square Error), MAE (Mean Absolute Error), and R2 error values of our model. These metrics show the error between true video ids and predicted video ids.

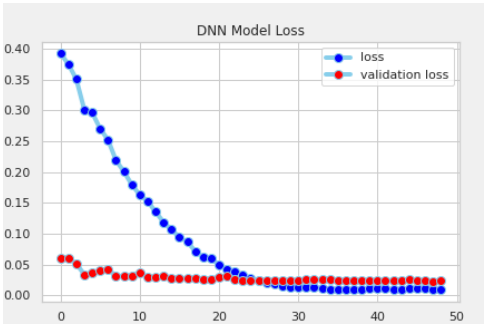

Fig. 3. DNN Model - Training and Validation Loss

TABLE IV
ERROR METRIC RESULTS

S. No.	Metric Name	Value
1	MSE	9.817
2	MAE	0.839
3	R2	0.821

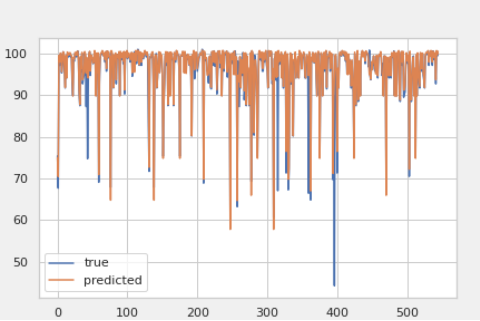

Fig. 4. Video Id comparison - True and Predicted

V. Conclusion

In this paper, we proposed a cache optimization model based on a deep neural network. Firstly, we analyze the publicly available dataset (Movie-Lens) to extract the user request hidden pattern. The user request pattern is used to generate the request for our experiment, and a monitoring system is deployed to captured the cache utilization and user request log. Later, their logs are used to train the DNN model and model predict the video id to remove from the cache. This will optimize the cache memory utilization, and our result suggests that the proposed model has a minimal error. In the end, we can store the most popular content on the cache.

Acknowledgment

This research was one of KOREN projects supported by National Information Society Agency (No.1711125875)

This research was also supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2016R1D1A1B01016322).

References

[1] SeaGate. *SeaGate Blog-Data Age by 2025*, 2021 (accessed March 9, 2021). https://www.seagate.com/as/en/our-story/data-age-2025/.

[2] CISCO. *Cisco Predicts More IP Traffic in the Next Five Years Than in the History of the Internet*, 2021 (accessed March 9, 2021). https://newsroom.cisco.com/press-release-content?type=webcontent&articleId=1955935.

[3] Cisco Visual Networking Index. Forecast and methodology, 2016–2021. *White paper, Cisco public*, 6, 2017.

[4] Li Qiu and Guohong Cao. Popularity-aware caching increases the capacity of wireless networks. *IEEE Transactions on Mobile Computing*, 19(1):173–187, 2019.

[5] Yipeng Zhou, Liang Chen, Chunfeng Yang, and Dah Ming Chiu. Video popularity dynamics and its implication for replication. *IEEE transactions on multimedia*, 17(8):1273–1285, 2015.

[6] Jeffrey Erman, Alexandre Gerber, Mohammad Hajiaghayi, Dan Pei, Subhabrata Sen, and Oliver Spatscheck. To cache or not to cache: The 3g case. *IEEE Internet Computing*, 15(2):27–34, 2011.

[7] Yun Chao Hu, Milan Patel, Dario Sabella, Nurit Sprecher, and Valerie Young. Mobile edge computing—a key technology towards 5g. *ETSI white paper*, 11(11):1–16, 2015.

[8] Muhammad Saqib, Asif Mehmood, Adeel Rafiq, Afaq Muhammad, and Wang-Cheol Song. Distributed sdn based network state aware architecture for flying ad-hoc network. In *2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pages 25–30. IEEE, 2020.

[9] Asif Mehmood, Faisal Mehmood, and Wang-Cheol Song. Cloud based e-prescription management system for healthcare services using iot devices. In *2019 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 1380–1386. IEEE, 2019.

[10] Shabir Ahmad, Faisal Mehmood, Asif Mehmood, and DoHyeun Kim. Design and implementation of decoupled iot application store: A novel prototype for virtual objects sharing and discovery. *Electronics*, 8(3):285, 2019.

[11] Asif Mehmood, Talha Ahmed Khan, Javier Diaz Rivera, and SONG Wang-Cheol. An intent-based mechanism to create a network slice using contracts. *abc*, pages 180–181, 2018.

[12] Khizar Abbas, Muhammad Afaq, Talha Ahmed Khan, Adeel Rafiq, Javed Iqbal, Ihtesham Ul Islam, and Wang-Cheol Song. An efficient sdn-based lte-wifi spectrum aggregation system for heterogeneous 5g networks. *Transactions on Emerging Telecommunications Technologies*, page e3943, 2020.

[13] Khizar Abbas, Muhammad Afaq, Talha Ahmed Khan, Adeel Rafiq, and Wang-Cheol Song. Slicing the core network and radio access network domains through intent-based networking for 5g networks. *Electronics*, 9(10):1710, 2020.

[14] Khizar Abbas, Talha Ahmed Khan, Muhammad Afaq, and Wang-Cheol Song. Network slice lifecycle management for 5g mobile networks: An intent-based networking approach. *IEEE Access*, 2021.

[15] Laha Ale, Ning Zhang, Huici Wu, Dajiang Chen, and Tao Han. Online proactive caching in mobile edge computing using bidirectional deep recurrent neural network. *IEEE Internet of Things Journal*, 6(3):5520–5530, 2019.

[16] Yayuan Tang, Kehua Guo, Jianhua Ma, Yutong Shen, and Tao Chi. A smart caching mechanism for mobile multimedia in information centric networking with edge computing. *Future Generation Computer Systems*, 91:590–600, 2019.

[17] Jie Liang, Dali Zhu, Haitao Liu, Heng Ping, Ting Li, Hangsheng Zhang, Liru Geng, and Yinlong Liu. Multi-head attention based popularity prediction caching in social content-centric networking with mobile edge computing. *IEEE Communications Letters*, 2020.

[18] Yu Chen, Yong Liu, Jingya Zhao, and Qinghua Zhu. Mobile edge cache strategy based on neural collaborative filtering. *IEEE Access*, 8:18475–18482, 2020.

[19] Wei Jiang, Gang Feng, Shuang Qin, and Yijing Liu. Multi-agent reinforcement learning based cooperative content caching for mobile edge networks. *IEEE Access*, 7:61856–61867, 2019.

[20] Xing Chen, Lijun He, Shang Xu, Shibo Hu, Qingzhou Li, and Guizhong Liu. Hit ratio driven mobile edge caching scheme for video on demand services. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1702–1707. IEEE, 2019.

[21] Danyue Wang, Xingshuo An, Xianwei Zhou, and Xing Lü. Data cache optimization model based on cyclic genetic ant colony algorithm in edge computing environment. *International Journal of Distributed Sensor Networks*, 15(8):1550147719867864, 2019.

[22] Sanshan Sun, Wei Jiang, Gang Feng, Shuang Qin, and Ye Yuan. Cooperative caching with content popularity prediction for mobile edge caching. *Tehnički vjesnik*, 26(2):503–509, 2019.

[23] Chunlin Li, Mingyang Song, Shaofeng Du, Xiaohai Wang, Min Zhang, and Youlong Luo. Adaptive priority-based cache replacement and prediction-based cache prefetching in edge computing environment. *Journal of Network and Computer Applications*, 165:102715, 2020.

[24] Korea MoS. *Korea Advance Research Network*, 2019 (accessed March 9, 2021). http://www.koren.kr/eng/Intro/intro01.html.

[25] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.