Journal of Network and Computer Applications 201 (2022) 103342

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Multi-neural network based tiled 360° video caching with Mobile Edge Computing

Shashwat Kumar [a,b,*], Lalit Bhagat [c], Antony Franklin A. [a], Jiong Jin [b]

[a] Department of Computer Science and Engineering, Indian Institute of Technology Hyderabad, India
[b] School of Science, Computing and Engineering Technologies, Swinburne University of Technology, Melbourne, VIC 3122, Australia
[c] Samueli School of Engineering, Computer Science Department, University of California, Los Angeles, USA

ARTICLE INFO

Keywords:
Multi-access Edge Computing (MEC)
Deep learning
360° video caching

ABSTRACT

It is challenging to stream 360° videos over the mobile network for its stringent latency and high bandwidth requirements. Although edge-based viewport adaptive tiled 360° video streaming solutions alleviate the bandwidth demand, the backhaul congestion and low latency concern remain persistent when data is served from the Content Delivery Network over the Internet. Edge caching can help mitigate these issues by storing the content at the edge of the cellular networks on the base station. However, caching 360° videos is challenging because of the large file size, which is further convoluted by tile selection in caching decisions. In this work, we propose a Mobile Edge Computing (MEC) based tiled 360° caching solution that uses Long–Short-Term-Memory (LSTM) and Convolutional Neural Network (CNN) in conjunction to address the challenges associated with 360° video caching. Specifically, the LSTM model predicts the future popularity of the videos, assisting in cache replacement decisions. For the selected videos, the CNN model, which is trained using the saliency map of the video, identifies the most engaging tiles in the videos for caching using the video content itself. The caching of tiles instead of the whole 360° videos improves the caching efficiency of the resource-constrained MEC server. The LSTM model is optimized based on the loss value of different hyperparameters, and AUROC (Ares Under ROC Curve) is used to evaluate the accuracy of the CNN model. Both the models produce highly accurate results. The results from extensive simulations show that the proposed solution significantly outperforms the existing methods. It improves the cache hit rate by at least 10% and reduces the backhaul usage by at least 35% with significant improvement in end-to-end latency, which is crucial for the quality of experience in 360° video streaming.

1. Introduction

The 360° video streaming, also known as panoramic or omnidirectional video, is getting popular for its immersive experience. It is expected to grow at 48.7% CAGR (Compound Annual Growth Rate) to become a global market of $47.7 billion by 2024 (Mordor Intelligence, 2020). The users access the 360° video content through dedicated Head-Mounted Displays (HMD) such as Oculus and HTC Vive, or by placing their smartphone in a dedicated headset, e.g., Google Cardboard and Samsung Gear VR. Smartphone-based 360° video streaming uses wireless networks and is expected to dominate the Virtual Reality (VR) market (Mordor Intelligence, 2020). However, delivering the 360° video over the wireless network is challenging for its high bandwidth and stringent latency requirements. The 360° video streaming requires 400 Mbps transmission rate to deliver high-definition (960×720) user experience (Song et al., 2019) with a smooth playback (Huawei, 2016). It is difficult to achieve in wireless networks, more specifically in cellular networks (McGarry, 2021). Besides the high bandwidth requirement, the end-to-end latency is another practical challenge in achieving high-fidelity 360° video streaming over cellular networks (Huawei, 2016).

In 360° video streaming, the viewer watches only a small part, which is in the Field-of-View (FoV) of the whole spherical scene. Previous studies (Qian et al., 2018; Zhang et al., 2019; Fan et al., 2020) have investigated this characteristic of 360° video streaming to reduce the bandwidth requirement by spatial prioritization of 360° video content. Tilling enables the viewport adaptive 360° video streaming where 360° video is spatially divided into small tiles which can be encoded and transmitted independently. Although viewport-adaptive tiled 360°

* Corresponding author at: Department of Computer Science and Engineering, Indian Institute of Technology Hyderabad, India.
E-mail addresses: cs15resch11011@iith.ac.in (S. Kumar), lalitbhagat7@cs.ucla.edu (L. Bhagat), antony.franklin@iith.ac.in (Antony Franklin A.), jiongjin@swin.edu.au (J. Jin).

https://doi.org/10.1016/j.jnca.2022.103342
Received 31 July 2021; Received in revised form 10 December 2021; Accepted 6 February 2022
Available online 24 February 2022

video streaming significantly reduces bandwidth requirements, streaming 360° video from distant content servers is still challenging due to network latency and backhaul congestion. VR services are susceptible to latency, and if Motion-to-Photon (MTP) latency is more than 20 ms, users may feel a strong sense of dizziness (Hu et al., 2021). Although the viewport-adaptive 360° video streaming methods reduce the bandwidth requirement, the whole video is still fetched from the remote server to the network edge. Transmitting these large-size 360° videos stress the cellular network infrastructure, demanding further investments to accommodate 360° video-related traffic. Prior research substantiates that edge caching is an effective solution to alleviate these issues (Elazhary, 2019).

In-network caching effectively mitigates the network latency and backhaul congestion by storing the popular content at the network edge on the base station (gNBs in 5G). Moreover, caching at the network edge prevents the redundant data transmission of popular content, resulting in transit service payment reduction to Internet Service Providers (ISP) and consequently the Operational Expenditure (OPEX) for the network operator. Multi-access Edge Computing (MEC) (ETSI, 2014) brings the cloud computing capabilities at the network edge (e.g., base station) and enables content caching. Edge caching with MEC can mitigate the challenges associated with 360° video streaming. However, 360° video caching presents unique challenges dissimilar to legacy video caching.

The 360° video is characterized by a large file size, and the resource-constrained MEC server cannot cache all the videos. Tiled 360° video caching can potentially help in utilizing the MEC storage capacity effectively. However, different users might view different tiles in tiled 360° video streaming, which complicates the tile caching decision. Therefore, determining the tiles that capture the view of many users is critical for effective caching, and tiles are addressed as the most engaging tiles in the paper. The regular video caching methods are not designed to tackle these issues and do not consider that many tiles in 360° video are viewed sparsely. Therefore, a new caching method is required that can exploit the characteristics of 360° video streaming for efficient caching. Previous studies (Carlsson and Eager, 2020; Qian et al., 2018; Xie et al., 2017) have established a correlation between viewers' behavior during 360° video playback where most of the users watch only a small subset of the tiles. Thus, caching the most requested video tiles on edge would reduce the load on the backhaul links and the access latency. Recent studies (Song et al., 2019; Dai et al., 2020; Maniotis et al., 2020; Ballard et al., 2019) have exploited it for 360° video caching. However, prior works require knowledge of video popularity distribution and users' viewing patterns, which is not always available in real-world scenarios. Moreover, previous studies need past viewport traces for tiles' viewing probability estimation, which is difficult to acquire in a heterogeneous environment with multiple content providers and involves user privacy concerns as well.

This paper explores a novel edge caching approach that does not require the viewing history of tiles to identify the most engaging tiles in the 360° videos and prior knowledge of video popularity distribution. We propose a MEC-based tiled 360° caching solution that uses Convolutional Neural Network (CNN) in conjunction with Long–Short-Term-Memory (LSTM) for 360° video caching. A small number of popular content accounts for most of the user requests in video streaming, which can be modeled as a time-series (Zink et al., 2009). Therefore, the proposed method uses the LSTM model for video popularity prediction, which can accurately forecast the time-series date (Brownlee, 2018). The LSTM-based prediction model assists the caching decision by determining the expected future popularity of the videos. Once the popular 360° videos are identified for caching, determining the most engaging tiles is pivotal to curtail the space requirement. To identify these tiles, we utilize the video content itself through the saliency maps. The salient points represent the feature maps of a video frame, illustrating the prominent regions, and the tiles containing a higher number of salient points are likely to engage the users' attention and are expected to be viewed by many users. We pose the problem of identifying the most engaging tiles through saliency map as a classification problem and employ CNN, which is widely used for image recognition and classification, to solve it. A CNN model is trained to identify these tiles in the 360° video using the video saliency map, which identifies the most engaging tiles of the 360° video through binary classification. Thus the proposed method does not require past viewing data to identify the most engaging tiles. By employing LSTM for video popularity prediction in conjunction with CNN-based tile estimation, the proposed solution outperforms the existing solutions for 360° video caching. Following are the major contributions of this work,

- This work presents a comprehensive analysis of a 360° video dataset containing the viewport traces and establishes that the viewing probability of tiles might help in cache decision.
- Through the empirical evaluation of the dataset, this work asserts that the video saliency map can assist in determining the most engaging tiles in the 360° videos.
- The proposed solution does not require any prior knowledge of video popularity distribution and employs an LSTM model to learn it dynamically.
- The proposed 360° video caching solution uses the CNN model to classify the most engaging tiles using the video saliency map without any need for the past viewport data.
- A cache replacement algorithm is proposed that uses the LSTM for video popularity prediction in conjunction with CNN-based tile classification to efficiently use the MEC resources.
- The performance gain of the proposed solution is illustrated through extensive evaluation in simulation using the real-world network 360° video streaming traces.

The rest of the paper is organized as follows. Section 2 discusses the related research work on 360° video streaming. The viewer's behavior and viewing pattern are analyzed in Section 3, which establishes the motivation of this work. The system framework is explained in Sections 4 and 5 provides specifics of the proposed solution. Section 6 presents the evaluation methodology and the detailed analysis of the results. Finally, the paper is concluded in Section 7 with some possible future work.

2. Related work

In this section, we briefly review the literature related to edge caching of 360° videos. Edge caching presents an effective solution to improve the 360° video streaming in cellular networks in terms of observed latency (Poularakis et al., 2019; Wang et al., 2020; Mahzari et al., 2018-10) and quality of delivered content (Dai et al., 2020; Maniotis and Thomos, 2021). Furthermore, caching the popular content on the network edge reduces the backhaul link usage (Dai et al., 2020; Mahzari et al., 2018-10), energy consumption (Chakareski, 2017), and network operational cost (Wang et al., 2020). Hence, caching the popular content at the network edge saves valuable resources. Since the storage capacity of MEC is usually limited, several optimization approaches have been proposed to maximize the cache utilization for large-size 360° videos. Different methods of integration of networking, caching, and computing in wireless networks have been discussed in Wang et al. (2018) which are followed by works (Pang et al., 2018; Sukhmani et al., 2019) for edge caching of 5G VR applications. As discussed in Sun et al. (2019) and Chakareski (2020), MEC for VR caching comprises a fundamental trade-off between caching, computing, and communication cost. These techniques optimize the viewport fidelity efficiently for 360° videos but cannot meet the stringent Motion-to-Photon (MTP) requirement due to the time required to solve the complex global system optimization problem.

Some existing works consider 360° video streaming characteristics to utilize the cache storage optimally. The work in Carlsson and Eager (2020) explores this space and reports correlations in the viewing direction of users watching the same 360° video. The authors have deduced its implication on video caching by exploiting the overlap in the viewports. The video or tile level granularity can be selected to cache the 360° videos. At the tile level, most viewed tiles of the videos can be cached independently with limited cache capacity (Lu et al., 2019; Papaioannou and Koutsopoulos, 2019; Maniotis and Thomos, 2021). On the contrary, caching the whole video may lead to inefficiency, considering some of the cached tiles will be requested sparsely.

In some works (Maniotis and Thomos, 2021; Wang et al., 2020), Reinforcement learning (RL) has been applied for caching decisions. To use the RL for tiles-based 360° video caching, the large action space problem is tackled by assuming the virtual viewports and a fixed number of videos are cached. The RL-based method shows the performance improvement without any prior assumption about video popularity but requires retraining on the change in action space caused by the change in cache size. The tile-based collaborative caching for 360° video has been proposed in Maniotis et al. (2020), Maniotis and Thomos (2021) and Liu et al. (2021) which shows the benefit of making the caching decision at tile level and collaboration among the cache servers. In contrast to Maniotis et al. (2020) and Maniotis and Thomos (2021), authors in Papaioannou and Koutsopoulos (2019) investigate a tile-based caching scheme that aims to optimize the error between the requested and cached tile resolution across different viewports and the coverage of the tiles set. In this work, the authors analyzed the caching of tiles in different resolutions with layered encoding. Authors in Lu et al. (2019) and Shi et al. (2020) examine the joint caching, transcoding, and delivery of the 360° videos. Although methodologies proposed in Maniotis and Thomos (2021), Maniotis et al. (2020), Papaioannou and Koutsopoulos (2019), Lu et al. (2019) and Shi et al. (2020) demonstrate promising results for offline caching of 360° videos, they are not directly applicable for solving the online caching problem as they assume that the popularity distribution is known a priori.

Cheng et al. (2021) proposed a proactive caching method for 360° videos. Authors jointly considered computation offloading, data transmission, video coding, and proactive caching for 360° video streaming. However, they only cache the next video chunk based on FoV prediction and requires head motion information along with the saliency map. Ale et al. (2019) proposed a Bidirectional Recurrent Neural Network (BRNN) with a CNN layer to accurately predict the content popularity for proactive caching. Likewise, Nguyen et al. (2021) proposed an ensemble-based LSTM method for popularity prediction. Authors first use the LSTM method to identify the real-time content preference in each demographic user group. Then regression-based meta-learning model to unify obtained multiple demographic user preferences into a caching strategy. These works use deep learning tools to predict the content popularity for caching, but do not address the tile selection problem associated with 360° video caching.

The discussed works require head motion and tiles' playback data to predict the future FoV and to identify the most engaging tiles in 360° videos. In absence of these data, these works cannot be used for 360° video caching. Other learning-based methods emphasize on popularity prediction for proactive or reactive content caching and does not explore the challenges associated with 360° video caching due to its large file size and spatial segmentation. In contrast to the existing works, this work presents a tiled 360° video caching solution that does not require any prior knowledge about popularity distribution of videos and tiles. The proposed solution exploits the video content features to identify the most engaging tiles using a CNN-based classification model, unlike the existing works which assume that tiles' popularity distribution is available through estimation from past playback traces. Moreover, the proposed solution employs an LSTM-based prediction model to estimate the video popularity to make the caching decisions.

3. Viewing behavior analysis and motivation

This section first presents the analysis of users' viewing behavior during the 360° video playback. Then, we discuss the potential of viewport history for caching with its limitations, highlighting the motivation of the proposed solution. Lastly, a learning-based tiled 360° video caching approach is introduced based on analytical insights.

Here, an open-source 360° dataset (Lo et al., 2017) is analyzed to identify the users' viewing patterns and understand the behavior while watching the 360° video content. The dataset contains the orientation traces of 50 subjects while each subject watched the playbacks of 10 different 360° videos. The videos pertain to three different categories: (i) Computer Graphics (CG), fast-paced (ii) Natural Image (NI), fast-paced, and (iii) NI, slow-paced. Different categories help to understand the user behavior across the videos. In the dataset, each video is recorded at 30 FPS (frames per second), and only the initial one-minute playback is used to collect the viewport traces. Hence, the dataset comprises 500 traces of 360° video playbacks, and each trace contains 1800 data points. In the dataset, the viewing orientation of each frame is recorded by yaw, pitch, and roll angles. The analysis of this dataset reveals some interesting aspects of 360° video streaming.

3.1. Viewing orientation distribution

To comprehend the users' interaction in 360° video streaming, we inspect the histogram of viewing angles (yaw, pitch, and roll) for different video categories. The histograms in Fig. 1 indicate that regardless of the video category, the viewers primarily focus on the front center (0°) and viewing angles broadly lie within the 30°. While shooting, the main subject usually kept in the front view, which substantiates the viewing behavior in 360° video streaming. We can also observe that users' head movement is dominantly in the horizontal (yaw) direction, while head rotations are sparse since the roll is concentrated near zero. Therefore, it can be asserted that even though 360° video offers a panoramic view, users predominantly watch the 360° videos around the front center. These characteristics of 360° video streaming can be exploited to design an effective caching scheme.

3.2. Outlook of a simple probability-based caching solution

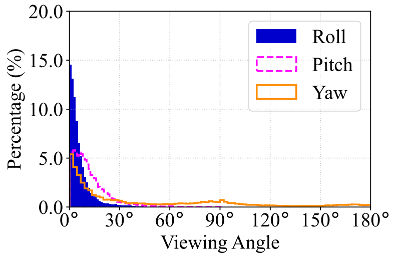
To further understand the viewing pattern in tiled 360° video streaming, an illustrative heatmap of viewing probabilities of the video tiles is plotted in Fig. 2a using previous viewing data. The higher brightness of some tiles depicts that these tiles have been viewed relatively more frequently. Fig. 2a reveals the disparity in viewership of the tiles that can be exploited to save the cache storage by not caching the sparsely viewed tiles.

Further, we study the tile caching for 360° video streaming based on viewing probabilities with predetermined probability thresholds to investigate the feasibility of a naive caching solution. For this study, the dataset is bifurcated into learning and testing traces, which are 75% and 25% of all traces, respectively. The probability (p_t) of tile t being in the FoV is calculated on learning traces and defined in Eq. (1),

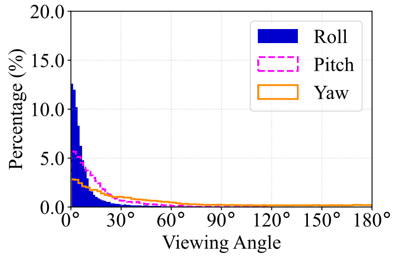
$$p_t = \frac{\sum_{i=1}^{i=N} x_t^i}{N} \mid x_t^i = \begin{cases} 1, & \text{if tile } t \text{ is in the FoV of } i^{th} \text{ playback.} \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

where N is the number of video requests and $x_t^i \in \{0, 1\}$ is one if tile t is in the FoV of i^{th} playback, zero otherwise.

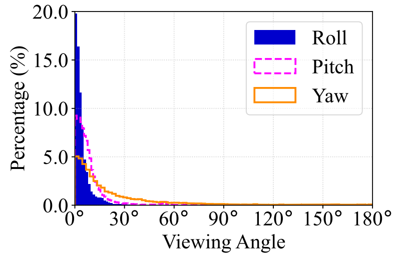
Figs. 2b and 2c represents the heatmaps for the cache miss-ratio and percentage of tiles cached for the videos, respectively. The horizontal axis represents the probability thresholds (P_T) such that if a tile t has viewing probability $p_t > P_T$, it is cached at the MEC. The vertical axis represents the total number of tiles in each video frame, where $(m \times n)$ signifies the number of tiles, horizontally (m) and vertically (n). For example, 3 × 3 denotes that each video frame is divided into three tiles horizontally and three tiles vertically, consisting of nine tiles. The heatmap in Fig. 2c indicates that an increase in probability threshold reduces the number of cached tiles and thereby increases the cache miss-ratio. For a probability threshold of 0.1 and 3 × 3 tiles, more than 90% of the 360° video tiles are cached. Consequently, cache miss-ratio



(a) NI, fast-paced

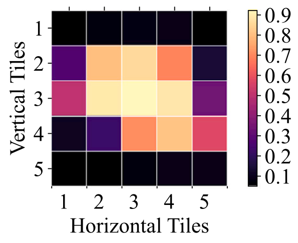


(b) NI, slow-paced

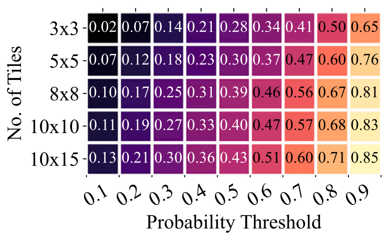


(c) CG, fast-paced

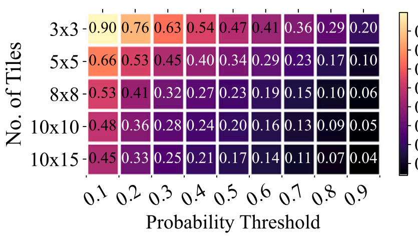
Fig. 1. Histograms representing users' viewing orientation (yaw, pitch, and roll angle) during video playback for three categories in the dataset (Lo et al., 2017).



(a) Video heatmap (5X5)

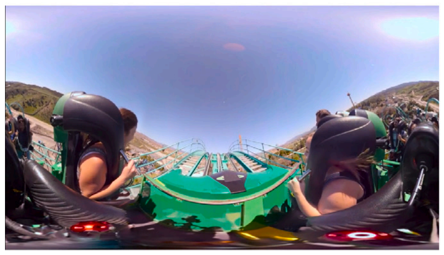


(b) Cache miss-ratio

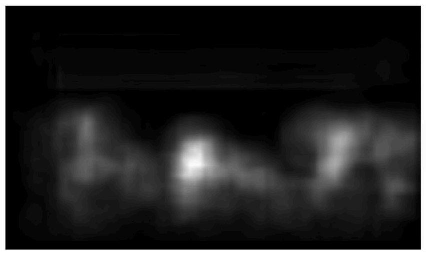


(c) Percentage of cached tiles

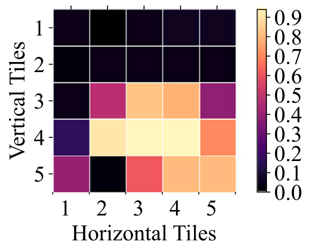
Fig. 2. (a) Illustrative heatmap of the tiles for Coaster video in the dataset. (b) Cache miss-ratio with different viewing probability thresholds. (c) Percentage of the cached tiles with different viewing probability thresholds.



(a) Video frame



(b) Saliency map of the frame



(c) Tiles' heatmap

Fig. 3. (a) A frame from the coaster video. (b) The Saliency map of the frame. (c) The heatmap of the tiles from the corresponding video frame.

is very low at 2%. But, caching a large number of tiles does not produce any significant storage savings compared to video caching. On the other end, for the probability threshold of 0.9 and 3 × 3 tiles, only 20% of the tiles are cached, but cache miss-ratio is significantly large (65%). Moreover, an increment in the number of tiles decreases the percentage of cached tiles for a given probability threshold while increasing the cache miss-ratio. An increase in the number of tiles results in a smaller tile size, permitting a fine-grained selection of tiles for a probability threshold.

This analysis illustrates that it is possible to apply a simplistic probability-based method for caching, but it has some serious limitations. First, the identification of a probability threshold that can balance the miss-ratio and storage requirement is an intricate task. Secondly, choosing the suitable tile size is also equally important for its impact on the cache hit-ratio and compression efficiency (Qian et al., 2018). More importantly, this method requires the viewing probability of the tiles, which might not be promptly available. Also, storing the viewport data may raise user privacy concerns as well. Hence, there is a need for a method that can identify the engaging tiles without using the viewport traces.

3.3. Exploration of saliency map to identify the most engaging tiles

We examine the video saliency map to identify the tiles that are expected to be viewed by many users for their prominent features. These tiles are labeled as the most engaging tiles in the paper. If the saliency map can help to determine these tiles, it can be exploited to find 360° video caching solutions without the need for past viewing data. This study uses the saliency maps provided in the 360° video dataset (Lo et al., 2017). The 360° video saliency maps in the dataset are generated using the CNN (LeCun et al., 1995). To generate the saliency maps, first, the saliency map of each frame is calculated using a Keras-based (Keras, 2021) script developed in Cornia et al. (2016). Then, saliency maps of frames are concatenated into a 1-min video. Fig. 3 shows a video frame from the dataset, its saliency map, and viewership heatmap of the tiles. Figs. 3b and 3c indicate a correlation between the salient points density and frequently viewed bright tiles in the heatmap. The salient points capture the notable regions in the frame, which eminently engage the users' attention. Therefore, the tiles containing these regions are relatively watched more frequently in the video. These preliminary results confirm that the saliency map can be utilized to identify the most engaging tiles without past viewport data

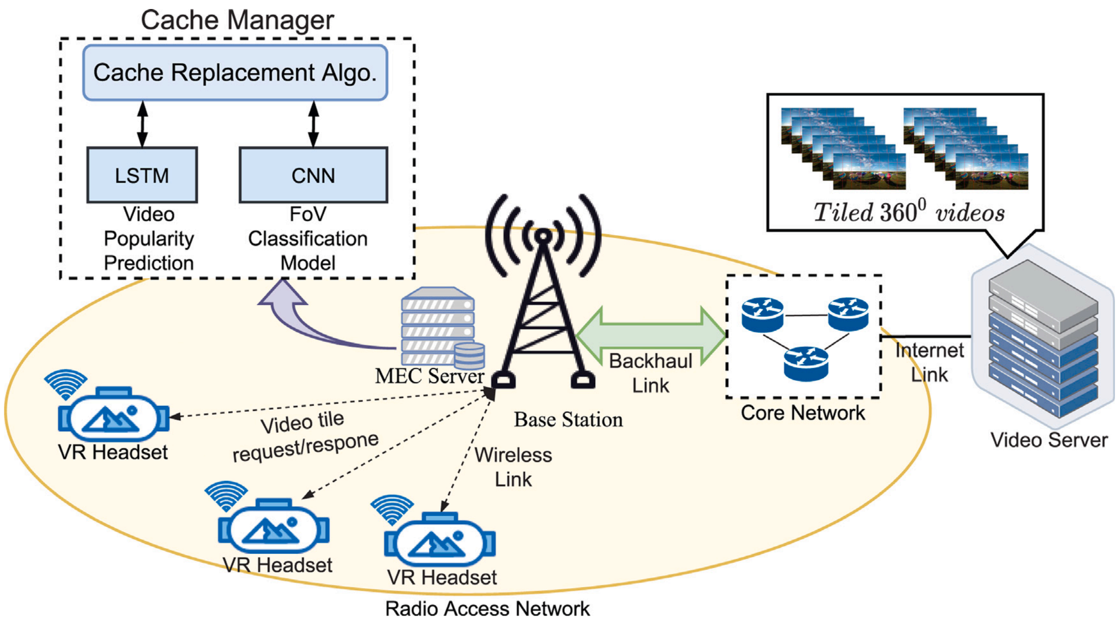


Fig. 4. System diagram depicting the deployment of the proposed solution on MEC server at the base station.

for probability estimations. Hence, the saliency maps can be exploited for the tile-based 360° video caching.

4. System framework

Fig. 4 depicts the system framework for the proposed solution deployed on the MEC architecture at the edge of the cellular network to satisfy the stringent latency requirement of 360° video streaming. The MEC at the network edge hosts the cache server, caching algorithm, popularity prediction module containing a pre-trained LSTM model, and a CNN model for classification of tiles. The viewers send the tile requests for 360° video segments to the MEC server at the network edge. The MEC server fetches these tiles from local storage if the video is cached on the MEC, otherwise requests it from the video server on the Internet. If video is not cached at the edge server, the cache replacement algorithm makes the replacement decision by considering the expected future requests estimated by the LSTM model. Once the replacement video is selected, the algorithm identifies the most engaging tiles for caching through classification using the CNN model. The proposed solution caches only the most engaging tiles of the video to boost cache storage utilization. Different modules of the solution framework are discussed as follows:

4.1. MEC platform

MEC provides storage and computation capabilities suitable for hosting the cache servers on the network edge (i.e., the base station) in the mobile network (ETSI, 2014) which can be accessed at very low latency. For the Ultra-Reliable Low Latency (URLL) applications such as virtual reality, edge deployment of the MEC servers are advisable (ETSI, 2018, 2014). However, the proposed solution can be employed with the other MEC deployment options, i.e., cell aggregation point, core network, etc. The cache manager module consists of (1) cache replacement algorithm, (2) LSTM model for popularity prediction, and (3) CNN model of tile classification. The cache manager is responsible for administering the cache server. Caching 360° video tiles at the MEC server minimizes the network latency during streaming and improves the users' Quality of Experience (QoE). If the requested tiles are available on the MEC servers, they are served from the network edge at very low latency without incurring any backhaul usage. Otherwise, it is fetched from the video server on the Internet, and the new tiles are cached on the MEC based on the decision taken by the cache replacement algorithm.

4.2. Cache replacement algorithm

The cache replacement algorithm is invoked if the requested video is not cached on the MEC server. It interacts with the popularity prediction and classification model for caching decisions. The decision to cache the new video is taken based on the expected future demand of the videos, predicted by the LSTM model. Instead of caching all the tiles, the most engaging tiles, which are predicted to be viewed more frequently by the users, are identified using a CNN classification model and cached on the MEC server to fulfill future requests.

4.2.1. LSTM model

LSTM networks are a special kind of Recurrent Neural Network (RNN), which are capable of learning long-term dependencies. LSTMs are explicitly designed to avoid the long-term dependency problem and remember information for long periods. Because of these capabilities, LSTMs are widely used for time-series predictions. In our framework, a trained LSTM model is deployed on the MEC server to predict the future demand based on the number of requests in the past. The LSTM assists the cache replacement algorithm to overcome the myopic nature by making the caching decision based on future expectations.

4.2.2. CNN model

A Convolutional Neural Network is a Deep Learning algorithm that receives an input image, assign weights and biases to different aspects in the image, and can differentiate one image from another. CNN requires much less pre-processing as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, CNN learns these filters/characteristics during training (Zhu et al., 2020). Considering these characteristics, we employ CNN for tile classification in the proposed solution. The cache manager uses the CNN model for tile classification based on the saliency map of the video to identify the most engaging tiles for caching. CNN model takes the tiles' saliency map as input and performs a binary classification to determine the most engaging tiles. These tiles are likely to contain a higher density of salient points and are anticipated to have higher views. Therefore, only the most engaging tiles from the video are cached at the MEC server.

5. Learning-based 360° video caching solution

In this work, we proposed a learning-based solution that does not require any prior knowledge of videos and tile popularity distribution.

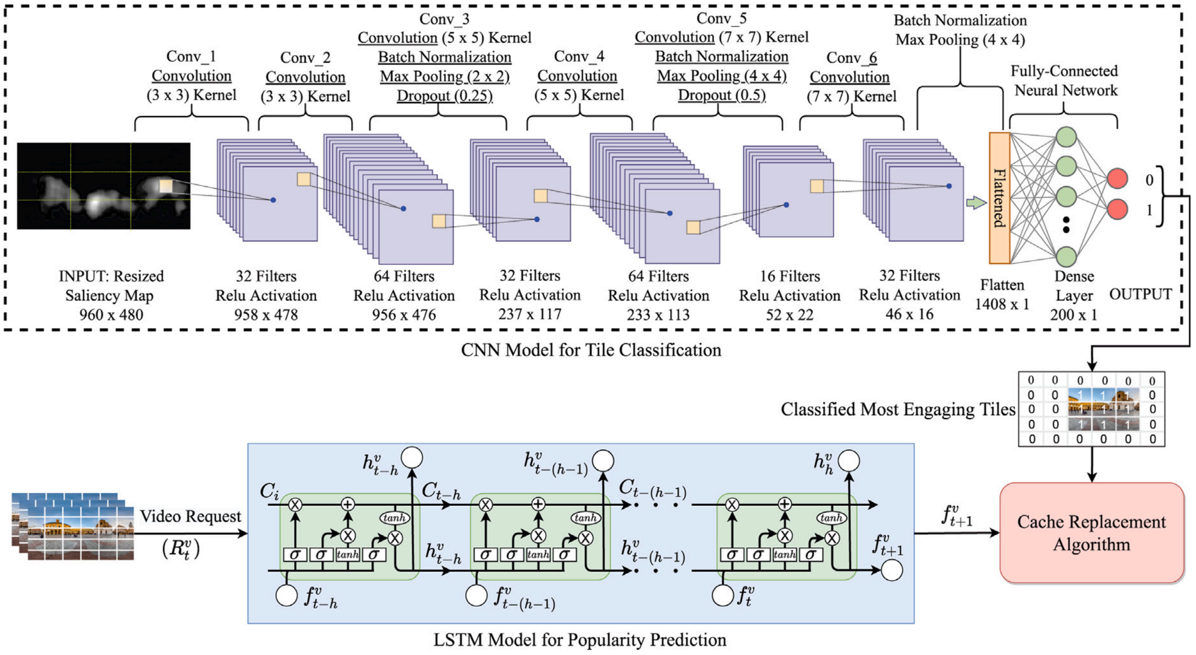

Conv_1 Convolution (3 x 3) Kernel
Conv_2 Convolution (3 x 3) Kernel
Conv_3 Convolution (5 x 5) Kernel Batch Normalization Max Pooling (2 x 2) Dropout (0.25)
Conv_4 Convolution (5 x 5) Kernel
Conv_5 Convolution (7 x 7) Kernel Batch Normalization Max Pooling (4 x 4) Dropout (0.5)
Conv_6 Convolution (7 x 7) Kernel
Batch Normalization Max Pooling (4 x 4)
Fully-Connected Neural Network
Flattened
INPUT: Resized Saliency Map 960 x 480
32 Filters Relu Activation 958 x 478
64 Filters Relu Activation 956 x 476
32 Filters Relu Activation 237 x 117
64 Filters Relu Activation 233 x 113
16 Filters Relu Activation 52 x 22
32 Filters Relu Activation 46 x 16
Flatten 1408 x 1
Dense Layer 200 x 1
OUTPUT
CNN Model for Tile Classification
Classified Most Engaging Tiles
Video Request (R_t^v)
C_i, C_{t-h}, $C_{t-(h-1)}$
h_{t-h}^v, $h_{t-(h-1)}^v$, h_h^v
f_{t-h}^v, $f_{t-(h-1)}^v$, f_t^v, f_{t+1}^v
LSTM Model for Popularity Prediction
Cache Replacement Algorithm

Fig. 5. Working model of the proposed solution with employed CNN and LSTM architectures .

We have applied deep learning tools to predict the video popularity and identify the most engaging tiles using the saliency map. The working of the proposed solution is depicted in Fig. 5. When a user sends a video request containing the tile information to the edge server, it checks the video in the cache. If video is cached on the cache server on MEC, then cached tiles are transmitted from the cache server. If video is not cached on the server, then video is fetched from the Content Delivery Network (CDN) or Internet, and the cache replacement algorithm is invoked to make the caching decision. The cache replacement algorithm uses an LSTM model to predict the future requests of the videos. If the new video is expected to get more requests than a cached video, then the most engaging tiles of the new video are cached on the MEC server. The cache replacement algorithm uses the pre-trained CNN model to identify the most engaging tiles from the video. This CNN model is trained using the video's saliency map and exploits the correlation between the saliency map and FoV in the frames. The cache replacement algorithm stores these tiles of the video in the cache after the replacement. Details of the different modules in the proposed solution are as follows.

5.1. LSTM-based popularity prediction

Recurrent Neural Networks (RNNs) have been effective for time series data forecasting (Ergen and Kozat, 2018) and can be used to predict video popularity over time. However, RNNs suffer from vanishing gradients problem, which hampers their ability to learn long data sequences (Vural et al., 2021). When the gradient, which carries the RNN parameter update information, becomes very small, the parameter updates become insignificant, which means no real learning is done. LSTMs are a special kind of RNNs that are explicitly designed to avoid the long-term dependency problem (Hochreiter and Schmidhuber, 1997) and can remember the information for long periods. Inspired by Narayanan et al. (2019), we use an LSTM network to estimate the future popularity of videos. The prediction is made for the currently requested and cached videos. As shown in Fig. 5, a request vector $\vec{f^v}$, representing the number of requests $[f_{t-h}^v, f_{t-(h-1)}^v, \dots, f_t^v]$ in previous $(h-1)$ time slots along with the current time slot t, is provided as input to the LSTM network to estimate the video popularity in the $(t+1)$ time slot. Here, f_t^v is a scalar value that represents the number of requests for video $v \in \mathcal{V}$ in tth time slot. Thus, LSTM uses the requests in previous h time slots with L requests in each slot to predict the request in the $(t+1)th$ time slot. The LSTM is initially pre-trained offline (warm-up phase) with historic data profiles using the backpropagation through time method weight initialization. The trained LSTM model is used for the popularity prediction of the cached and latest videos. The cache replacement algorithm makes use of trained the LSTM model to make the farsighted caching decision.

The LSTM network comprises four layers: the input layer, two hidden layers, and one output layer. The input layer gets a 3D tensor as input of shape $[samples, time-steps, 1]$. Each of the two hidden layers is an LSTM layer with 128 LSTM cells. The output layer is a dense layer comprised of 1 unit with a linear activation function.

5.2. CNN model for tile classification

Although MEC brings the cloud computing functions to the edge of the network, it is constrained by resources. The size of the 360° video is extensive, and all the videos cannot be cached on the MEC server. As discussed in Section 3, a user views only a portion of the video at a time in 360° video streaming, and viewing patterns among the users have a substantial correlation. Therefore, some tiles in the tiled 360° video streaming are requested more than the others. Identifying and caching only these tiles can significantly improve system efficiency. Some existing works (Maniotis and Thomos, 2021; Maniotis and Thomos, 2021) use the previous playback data to identify these tiles, but this data is not always available. Therefore, in this work, we use a different approach by utilizing saliency maps to identify these tiles without using the previous playback data. The saliency map identifies the important pixels which contain the prominent details in the frame that might attract the users' attention. Given the saliency map of a tile and an accurate classification model, it can be determined whether the tile should be cached or not to achieve to higher hit rate.

Convolutional Neural Network (CNN/ConvNet) is widely used in image classification (Bhandare et al., 2016). ConvNet encodes the image-specific features into the network architecture, making it suitable for image-based feature learning (LeCun et al., 1995). Unlike classical models, CNNs take image data, train the model, and classify the features automatically for better classification. There are three principal components of CNNs: convolution layer, pooling layer, and fully connected layers. Convolutional layers are made from several feature maps, and each unit of feature maps is made from convolving a small region in input data, which is called the local receptive field. The convolutional layer is sometimes called the feature extractor layer because features of the image are extracted within this layer. Pooling layers are commonly used immediately after convolutional layers. These layers are generated to simplify the information and reduce the scale of feature maps. In other words, pooling layers make a condensed feature map from each feature map in convolutional layers. Fully connected layers are the final layers in the CNN structure that can be one or more layers and placed after a sequence of convolution and pooling layers. This part of CNN comprises the composite and aggregates of the most important information from all procedures of CNN. Consequently, these layers provide the feature vector for the input data, which can be used for machine learning tasks such as classification and prediction (LeCun et al., 1995). The last layer of fully connected layers is known as soft-max classifier and determines the probability of each class label over N classes (LeCun et al., 1995; Bhandare et al., 2016). Fig. 5 illustrates the CNN architecture used in this work. The proposed solution uses a CNN with six convolutions layers and two max-pooling layers with a dropout of 0.25 and 0.5, applied after the third and fifth convolution layers. The fully connected binary output layer is placed after the sixth layer through max pooling. The CNN model is trained to identify the most engaging tiles through binary classification. The trained CNN model takes a saliency map as input and determines the most engaging tiles which should be cached.

5.3. Cache algorithm for tiles 360° video streaming

In this paper, we consider that each base station is deployed with a cache server having the caching capacity $Z \geq 0$. We assume that U users, $\mathcal{U} = \{1, \dots, u, \dots, U\}$ connected to a base station, are served by the cache server on the MEC deployed at that base station. The users request 360° video files from a content catalog of $V = |\mathcal{V}|$ files, with $\mathcal{V} = \{1, \dots, v, \dots, V\}$ being the set of 360° videos. In tile-based 360° videos streaming, videos are generally divided into several tiles that can be encoded and decoded independently. In tilling, a 360° video is spatially divided into $|\mathcal{M}|\times|\mathcal{N}|$ tiles, where $\mathcal{M} = \{1, 2, \dots, m, \dots |\mathcal{M}|\}$ and $\mathcal{N} = \{1, 2, \dots, m, \dots |\mathcal{N}|\}$, and $|\mathcal{M}|$ and $|\mathcal{N}|$ are the set of tiles in a column (vertical dimension) and row (horizontal dimension) of each video segment, respectively. We assume that time is slotted in T time slots and in each time slot $t \in \mathcal{T}$, the request of a user $u \in \mathcal{U}$ for the tiles of video $v \in \mathcal{V}$ is denoted by $r^t_{u,v}$. All the cached videos on the MEC server are defined by the set C. For each cached video v, the set of cached tiles is defined by $(C_v, \forall v \in \mathcal{V})$ and if a video v' is not cached, then set $C_{v'} = \emptyset$.

Algorithm 1 outline the steps followed in the proposed solution. The caching algorithm uses pre-trained LSTM and CNN models to predict future video requests and classify the tiles. On a user request $r^t_{u,v}$ for a video v in time slot t, the MEC server check for the video in the cache server. As stated in line-4 of the Algorithm 1, if video v is cached on the server, then cached tiles from the video request $({}^t_{u,v} \bigcap C_v)$ are transmitted to the user while any missed tiles $({}^t_{u,v} - C_v)$ are fetched from the server to serve the user. To make the cache replacement decision, future requests $(f^{t+1}_{v'}, \forall v' \in C)$ for all the cached videos and the presently requested video (f^{t+1}_v) are estimated. A cached video (v') with the least number of predicted requests is replaced if its future demand is lower than the present video (v) else, the incoming video is not cached. Since the MEC servers are resource-constrained and users do not watch the whole view of 360° video, caching all the tiles of the video results in inefficient resource utilization. Therefore, the proposed method uses a CNN model to identify $(CNN(S_v) \rightarrow \mathcal{I}_v)$ the most engaging tiles I_v of the video using the saliency map S_v. The CNN is trained using the saliency maps of video to identify such tiles. Once the tiles are classified, most engaging tiles $(\mathcal{I}_v)$ are cached on the MEC server by replacing the tiles of video with the least future requests. The implementation details and training procedures of CNN

Algorithm 1: Caching algorithm for tiled 360° video streaming

```
Input: Pre-trained LSTM model with previous request data for
       popularity prediction;
Pre-trained CNN model using the saliency map for tiles'
classification;
Set of all cached videos $C$; Set of tiles $(C_v, \forall v \in \mathcal{V})$ for all
cached videos;
A set of requested tiles $r^t_{u,v}$ by the user $u$ for video $v$ in time slot
$t$;
Saliency map $(S_v, \forall v \in \mathcal{V})$ of the videos;
Output: Updated set of cached videos $C$ and tiles $C_v, \forall v \in \mathcal{V}$
1  for each time slot $t$ do
2    for each user $u$ do
3      for each request $r^t_{u,v}$ do
4        if $v \in C$ then          /* if video is cached */
5          serve the cached tiles $(r^t_{u,v} \bigcap C_v)$ from cache ;
            /* for cached tiles */
6          fetch missed tiles $(r^t_{u,v} - C_v)$ from the content
            server ;            /* for missed tiles */
7        else              /* if video is not cached */
8          fetch video $v$ from CDN and serve the user;
9          $f^{v'}_{t+1} \leftarrow LSTM(\vec{f_{v'}}), \forall v' \in C$ ;      /* predict
            future requests of cached videos */
10         $f^{v}_{t+1} \leftarrow LSTM(\vec{f_v})$ ;      /* predict future
            requests of requested video */
11         if $f^v_{(t+1)} > min(f^{v'}_{t+1} : v' \in C)$ then        /* if
            requested video expected to have
            more requests */
12           $\mathcal{I}_v \leftarrow CNN(S_v)$ ;  /* identify the most
              engaging tiles using CNN */
13           Replace cached tiles of $v'$ with $\mathcal{I}_v$ ;
              /* Cache replacement */
14         else
15           No replacement;
16         end
17       end
18     end
19   end
20 end
```

and LSTM are discussed in the next section. Considering that the trained CNN and LSTM models take constant time $(O(1))$ for classification and prediction. For each uncached video request, line 11 in Algorithm 1 takes $O(n)$ time, in the worst-case scenario, to find the cached item with minimum expected requests. Thus, the worst-case time complexity of the proposed algorithm is $O(n)$.

6. Results and analysis

The simulation environment is developed in Python using open-source libraries for machine learning. For simulation, we consider a single cell cellular network where users are connected to a base station equipped with a MEC server. While we consider a single cell network to demonstrate the efficacy of the proposed work, this work can be easily extended for a multi-cell scenario, where MEC servers collaborate, without any significant change. Furthermore, tiles are deemed to be cached in the highest available quality while different user requests can be fulfilled through transcoding at the MEC server itself (Lu et al., 2019; Shi et al., 2020).

We use an open-source dataset (Lo et al., 2017) for 360° video streaming simulations. A summary of this dataset is discussed in Section 3. In the current section, we highlight some details which are

Table 1
Accuracy of CNN classification corresponding to different tile sizes.

Tile size	AUROC	Validation accuracy
3 × 3	0.9002	0.8056
5 × 5	0.9425	0.8871
8 × 8	0.9418	0.8947
10 × 10	0.9485	0.9105
10 × 15	0.9271	0.8910
10 × 20	0.9572	0.8852

Table 2
Training loss for LSTM corresponding to different hyperparameter settings.

No. neurons	LSTM layers	Dropout	Loss value	Dropout	Loss value
32	1	F	0.000893	T	0.0010
32	2	F	0.000902	T	0.000912
64	1	F	0.000896	T	0.000939
64	2	F	0.000905	T	0.000884
128	1	F	0.000895	T	0.000863
128	2	F	0.000897	T	0.000894

important for implementation. The dataset contains viewing orientation traces of 50 subjects while each subject watched 10 different 360° videos. It comprises 500 traces, each containing the one-minute 360° video playback. These videos are in 30 fps resultantly each trace consists of 1800 data points. Each data point has viewing angles indicated as raw, pitch, and yaw of the user in each frame which is sufficient to identify the FoV. The 360° video dataset also contains the saliency map of all the videos that identify the features in the video frame by signaling out the important pixel. Cumulatively, these pixels help to identify the tiles with more detailed objects that the users will likely perceive.

User requests contain the tiles information in video segments, therefore, one-second video segments are used to generate the user requests. Since videos are shot at 30 fps, there are a total of $(1800 * 10/30) = 600$ video segments in the library to be requested by the users. The user's 360° video requests are generated following the Zipf's popularity (Alexander et al., 1998) distribution which determines the probability of next request to be for ith popular video and defined as $p_i = \frac{i^{-\alpha}}{\sum_{j=1}^{|\mathcal{V}|} j^{-\alpha}}$, where Zipf parameter $\alpha = 0.8$ is used, as in Zink et al. (2009). Using this distribution, 100 requests are generated in each time slot, and simulation runs for the 5000 time slots. The end-to-end latency for fetching the video content is randomly assigned between $[10, 20]$ ms uniformly when tiles are cached on the MEC server and $[100, 200]$ ms for fetching the tile from the origin content server or CDN on the Internet. The 360° video segment size has been used to calculate the saved bandwidth. Different evaluation parameters, compared methods, and implementation and training details of LSTM and CNN are discussed further in this section.

6.1. CNN model training

To realize a content-based FoV classification method, we utilized the saliency map of the video and employed CNN to identify the most engaging tiles in the video segments. The CNN model is considered to be trained on the cloud server with sufficient input data in an offline manner and the trained model is pushed to the edge server. While training process does not require any special hardware and can be done on an MEC server with sufficient training data. To speed up the CNN training process, the 3840 × 1920 resolutions saliency map of the input video is down-sampled four times to 960 × 480 resolution and is used as raw input for the tile classification. There are 1800 frames of saliency maps for each video, and each frame comprises 200 tiles of 192 × 192 pixels each. The output of the CNN model is either 1 or 0, representing a relevant or irrelevant tile for caching, respectively. During training, K-fold cross-validation is performed to verify the robustness of the model. The value of K is set to 5, and eight videos are used for training, while two videos are used for validation. The data is not shuffled before making K-fold to the data division across the videos. In training, input data of 8 videos are shuffled so that the model can robustly learn the patterns from saliency maps of all videos.

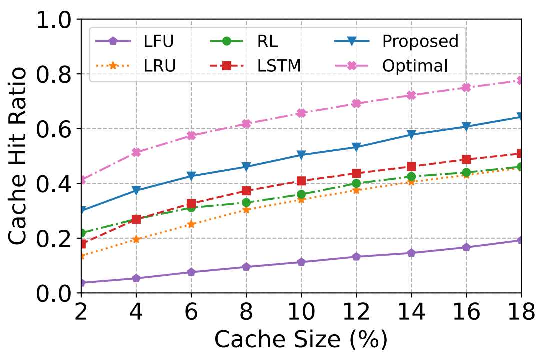
The dataset (Lo et al., 2017) contains the traces for 50 different users, and a tile might be in FoV for one user while in Out-of-Sight (OOS) for another. To overcome this issue while labeling the input data, we use a threshold of 30%, which is determined through empirical results and found to work well. If more than 30% users have watched the tile, it is considered as an engaging tile, otherwise as an irrelevant tile. In the training data, engaging tiles are labeled as 1 while other tiles are labeled as 0. On classification, 23.2% of tiles are classified as engaging tiles. AUROC (Area Under the ROC Curve) is calculated during the training of the model for validation, and the best weights of the model are stored based on maximum validation AUROC. The mean testing AUROC is 0.8791 with a standard deviation of 0.0652. We test the classification method for different tile sizes and results in Table 1 exhibit that the CNN model achieves good accuracy regardless of the tile size.

6.2. Training of LSTM model

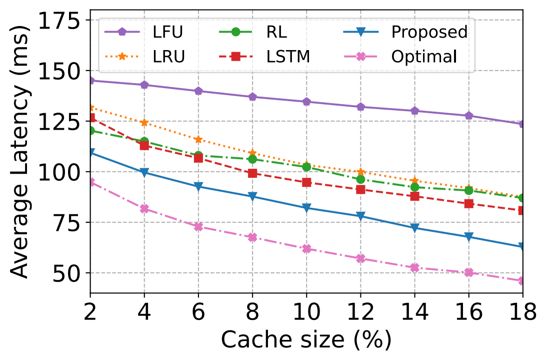
For video popularity prediction, an LSTM network is implemented in Python with the open-source Keras (Keras, 2021) library. First, the LSTM network parameters are tuned through an empirical study, and results are tabulated in Table 2. During training, the LSTM network is initially pre-trained (warm-up phase) with 2000 samples of historic data profiles. Each sample represents the evolution in the popularity of 360° videos over $h = 6$ consecutive time-slots. The LSTM model is pre-trained with a batch size of 100 for 50 epochs, using the Mean Square Error (MSE) loss function between the LSTM's outputs and the actual popularity. The historic data profiles are split into a training set and validation set, where the training set accounts for 80% of the past data and the validation set the rest, 20%. After training the LSTM, the trained weights are used by the prediction module to forecast future content popularity in the online phase.

6.3. Training and processing time of CNN and LSTM models

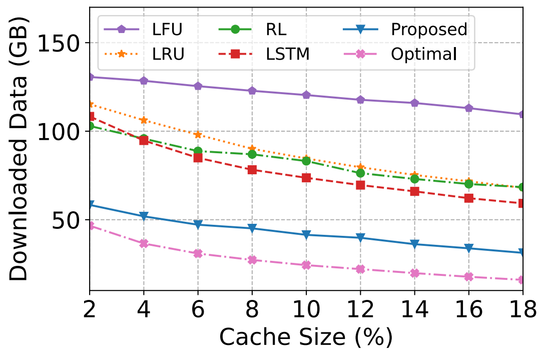
Both the models are trained in an offline manner on a mobile workstation with 8 GB RAM and $Intel\ Core\ i7$ processor without any hardware acceleration. It takes ≈5 h to train the CNN model and training the LSTM model only requires 30 min. Since, both the models are considered to be trained on the cloud server in an offline manner, training time does not matter for our application. On the deployment of trained models, the LSTM model takes 28 ms to make the popularity predictions and the CNN model takes 0.25 s to classify the tiles. While the LSTM model needs to be executed for each uncached video request in the cache replacement algorithm, the CNN model needs to be executed only once for each video, and the results can be stored as a binary map of tiles, representing the most engaging tiles for future reference. In the 360° video caching solution, the runtime of these models does not affect the streaming latency. Because if the video is already cached, there is no need to execute the models, and if the video is not cached, the models are executed alongside while content is served to the user. For model training and deployment, a general-purpose mobile workstation is utilized without any hardware acceleration, which affirms that these models can be trained and deployed on the resource-constrained MEC server without any need for special hardware.



(a) Hit Rate



(b) Latency



(c) Downloaded Data

Fig. 6. Evaluation of hit rate, latency, and backhaul usage with respect to change in cache size.

6.4. Evaluation methodology

The performance of the proposed solution is evaluated on the following metrics, which are crucial for user QoE in 360° video streaming.

- *Cache Hit Rate:* The hit rate is a direct performance metric for content caching and represents the fraction of requests fulfilled by the cache on the MEC servers.
- *Latency:* 360° video streaming has stringent latency requirements; therefore, it is essential to analyze how the proposed solution affects the latency. Here, latency is considered as average end-to-end latency between the user and content server (either cache or main content server) serving the video.
- *Backhaul Usage:* It represents the bandwidth consumed in the backhaul for 360° video streaming. Backhaul usage help in showing the decrease in data transmission for 360° video streaming when edge caching is employed. This metric exhibits the ability of a method to enable the 360° video streaming over the bandwidth constraint cellular networks and indicates the reduction in OPEX of the network operator.

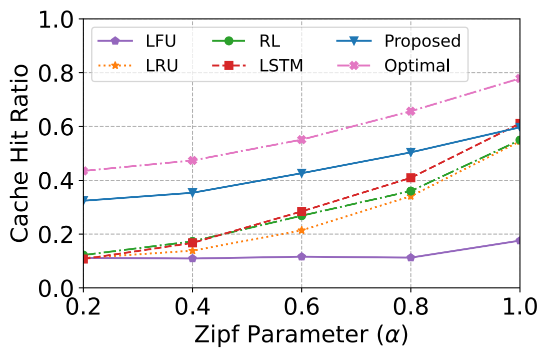
These performance metrics are evaluated for the various values of the following parameters:

- *Cache Size:* MEC is constrained by resources and offers limited storage space. Therefore, it is essential to assess the performance of different cache sizes. In the evaluation results, cache size represents the percentage of video library size that can be cached (e.g., 10% cache size means that the MEC server is able to cache 10% of the video library) on the MEC server.
- *Video Popularity Distribution:* If popularity distribution is highly skewed, even a simple LRU cache replacement algorithm can perform well. But, a robust solution should perform considerably well over a range of popularity distributions. Therefore, the Zipf parameter α has been varied in the simulations to evaluate the performance of the proposed solution for different popularity distributions.

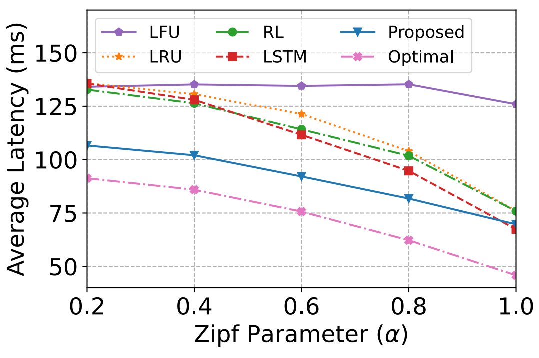
6.5. Scheme under comparison

Based on the above-mentioned metrics and parameters, we have compared the proposed solution with the following methods:

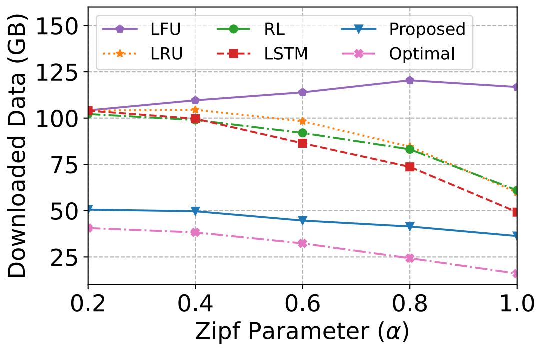
- *Least Recently Used (LRU):* In this scheme, the MEC server associates a timer with all the cached items and updates its value whenever an item is requested. If a requested 360° video is not cached, then all the tiles of the 360° video that were requested least recently are replaced with the tiles of the newly requested video.
- *Least Frequently Used (LFU):* In LFU, the cache server keeps track of the number of requests for each cached 360° video, and all the tiles of the 360° video are cached on the MEC server. When a requested 360° video is not cached at the MEC server, the video requested the least amount of time is replaced with the new video. If the requested 360° video is cached at the MEC server, then the corresponding frequency counter is updated.
- *RL-based Caching Solution:* Reinforcement Learning has been used for legacy and 360° video caching (Zhang et al., 2019; Maniotis and Thomos, 2021). In these methods, RL is used to make the cache replacement decision. The number of requests, time of the last request is associated with each cached 360° video and provided as state variables to the RL model. When a 360° video request arrives, which is not cached at the MEC server, the RL model determines whether the new video should be cached or not. In case of replacement, it also specifies which one of



(a) Hit Rate



(b) Latency



(c) Downloaded Data

Fig. 7. Evaluation of hit rate, latency, and backhaul usage for various Zipf's popularity distributions.

the cached videos should be replaced with the new video. We implemented a state-of-the-art Asynchronous Advantage Actor Critic (A3C) algorithm for the result comparison.

- *LSTM-based Caching Solution:* This solution uses LSTM to predict the future request of the cached videos (Narayanan et al., 2019). When a 360° video request arrives that is not cached, cache replacement is done based on the future expectation of the video requests. If the expected number of future requests of new 360° video is higher than the cached video with the least number of expected future requests, then replacement takes place; otherwise, the user is served from the Internet without caching at the MEC server.
- *Optimal Solution:* For the optimal solution, video popularity and tiles' viewing probability is known in advance. The tiles with highest joint probability, calculated using video popularity and tiles' viewing probability, are used to cache the tiles at the MEC server.

6.6. Results and analysis

6.6.1. Effect of change in cache size

Results in Fig. 6a depict that with the increase in cache size, the hit rate improves owing to the possibility of caching a higher number of videos in the MEC server with larger cache storage. The proposed method improves the cache hit rate by 10% as compared to LSTM based caching and widely outperform the other methods. The performance of the proposed solution is closest to the optimal results, with a gap of $\sim 8\%$. Interestingly, the RL-based caching method performs better than the LSTM-based 360° video caching when the cache size is smaller than 4%, but LSTM surpasses the RL-based method when the cache size is larger than 4%. Moreover, the performance of the RL-based method is inconsistent across the different cache sizes. A possible explanation for this might be that a separate RL model is trained for each cache size due to different action spaces, and the performance of these models is not similar. The LFU performs inadequately among all the tested methods and was found unsuitable for the 360° video caching.

6.6.2. Improvement in network latency

As illustrated in Fig. 6b, the proposed method improves the end-to-end latency by at least 15 ms over the other solutions, which is very significant for 360° video streaming. The performance of LFU is the worst among all evaluated methods, and its latency is more than 120 ms in most cases, which is not suitable for 360° video streaming. As cache size increases, more requests can be served from the MEC server at the network edge with very low latency. Therefore, latency decreases with the increase in the cache size. To evaluate the effect of popularity distribution on latency, we assess the proposed method for various popularity distributions. To achieve that, we increase the value of Zipfs' parameter α in steps from 0.2 to 1.0 while keeping a constant cache size of 10%. Fig. 7b presents the results obtained from this analysis which conveys that as the popularity distribution gets more skewed, on the increase in the Zipfs' parameter α, the latency of all the methods decreases. For $\alpha = 1$, LSTM performs correspondingly well to the proposed method.

6.6.3. Reduction in backhaul usage

Data requirement of 360° video streaming can cause backhaul congestion and poses a challenge for cellular networks. Therefore, we analyze the effect of edge caching on backhaul data usage for different cache sizes and popularity distributions. The results in Fig. 6c show that the proposed method reduces data consumption in the backhaul network by at least 35% as compared to any other evaluated method. This reduction in downloaded data from the ISP also reduces the OPEX for the network operators. The increase in cache size positively affects the backhaul load, and it decreases with an increase in cache size. Since a large number of user requests are served from the edge of the network for a bigger cache size, it decreases the data over the backhaul network. As skewness of popularity distribution increases, a decrease in the downloaded data can be noticed in Fig. 7c.

6.6.4. *Effect of change in video popularity distribution*

The value of the Zipfs' parameter α is step-wise varied between 0.2 and 1.0 to analyze the performance of the proposed method for different popularity distributions. The results in Fig. 7 exhibits that as the popularity skewness increases, the performance gap converges among all the evaluated methods except LFU. Moreover, the performance gap between the optimal results and proposed solution widen marginally as the value of Zipfs' parameter is increased. For highly skewed popularity distribution, only a few popular videos get most of the user requests. Therefore, caching the tiles to use the storage space effectively gets less advantageous because the tiles with less salient points also get some views in the highly popular videos. The proposed method provides a 20% improvement in the hit rate for $\alpha = 0.8$, which is close to real-world values. Hence, in real-world settings, caching the tile has a performance advantage over caching the whole 360° video.

Overall, these results indicate that the content-aware method can help overcome the need for the past playback data to identify the most engaging tiles by employing a CNN-based classification method using video saliency maps. Moreover, the proposed deep learning-based method performs consistently well for different cache sizes and popularity distributions.

7. Conclusion

This work presents a caching solution for tiled 360° video streaming. The proposed solution caches specific tiles instead of the whole 360° video to optimally utilize the MEC resources. The existing solutions require past playback data to estimate the tiles' viewing probabilities for tiled 360° video caching. In contrast, the proposed solution uses deep learning tools to identify the most engaging tiles in 360° video and predict the video popularity. Specifically, LSTM is used to estimate the future demand for the 360° video content, and a CNN model is employed to identify the most engaging tiles for caching. The LSTM model uses the video requests in previous time slots to predict future popularity. The CNN model is trained to identify the most engaging tiles based on the video content itself, namely the video saliency map. This work demonstrates that in absence of the previous playback data content features can be exploited to identify the tiles which can maximize the cache hit ratio. The proposed framework is extensively evaluated through simulations with real-world head movement traces. Results assert that the proposed solution improves the cache hit rate by 10% and substantially reduces the backhaul usage by at least 35%. Moreover, the proposed solution significantly reduces the end-to-end latency, which is critical for user QoE in 360° video streaming. This work focuses on the identification of the most engaging tiles for 360° video caching, in the future, it will be interesting to explore the collaborative caching among the multiple MEC servers with transcoding and joint consideration of network and computation resources.

CRediT authorship contribution statement

Shashwat Kumar: Conceptualization, Methodology, Software, Writing – original draft, Writing – review & editing. **Lalit Bhagat:** Methodology, Software, Writing – original draft, Writing – review & editing. **Antony Franklin A.:** Conceptualization, Validation, Supervision, Writing – review & editing. **Jiong Jin:** Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work is partially supported by the project "Autonomous driving enabling fog computing platform with edge cloud orchestration and edge analytics", funded by Department of Science and Technology, Government of India.

References

Ale, L., Zhang, N., Wu, H., Chen, D., Han, T., 2019. Online proactive caching in mobile edge computing using bidirectional deep recurrent neural network. IEEE Internet Things J. 6 (3), 5520–5530. http://dx.doi.org/10.1109/JIOT.2019.2903245.

Alexander, L., Johnson, R., Weiss, J., 1998. Exploring Zipf's law. Teach. Math. Appl.: Int. J. IMA 17 (4), 155–158. http://dx.doi.org/10.1093/teamat/17.4.155.

Ballard, T., Griwodz, C., Steinmetz, R., Rizk, A., 2019. RATS: Adaptive 360-degree live streaming. In: Proceedings of the 10th ACM Multimedia Systems Conference. In: MMSys '19, pp. 308–311. http://dx.doi.org/10.1145/3304109.3323837.

Bhandare, A., Bhide, M., Gokhale, P., Chandavarkar, R., 2016. Applications of convolutional neural networks. Int. J. Comput. Sci. Inf. Technol. 7 (5), 2206–2215.

Brownlee, J., 2018. How to develop LSTM models for time series forecasting. https://machinelearningmastery.com/how-to-develop-lstm-models-for-time-series-forecasting/, Accessed February 9, 2022.

Carlsson, N., Eager, D., 2020. Had you looked where i'm looking? Cross-user similarities in viewing behavior for 360-degree video and caching implications. In: Proceedings of the ACM/SPEC International Conference on Performance Engineering. In: ICPE '20, Association for Computing Machinery, pp. 130–137. http://dx.doi.org/10.1145/3358960.3379129.

Chakareski, J., 2017. VR/AR Immersive Communication: Caching, Edge Computing, and Transmission Trade-Offs. In: VR/AR Network '17, Association for Computing Machinery, New York, NY, USA, pp. 36–41. http://dx.doi.org/10.1145/3097895.3097902.

Chakareski, J., 2020. Viewport-adaptive scalable multi-user virtual reality mobile-edge streaming. IEEE Trans. Image Process. 29, 6330–6342. http://dx.doi.org/10.1109/TIP.2020.2986547.

Cheng, Q., Shan, H., Zhuang, W., Yu, L., Zhang, Z., Quek, T.Q.S., 2021. Design and analysis of MEC- and proactive caching-based 360 mobile VR video streaming. IEEE Trans. Multimed. http://dx.doi.org/10.1109/TMM.2021.3067205.

Cornia, M., Baraldi, L., Serra, G., Cucchiara, R., 2016. A deep multi-level network for saliency prediction. In: 2016 23rd International Conference on Pattern Recognition (ICPR). pp. 3488–3493. http://dx.doi.org/10.1109/ICPR.2016.7900174.

Dai, J., Zhang, Z., Mao, S., Liu, D., 2020. A view synthesis-based 360°VR caching system over MEC-enabled C-RAN. 30 (10), 3843–3855. http://dx.doi.org/10.1109/TCSVT.2019.2946755.

Elazhary, H., 2019. Internet of Things (IoT), mobile cloud, cloudlet, mobile IoT, IoT cloud, fog, mobile edge, and edge emerging computing paradigms: Disambiguation and research directions. J. Netw. Comput. Appl. 128, 105–140. http://dx.doi.org/10.1016/j.jnca.2018.10.021.

Ergen, T., Kozat, S.S., 2018. Online training of LSTM networks in distributed systems for variable length data sequences. IEEE Trans. Neural Netw. Learn. Syst. 29 (10), 5159–5165. http://dx.doi.org/10.1109/TNNLS.2017.2770179.

ETSI, 2014. Mobile-Edge Computing - Introductory Technical White Paper.

ETSI, 2018. Mobile Edge Computing (MEC); MEC in 5G networks.

Fan, C., Yen, S., Huang, C., Hsu, C., 2020. Optimizing fixation prediction using recurrent neural networks for 360° video streaming in head-mounted virtual reality. IEEE Trans. Multimed. 22 (3), 744–759.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Hu, M., Luo, X., Chen, J., Lee, Y.C., Zhou, Y., Wu, D., 2021. Virtual reality: A survey of enabling technologies and its applications in IoT. J. Netw. Comput. Appl. 178, 102970. http://dx.doi.org/10.1016/j.jnca.2020.102970.

Huawei, 2016. Whitepaper on the VR-oriented bearer network requirement. https://www-file.huawei.com/~/media/CORPORATE/PDF/white%20paper/whitepaper-on-the-vr-oriented-bearer-network-requirement-en.pdf.

Keras, 2021. Keras API reference - LSTM layer. https://keras.io/api/layers/recurrent_layers/lstm/, Accessed on February 9, 2022.

LeCun, Y., Bengio, Y., et al., 1995. Convolutional networks for images, speech, and time series. In: The Handbook of Brain Theory and Neural Networks, Vol. 3361. p. 1995, (10).

Liu, Y., Liu, J., Argyriou, A., Wang, L., Xu, Z., 2021. Rendering-aware VR video caching over multi-cell MEC networks. IEEE Trans. Veh. Technol. 70 (3), 2728–2742. http://dx.doi.org/10.1109/TVT.2021.3057684.

Lo, W.-C., Fan, C.-L., Lee, J., Huang, C.-Y., Chen, K.-T., Hsu, C.-H., 2017. 360° Video viewing dataset in head-mounted virtual reality. In: Proceedings of the 8th ACM on Multimedia Systems Conference. In: MMSys'17, pp. 211–216. http://dx.doi.org/10.1145/3083187.3083219.

Lu, Q., Li, C., Zou, J., Tang, K., Wang, Q., Xiong, H., 2019. Transcoding-enabled edge caching and delivery for tile-based adaptive 360-degree video streaming. http://dx.doi.org/10.1109/VCIP47243.2019.8965938.

Mahzari, A., Taghavi Nasrabadi, A., Samiei, A., Prakash, R., 2018-10. FoV-Aware edge caching for adaptive 360° video streaming. In: Proceedings of the 26th ACM International Conference on Multimedia. ACM, pp. 173–181. http://dx.doi.org/10.1145/3240508.3240680.

Maniotis, P., Bourtsoulatze, E., Thomos, N., 2020. Tile-based joint caching and delivery of 360° videos in heterogeneous networks. IEEE Trans. Multimed. 22 (9), 2382–2395. http://dx.doi.org/10.1109/TMM.2019.2957993.

Maniotis, P., Thomos, N., 2021. Viewport-aware deep reinforcement learning approach for 360° video caching. 9210 (c), 1–14. http://dx.doi.org/10.1109/TMM.2021.3052339.

Maniotis, P., Thomos, N., 2021. Tile-based edge caching for 360° live video streaming. IEEE Trans. Circuits Syst. Video Technol. 1. http://dx.doi.org/10.1109/TCSVT.2021.3055985.

McGarry, C., 2021. 5G speed: 5G vs 4G performance compared. https://www.tomsguide.com/features/5g-vs-4g, Accessed on February 9, 2022.

Mordor Intelligence, 2020. Virtual reality (VR) market - growth, trends, COVID-19 impact, and forecasts (2021 - 2026). https://www.mordorintelligence.com/industry-reports/virtual-reality-market/.

Narayanan, A., Verma, S., Ramadan, E., Babaie, P., Zhang, Z.-L., 2019. Making content caching policies 'smart' using the deepcache framework. SIGCOMM Comput. Commun. Rev. 48 (5), 64–69. http://dx.doi.org/10.1145/3310165.3310174.

Nguyen, T.-V., Dao, N.-N., Tuong, V.D., Noh, W., Cho, S., 2021. User-aware and flexible proactive caching using LSTM and ensemble learning in IoT-MEC networks. IEEE Internet Things J. http://dx.doi.org/10.1109/JIOT.2021.3097768.

Pang, H., Liu, J., Fan, X., Sun, L., 2018. Toward smart and cooperative edge caching for 5G networks: A deep learning based approach. In: 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS). pp. 1–6. http://dx.doi.org/10.1109/IWQoS.2018.8624176.

Papaioannou, G., Koutsopoulos, I., 2019. Tile-based caching optimization for 360° videos. pp. 171–180. http://dx.doi.org/10.1145/3323679.3326515.

Poularakis, K., Iosifidis, G., Argyriou, A., Koutsopoulos, I., Tassiulas, L., 2019. Distributed caching algorithms in the realm of layered video streaming. IEEE Trans. Mob. Comput. 18 (4), 757–770. http://dx.doi.org/10.1109/TMC.2018.2850818.

Qian, F., Han, B., Xiao, Q., Gopalakrishnan, V., 2018. Flare: Practical viewport-adaptive 360-degree video streaming for mobile devices. In: Proceedings of the 24th Annual International Conference on Mobile Computing and Networking. In: MobiCom '18, pp. 99–114. http://dx.doi.org/10.1145/3241539.3241565.

Shi, J., Pu, L., Xu, J., 2020. Allies: Tile-based joint transcoding, delivery and caching of 360° videos in edge cloud networks. 2020-October, 337–344. http://dx.doi.org/10.1109/CLOUD49709.2020.00054.

Song, Y., Zhao, Y., Li, C., 2019. A user behavior aware immersive video caching algorithm. In: 2019 11th International Conference on Wireless Communications and Signal Processing (WCSP). pp. 1–6. http://dx.doi.org/10.1109/WCSP.2019.8927955.

Sukhmani, S., Sadeghi, M., Erol-Kantarci, M., El Saddik, A., 2019. Edge caching and computing in 5G for mobile AR/VR and tactile internet. IEEE MultiMedia 26 (1), 21–30. http://dx.doi.org/10.1109/MMUL.2018.2879591.

Sun, Y., Chen, Z., Tao, M., Liu, H., 2019. Communications, caching, and computing for mobile virtual reality: Modeling and tradeoff. IEEE Trans. Commun. 67 (11), 7573–7586. http://dx.doi.org/10.1109/TCOMM.2019.2920594.

Vural, N.M., Ergüt, S., Kozat, S.S., 2021. An efficient and effective second-order training algorithm for LSTM-based adaptive learning. IEEE Trans. Signal Process. 69, 2541–2554. http://dx.doi.org/10.1109/TSP.2021.3071566.

Wang, C., He, Y., Yu, F.R., Chen, Q., Tang, L., 2018. Integration of networking, caching, and computing in wireless systems: A survey, some research issues, and challenges. IEEE Commun. Surv. Tutor. 20 (1), 7–38. http://dx.doi.org/10.1109/COMST.2017.2758763.

Wang, F., Wang, F., Liu, J., Shea, R., Sun, L., 2020. Intelligent video caching at network edge: A multi-agent deep reinforcement learning approach. 2020-July, 2499–2508. http://dx.doi.org/10.1109/INFOCOM41043.2020.9155373.

Xie, L., Xu, Z., Ban, Y., Zhang, X., Guo, Z., 2017. 360ProbDASH: IMproving QoE of 360 video streaming using tile-based HTTP adaptive streaming. In: Proceedings of the 25th ACM International Conference on Multimedia. In: MM '17, pp. 315–323. http://dx.doi.org/10.1145/3123266.3123291.

Zhang, Y., Zhao, P., Bian, K., Liu, Y., Song, L., Li, X., 2019. DRL360: 360-degree video streaming with deep reinforcement learning. In: IEEE INFOCOM 2019 - IEEE Conference on Computer Communications. pp. 1252–1260.

Zhu, R., Tu, X., Xiangji Huang, J., 2020. Chapter seven - Deep learning on information retrieval and its applications. In: Das, H., Pradhan, C., Dey, N. (Eds.), Deep Learning for Data Analytics. Academic Press, pp. 125–153. http://dx.doi.org/10.1016/B978-0-12-819764-6.00008-9.

Zink, M., Suh, K., Gu, Y., Kurose, J., 2009. Characteristics of YouTube network traffic at a campus network – Measurements, models, and implications. Comput. Netw. 53 (4), 501–514.

Shashwat Kumar received his B.Tech. degree in Information Technology from Uttar Pradesh Technical University, India, in 2011 and the M.Tech. degree in Computer Science and Engineering from the Sardar Vallabhbhai National Institute of Technology, Surat, India, in 2013. He completed his Ph.D. degree in Computer Science and Engineering at the Indian Institute of Technology Hyderabad (IITH), India, in 2022. His current research interests include Multi-access Edge Computing (MEC), Mobile Networks (5G and Beyond 5G), Multimedia Services Optimization, Deep Learning, and Reinforcement Learning.

Lalit Bhagat received his B.Tech. degree in Computer Science and Engineering from Jaypee Institute of Information Technology (JIIT), Noida, India, in 2021. He is currently working toward his M.S. degree in Computer Science at the University of California, Los Angeles (UCLA), USA. His current research interests include Machine Learning and Deep Learning focusing on Computer Vision and Reinforcement Learning.

Dr. Antony Franklin A. received his B.E. degree in Electronics and Communication Engineering from Madurai Kamaraj University, India, in 2000 and M.E. degree in Computer Science and Engineering from Anna University, India, in 2002. He received his Ph.D. degree in Computer Science and Engineering from the Indian Institute of Technology Madras, India, in 2010. He is currently working as an Associate Professor at Indian Institute of Technology Hyderabad (IITH), India. Before joining IITH, he worked as a Senior Engineer at DMC R&D Center, Samsung Electronics, South Korea between 2012 and 2015 where he was involved in the development of 5G networking technologies. He also worked as a Research Engineer in Electronics and Telecommunications Research Institute (ETRI), South Korea between 2010 and 2012 where he was involved in Cognitive Radio Technology research. He is co-recipient of Best Paper Award at COMSNETS 2022, Best Academic Demo Award at COMSNETS 2018, and 2nd Best Paper Award at IEEE ANTS 2017 conferences. His current research interests include Mobile Networks (5G and Beyond 5G), Cloud Radio Access Networks (C-RAN), Mobile Edge Computing (MEC), Internet of Things (IoT), and SDN/NFV. He served as a TPC co-chair of National Conference on Communications (NCC) 2018 and COMSNETS 2019 (Poster Session) conferences and General co-chair of the workshop "5G from theory to practice" in 5G World Forum 2020. He is a senior member of IEEE and a member of ACM.

Jiong Jin (M'11) received the B.E. degree with First Class Honours in Computer Engineering from Nanyang Technological University, Singapore, in 2006, and the Ph.D. degree in Electrical and Electronic Engineering from the University of Melbourne, Australia, in 2011. From 2011 to 2013, he was a Research Fellow in the Department of Electrical and Electronic Engineering at the University of Melbourne. He is currently an Associate Professor in the School of Science, Computing and Engineering Technologies, Swinburne University of Technology, Melbourne, Australia. His research interests include network design and optimization, edge computing and networking, robotics and automation, and cyber-physical systems and Internet of Things as well as their applications in smart manufacturing, smart transportation and smart cities.