Smart caching for live 360^o video streaming in mobile networks

Pantelis Maniotis
CSEE Department
University of Essex
Colchester, United Kingdom
p.maniotis@essex.ac.uk

Nikolaos Thomos
CSEE Department
University of Essex
Colchester, United Kingdom
nthomos@essex.ac.uk

***Abstract*—Despite the advances of 5G systems, the delivery of 360^o video content in mobile networks remains challenging because of the size of 360^o video files. Recently, edge caching has been shown to bring large performance gains to 360^o Video on Demand (VoD) delivery systems, however existing systems cannot be straightforwardly applied to live 360^o video streaming. To address this issue, we investigate edge cache-assisted live 360^o video streaming. As videos' and tiles' popularities vary with time, our framework employs a Long Short-Term Memory (LSTM) network to determine the optimal cache placement/evictions strategies that optimize the quality of the videos rendered by the users. To further enhance the delivered video quality, users located in the overlap of the coverage areas of multiple SBSs are allowed to receive their data from any of these SBSs. We evaluate and compare the performance of our method with that of state-of-the-art systems. The results show the superiority of the proposed method against its counterparts, and make clear the benefits of accurate tiles' popularity prediction by the LSTM networks and users association with multiple SBSs in terms of the delivered quality.**

***Index Terms*—Tile-encoding, live 360^o video streaming, edge-caching, LSTM networks.**

I. Introduction

Virtual Reality (VR) and 360^o video are becoming increasingly popular technologies on commercial live streaming platforms such as YouTube Live, Facebook Live, Twitch, Periscope, and Meerkat. To provide users an immersive experience, 360^o videos may be viewed with the help of Head Mounted Displays (HMDs) [1]. In HMDs, each 360^o scene is projected in the internal part of a spherical surface [2]. A user wearing an HMD views only a Field of View (FoV) of the spherical scene, known as viewport (see Fig. 1). To prevent users from experiencing motion sickness, the response of the mobile network to the users' head movements should be as fast as the HMD refresh rate [3], which is commonly 120Hz. This challenges mobile networks that need to deliver the requested viewport to the users in less than 10ms.

A remedy for such tight delivery deadlines is to transmit the whole scene, however this is not efficient as it necessitates significant bandwidth resources due to the high resolution of 360^o videos, e.g., 4K, 8K. Further, this strategy leads to significant waste of bandwidth resources since only a part of the 360^o video will be displayed. The problem becomes even more challenging when the network infrastructure must support a large number of users, which may want to access the requested content from various locations. Tile-encoding has been exploited for live 360^o video streaming in [4], in order to facilitate the delivery of only the requested FoV in high quality. The live streaming platform is supported by an architecture that combines RTP and DASH. The results show 50% of bandwidth savings and shed light on the trade-off between video quality and bandwidth usage. Differently from [4], authors in [5] proposed a multicasting system to support live 360^o video streaming. Users with low bandwidth capacities experienced a quality gain of 3 dB, while users with high bandwidth capacities saw a gain of 3.5-4 dBs. A live streaming system based on MPEG media transport (MMT) is proposed in [6] to guarantee the delivery of high quality 360^o videos to the users.

978-1-7281-9320-5/20/$31.00 ©2020 IEEE

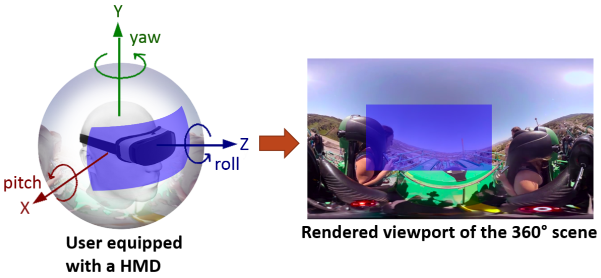


Fig. 1. User equipped with a Head Mounted Display (HMD).

The existing live streaming systems for 360^o video fail to exploit efficiently opportunities to reuse the communicated content in the networks, in particular, when users' requests are asynchronous. Further, some systems require computation of multicast trees, which are difficult to maintain. A more efficient method is to use edge caching infrastructure, which has been previously used to support standard video [7], and 360^o video [8]–[10] delivery. The advantages of using edge caches to store 360^o videos are demonstrated in [8], [9] where decisions are made regarding where to cache a content and where to retrieve it from on per tile and per layer basis. Similarly to [9], in [10] a tile-based caching scheme is proposed which investigates the efficiency of cache systems storing 360^o videos encoded at different resolutions, and at multiple video layers. These works showed that offline edge caching is a prominent solution for 360^o Video on Demand (VoD) systems. However, [9], [10] address a VoD scenario and assumate that the content popularity profile is known in advance, which is not the case in live streaming scenarios where the content popularity changes dynamically and may not be known a priori, or the estimated distribution may not be accurate. Further, the solutions presented in [8]–[10] do not scale well with big content because of the cache optimization complexity. This naturally calls for edge caching assisted live streaming systems that exploit tiles encoding and can estimate future content popularity trends, while preserving low complexity and scalability with respect to the number of 360^o video files.

In this paper, we propose a novel caching framework to support live streaming for 360^o video. To the best of our knowledge, this is the first edge caching scheme for supporting live streaming of 360^o videos. Our system aims at maximizing the overall cache hit ratio, which results in improved rendered video quality, and reduced backhaul links' use. To this aim, our algorithm uses limited observations of the users' requests for the various 360^o videos and viewports in order to decide the optimal cache eviction/placement strategy. Similarly to [9], we exploit encoding of the 360^o videos into multiple quality layers and tiles. Further, we use LSTM networks to determine which 360^o videos will be cached at the base quality and which tiles of the 360^o videos will be cached in high quality. LSTM networks accurately predict the popularity of the 360^o videos and tiles in high quality for the next Group of Pictures (GOP). This permits the content to be prefetched and cached at the SBSs prior to actual requests happening and enables timely delivery of the requested 360^o video files. The results illustrate the superiority of the proposed method against Least Frequently Used (LFU), Least Recently Used (LRU), and First In First Out (FIFO) algorithms, which are commonly used in the literature.

II. System Setup

In this paper, we consider the mobile edge caching architecture shown in Fig. 2. The mobile edge network consists of N Small Base Stations (SBSs), and a Macrocell Base Station (MBS). Let $\mathcal{N} = \{1, \dots n, \dots, N\}$ denote the set of the N SBSs, and $N+1$ represent the MBS. The connection of the MBS to the core network is established through a high capacity backhaul link, while the connection of the SBSs to the core network is established through the MBS using millimeter-wave links. The communication ranges of the SBSs are represented by the set $\mathcal{P} = \{p_1, \dots p_n, \dots, p_N\}$, with p_n being the communication range of the nth SBS. The communication range of the MBS is p_{N+1}. The cache capacity of the $n \in \mathcal{N}$ SBS is denoted by $C_n \geq 0$, $\forall n \in \mathcal{N}$.

The set of users $\mathcal{U} = \{1, \dots u, \dots, U\}$ contains U users in total. Users located in the overlap of the coverage areas of multiple SBSs are associated with any of these SBSs. The primary SBS for each user is the one that has the maximum signal-to-interference-plus-noise ratio (SINR). Users' association with multiple SBSs allows requested tiles that are not cached at the primary SBS, but are stored in the cache of the other SBSs the user resides, to be delivered to the user from these caches. We assume that time is slotted in T time slots. The request of the user $u \in \mathcal{U}$ for GOP $g \in \mathcal{G}$ of a 360^o video $v \in \mathcal{V}$ is denoted by w_u^t. Let $W_u = \{w_u^1, \dots, w_u^t, \dots, w_u^T\}$ be the set which has T consecutive requests from user $u \in \mathcal{U}$.

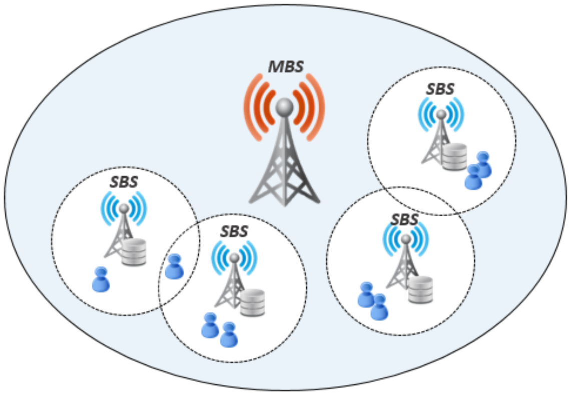


Fig. 2. Considered mobile-edge architecture.

The users request 360^o videos from a content catalogue $\mathcal{V} = \{1, \dots v, \dots, V\}$, where $V = |\mathcal{V}|$ is the total number of 360^o videos. Each video is encoded into M tiles and L quality layers with the first quality layer being the base layer, while the rest $L-1$ layers are the enhancement layers. The acquisition of a tile at the base quality layer offers tile's reconstruction at the lowest quality, while the acquisition of up to the lth quality layer gradually enhances the quality of that tile.

The delay needed to transmit one Mbit from the cache of the nth SBS to a user is denoted by d_n. Similarly, let d_{N+1} be the delay needed to transmit one Mbit that is fetched from the backhaul of the MBS to a user. In order to guarantee the timely delivery of the tiles of each GOP to the users, the following constraint should be met:

$$\sum_{n \in \mathcal{N}_B} \sum_{l \in \mathcal{L}} \sum_{m \in \mathcal{M}} o_{vglm} \cdot d_n \cdot y_{vglm}^n \leq t_{disp}, \forall v \in \mathcal{V}, g \in \mathcal{G} \quad (1)$$

where o_{vglm} stands for the size (in Mbits) of the mth tile of the lth quality layer of the gth GOP of the vth 360^o video. The variable y_{vglm}^n takes the value 1 when the the mth tile of the lth quality layer of the gth GOP of the vth 360^o video is delivered on time to the uth user from the nth SBS ($n \in \mathcal{N}$) or the MBS ($n = N+1$), and 0 otherwise. Finally, the parameter t_{disp} denotes the duration each GOP is being displayed.

III. System Model

1) Caching Entity (CE): Each SBS in the Fig. 2 is equipped with a CE, as illustrated in Fig. 3. Each caching entity is composed of a number of modules that their operation will be discussed later.

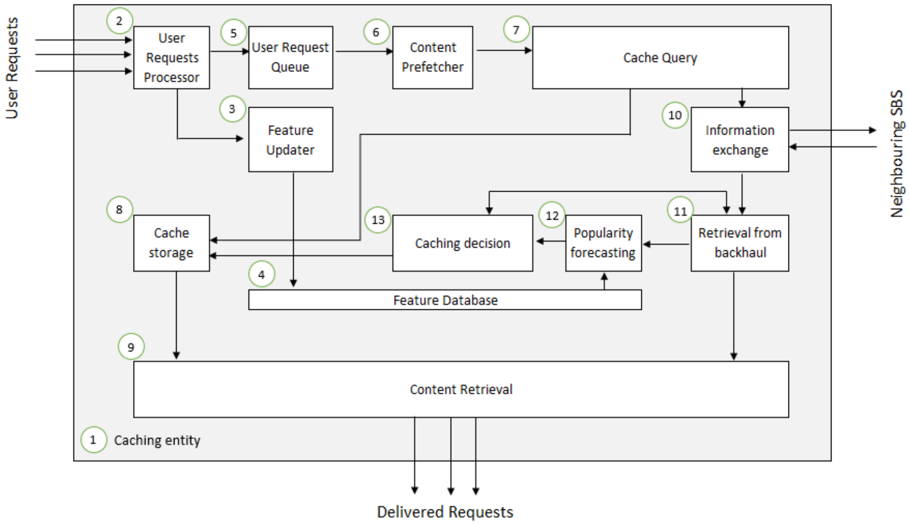


Fig. 3. Caching entity - Request/retrieval process.

2) Users Request Processor (URP): The URP module is responsible for the decomposing of the user requests $w_u^t,\ t \in \mathcal{T},\ \ u \in \mathcal{U}$. Specifically, if a viewport consists of k tiles, each user request w_u^t is decomposed into $k+1$ requests $w_{u,i}^t$. The first request $w_{u,0}^t$ is for receiving all the tiles of the requested 360^o video at the base quality. The rest k requests $\{w_{u,1}^t, \ldots, w_{u,i}^t, \ldots, w_{u,k}^t\}$ are for receiving each of the k tiles of the requested viewport in high quality. The set of requests are given by $\mathcal{W}_u^t = \{w_{u,0}^t, w_{u,1}^t, \ldots, w_{u,i}^t, \ldots, w_{u,k}^t\}$.

3) Feature Updater (FU): The FU module calculates in each time slot $t \in \mathcal{T}$ the features, which represent the number of times each request was encountered. Requests can be either for receiving a 360^o video at the base quality or a tile of a viewport in high quality. These features are transferred to the Feature Database (FD) module, where they are stored.

4) Feature Database (FD): The FD module stores the features computed by the FU module regarding the number of times the various 360^o videos (at base quality) and tiles (in high quality) were requested at an SBS.

5) Users' Requests Queue (URQ): After the decomposition of each user request by the URP module into multiple requests, the decomposed requests are directed to the URQ module, which implements a technique called request coalescing [11]. According to this technique, when multiple users' requests for the same content arrive at an SBS, the first request is prioritized for processing, while the rest of the users' requests are held in a queue for later processing. This mechanism is needed in live streaming scenarios because if a requested content is not cached at the SBS, a cache miss will occur for all the users' requests for that content. This would lead to all that traffic to be directed to the core network.

6) Content Prefetcher (CP): Due to the end-to-end delay, SBSs are not able to respond instantly to the users' head movements. To overcome this problem, the CP module is used. When the CP module receives the requests $\mathcal{W}_u^t = \{w_{u,0}^t, w_{u,1}^t, \ldots, w_{u,i}^t, \ldots, w_{u,k}^t\}$ for the various 360^o videos (in base quality) and tiles (in high quality) from the URP at the time slot t, it decides what content should be the prefetched for the next time slot $t+1$. Let the set $\mathcal{Z}_u^{t+1} = \{z_{u,0}^{t+1}, z_{u,1}^{t+1}, \ldots, z_{u,i}^{t+1}, \ldots, z_{u,k}^{t+1}\}$ denote the content that will be prefetched to the users regarding the time slot $t+1$. To decide which content should be prefetched, we use the Last Sample Replication (LSR) [12] algorithm. According to this algorithm, when the CP module receives a user request $w_{u,i}^t$ at time slot t, the content $z_{u,i}^{t+1}$ that is decided to be prefetched for the time slot $t+1$ is considered to be the same with the request $w_{u,i}^t$. For the sake of simplicity, we assume that for the first time slot, the content that will be prefetched to the users is the requested content.

7) Cache Query (CQ): The CQ module examines whether the content indicated by the CP module is already cached at the SBS. When that content is cached, it is served to the user locally from the Cache Storage (CS). When the content is not available at the SBS, the Information Exchange (IE) module checks whether a neighboring SBS can serve it to the user, provided the user is in the communication range of the neighboring SBS. If the content is cached at a neighboring SBS with which the user is associated, the content is delivered to them from that SBS. Differently, the content is fetched at the SBS through the backhaul of the MBS, and then it is served to the user.

8) Cache Storage (CS): The CS module is responsible for caching the content for the tiles of the 360^o videos. Each SBS has a separate cache storage module with capacity C_n with n being the index of the SBS node.

9) Content Retrieval (CR): The CR module is responsible for the delivery of the content to the users. This module retrieves the content either through: a) the CS module when it is cached at the SBS, and b) the Retrieval from Backhaul (RFB) module that retrieves the content from the backhaul when it is not cached at the SBS.

10) Information Exchange (IE): The IE module is responsible for the communication of an SBS with its neighbouring SBSs. Using this module, SBSs exchange information regarding their cached content, which is used to indicate to the users from where to retrieve their data.

11) Retrieval from Backhaul (RFB): The RFB module is responsible for the retrieval of the content that will be prefetched to caches of the SBSs through the backhaul. After the retrieval of the content by the RFB module, its popularity is predicted by the Popularity Forecasting (PF) module.

12) Popularity Forecasting (PF): The PF module forecasts at each time t the popularity of the content indicated by the CP module (for the time slot $t+1$). The predicted popularity is used to prefetch the caches at the SBSs with content that will be popular in the future. The popularity forecasting by the PF module is performed using the features stored at the FD module regarding the previous $h-1$ time slots and the current time slot $t \in \mathcal{T}$. Let us denote the popularity of the 360^o video $v \in \mathcal{V}$ (base quality) at the SBS $n \in \mathcal{N}$ in the time slot $t \in \mathcal{T}$ as $\lambda^t_{n,v,0}$. Similarly, if $\mathcal{M} = \{1, \ldots, m, \ldots, M\}$ is the set of the M encoded tiles of each 360^o video $v \in \mathcal{V}$, we denote the popularity of the tile $m \in \mathcal{M}$ of the video $v \in \mathcal{V}$ at the SBS $n \in \mathcal{N}$ as $\lambda^t_{n,v,m}$. An LSTM network is used to predict the content popularity evolution. The output of the LSTM module is used by the Caching Decision (CD) module to make caching update decision about the retrieved content.

13) Caching Decision (CD): The CD module decides whether to cache the retrieved content by the RFB module, using the popularities predicted by the PF module. Let us denote the total number of cached 360^o videos (at the SBS $n \in \mathcal{N}$) at the base quality as b_n, and the total number of cached tiles in high quality as f_n. In addition, let the forecast popularities of the cached 360^o videos at the base quality at the time slot $t+1$ to be described by the set $\mathcal{B}^{n,t+1} = \{B^{n,t+1}_1, \ldots, B^{n,t+1}_i, \ldots, B^{n,t+1}_{b_n}\}$, and the forecast popularities of the cached tiles in high quality by the set $\mathcal{F}^{n,t+1} = \{F^{n,t+1}_1, \ldots, F^{n,t+1}_j, \ldots, F^{n,t+1}_{f_n}\}$. When the content that will be prefetched to a user is cached locally or at a neighbouring SBS the user resides, it is delivered to the user from the SBS it is cached. In such case, no decision is made at the CD module. In different case, the content is retrieved at the RFB module, and a decision is made about whether to cache it. Specifically, a decision is initially made about whether to cache the $z^{t+1}_{u,0}$ regarding the 360^o video indicated by the CP module (when it is not cached). If the predicted popularity by the PF module for the $z^{t+1}_{u,0}$ is denoted by $\lambda^{t+1}_{n,v,0}$, the $z^{t+1}_{u,0}$ will be cached at the SBS in the place of the ith cached 360^o video at the base quality if $\lambda^{t+1}_{n,v,0} > B^{n,t+1}_i$ and $\min(\mathcal{B}^{n,t+1}) = B^{n,t+1}_i$. Next, in case the 360^o video is cached at the base quality, a decision is made for each one of the tiles in high quality $\{z^{t+1}_{u,1}, \ldots, z^{t+1}_{u,i}, \ldots, z^{t+1}_{u,k}\}$ that are not cached about whether they should be stored at the CS module. Specifically, if the predicted popularity by the PF module for the tile $z^{t+1}_{u,i}$ is denoted by $\lambda^{t+1}_{n,v,m}$, the tile $z^{t+1}_{u,i}$ will be cached in the place of the jth cached tile in high quality if $\lambda^{t+1}_{n,v,m} > F^{n,t+1}_j$ and $\min(\mathcal{F}^{n,t+1}) = F^{n,t+1}_j$. However, if the decision regarding the request $z^{t+1}_{u,0}$ was not to cache it, no further decision is made, and none of the content requests $\{z^{t+1}_{u,1}, \ldots, z^{t+1}_{u,i}, \ldots, z^{t+1}_{u,k}\}$ regarding the tiles in high quality are examined.

IV. Performance Evaluation

In this section, we evaluate the performance of the proposed framework for deciding the optimal caching/eviction policies for 360^o videos in a live streaming setup. In all examined schemes, users may be associated with multiple SBSs provided they reside in the transmission range of these SBSs, from where they can obtain their data. When a user request arrives at an SBS, the decision regarding whether to update the cached content is made at that SBS regardless if the user obtained their data from a neighboring SBS. This strategy helps popular content to be prefetched to the caches of the SBSs for accommodating future users' requests. Moreover, in order to derive conclusions that are not affected by the request's forecasting mechanism, all the schemes under comparison employ the LSR algorithm similar to the proposed scheme. All the decisions regarding whether to cache a tile and in what quality are made on per GOP basis. In particular, when a request is received for a tile of the gth GOP of a 360^o video, a cache update decision is made for all the tiles of the next GOP of this video. A high-level description of the schemes under comparison is given below:

1) *Least Frequently Used (LFU):* This algorithm replaces content that was requested the least frequently at an SBS with the content indicated by the LSR algorithm.
2) *Least Recently Used (LRU):* This algorithm replaces content that was requested the least recently at an SBS with the content indicated by the LSR algorithm.
3) *First In First Out (FIFO):* This algorithm replaces content that was requested the earliest at an SBS with the content indicated by the LSR algorithm.

Unless otherwise specified, we assume a cellular network that consists of $N=3$ SBSs along with an MBS. The coverage range of each SBS is p_n = 200m, and the coverage range of the MBS is p_{N+1} = 2000m. The cache capacity of the SBSs is set to 10% of the 360^o videos content library. We consider that the total number of users is $U = 540$, randomly placed in the coverage area of the three SBSs. The bit-rate at which data is delivered from the cache of the users' primary SBS is 14 Mbit/sec, while when data is delivered from the cache of an SBS that is not the primary one to a user is 13 Mbit/sec. Finally, when data is retrieved by the backhaul a user can download the data with a rate of 2.9 Mbit/sec. The content library is comprised of $V = 100$ 4K videos. Each 360^o video has a duration of 300 GOPs, with each GOP lasting $t_{disp} = 1$ sec. Each GOP is encoded in $M = 12$ tiles and $L = 2$ quality layers, while the size of each viewport is 4 tiles. The bitrate of the base layer is 2 Mbps, while the bitrate of the enhancement layer is 12 Mbps. The distortion reduction achieved by acquiring a tile at the base quality layer is $\delta_{v,g,1,m} = 30$ dB, while the distortion reduction achieved by obtaining a tile at the enhancement quality layer is $\delta_{v,g,2,m} = 10$ dB.

We assume that the popularity of the 360^o videos through the GOPs is time-varying, and at any given moment, users may drop the 360^o video they are watching to view another one. To this aim, we assume that for the first GOP the users' requests for the various 360^o videos follow the Zipfian distribution [13] with shape parameter $\eta_v = 1$. Hence, the probability of a 360^o video to be selected is given by:

$$p_v = \frac{1/v^{\eta_v}}{\sum_{v \in \mathcal{V}} 1/v^{\eta_v}} \tag{2}$$

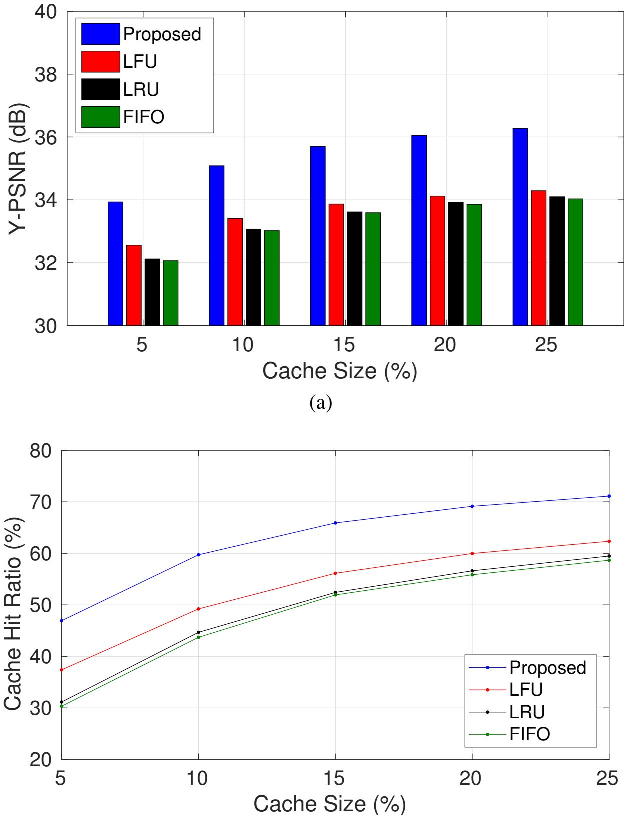
To differentiate the popularity of the 360^o videos through the GOPs, inspired from [14], we assume that the probability of a user to drop watching a 360^o video at the GOP $g \in \{2, \ldots, G\}$ follows the Weibull distribution. Specifically, we assume that each one of the 360^o videos that comprise our content library falls with equal probability to one of the 14 video categories, e.g., News, Sports, etc., presented in [14]. Then, according to the video category each 360^o video falls into, we use the corresponding Weibull distribution parameters presented in [14] to estimate for each user the probability of dropping a 360^o video at GOP $g \in \{2, \ldots, G\}$. For each GOP, when users drop watching a 360^o video, the probability to select another video to watch follows the Zipfian distribution like in (2).

In terms of the viewports' requests, we simulated them with realistic navigation patterns obtained from the dataset described in [15]. To this aim, we initially sampled 10 different videos from this dataset. For each sampled video we obtained 30 trajectories. We mapped with equal probability, each of the $V = 100$ videos that comprise the content library to one of the 10 sampled videos. Then, for each 360^o video request, according to its mapped index, we assigned with equal probability one of the 30 available trajectories for that video.

The LSTM network used for popularity prediction is comprised of four layers. Each hidden layer is dense and has 100 LSTM cells. The hidden layers have as activation function the ReLu function. The input layer is a 3D tensor with shape (samples, time-steps, 1), while the output layer has a single unit. The LSTM network accepts as inputs the features of the current time slot t along with the features computed at the previous nine time slots. The LSTM is initially pre-trained (warm-up phase) with historic data profiles using the backpropagation through time method in order to find a good starting point for its weights. Then, these weights are used for the popularity prediction of the content retrieved by the RFB module. Adam optimizer is used, and the implementation is done with Keras. The LSTM is pre-trained with a batch size of 300 for 50 epochs, using the MSE loss function between the outputs of the LSTM and the actual popularities.

A. Parameter Analysis

1) Cache Size: We first study the impact of the cache size on the overall quality of the rendered viewports in Fig. 4a. From this figure, it can be observed that for small cache sizes, i.e., 5%, the performance gap between the proposed scheme and the LFU, LRU, and FIFO is approximately 1.4 dB, 1.8 dB, and 1.9 dB, respectively. For large cache size values, i.e., 25%, this gap increases to about 2 dB, 2.2 dB, and 2.3 dB, respectively. This gap is attributed to the fact that the proposed scheme achieves a better cache hit ratio, as shown in Fig. 4b. The increased cache hit ratio of the proposed scheme is attributed to the use of an LSTM network to predict the popularity of the content that will be prefetched to the SBSs caches. Also, the proposed scheme takes advantage of the flexibility regarding the number of cached tiles in high quality for each cached 360^o video. This results in caching in high quality most or even all the tiles of a 360^o scene for the most popular 360^o videos, while for the less popular 360^o videos only a few tiles are cached or even none.



(a)

(b)

Fig. 4. (a) Y-PSNR of the rendered viewports and (b) cache hit ratio with respect to the cache size for all the schemes under comparison.

2) Video popularity distribution: Next, in Fig. 5, we investigate the impact of the users' requests for the various 360^o videos on the overall quality of the rendered viewports. To this aim, we vary the Zipf shape parameter η_v in the range $[0.8, 1.6]$. We can note that an increase in the Zipf shape parameter η_v leads to an increase in the overall quality of the rendered viewports, for all the schemes. This is because as the Zipf shape parameter η_v increases, the video popularity distribution gets steeper, and a smaller number of 360^o videos becomes more popular. This leads to more tiles to be served directly from the caches of the SBSs at a small delay, allowing

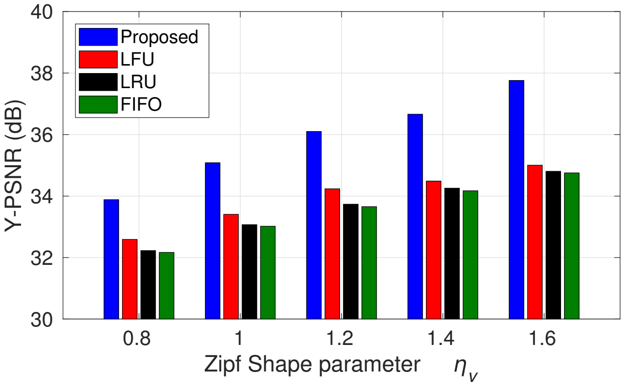


Fig. 5. Y-PSNR of the rendered viewports with respect to the Zipf shape parameter of the 360^o videos for all schemes under comparison.

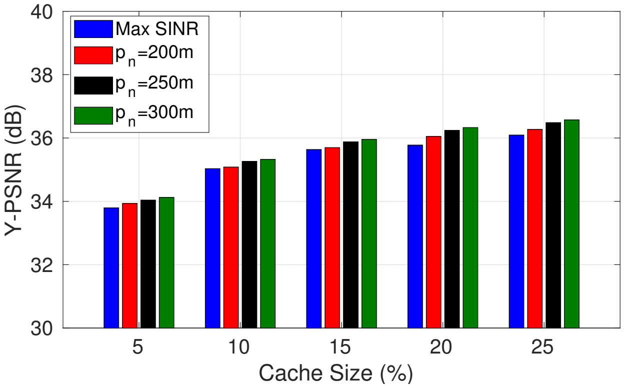


Fig. 6. Y-PSNR of the rendered viewports with respect to the cache size for all schemes under comparison.

more tiles to be served in total to the users under the end-to-end time constraint.

3) SBS Radius: To investigate the impact of users' association with multiple SBSs on the overall quality of the rendered viewports, we vary the SBSs radius p_n in the range $[200, 300]$ m. For the sake of completeness, we also examine the case where users are assigned to only one SBS, i.e. the SBS with the maximum SINR. The simulation results are depicted in Fig. 6. As we can observe, an increase in the radius of the SBSs leads to an increase in the overall quality of the rendered viewports due to the fact that more users will be associated with multiple SBSs. Obviously, in case users are assigned to the SBSs that have the maximum SINR, the overall quality of the rendered viewports is lower compared to its counterparts. This happens as in the former case users are associated with only one SBS from where they can download their data.

V. Conclusion

In this paper, we proposed an online caching system to support live streaming 360^o videos in mobile networks with the aim to maximize the overall quality of the rendered viewports and minimize the usage of the backhaul by optimizing the cache hit ratio. To populate the content of the caches of the SBSs with the most popular content, we used LSTM networks to decide whether to cache the prefetched content. To enhance the performance of our proposed method, we exploited the association of users with multiple SBSs. For the evaluation, we used real 360^o video navigation patterns and compared our method with that of LFU, LRU, and FIFO schemes. The results showed that our method outperforms its counterparts significantly. Further, the benefits of using LSTM networks to predict the popularity of the content that will be prefetched to the caches of the SBSs and users' association with multiple SBSs were made apparent.

References

[1] X. Xie and X. Zhang, "POI360: Panoramic mobile video telephony over lte cellular networks," in *Proc. of the ACM Int. Conf. on Emerging Networking EXperiments and Technologies, CoNEXT '17*, Incheon, Republic of South Korea, Dec 2017.

[2] A. T. Nasrabadi, A. Mahzari, J. D. Beshay, and R. Prakash, "Adaptive 360-degree video streaming using layered video coding," in *Proc. of IEEE Virtual Reality, VR'17*, Los Angeles, CA, USA, Mar. 2017.

[3] X. Corbillon, G. Simon, A. Devlic, and J. Chakareski, "Viewport-adaptive navigable 360-degree video delivery," in *Proc. of IEEE Int. Conf. on Communications, ICC'17*, Paris, France, May 2017.

[4] M. Jeppsson, H. Espeland, C. Griwodz, T. Kupka, R. Langseth, A. Petlund, P. Qiaoqiao, C. Xue, K. Pogorelov, M. Riegler, D. Johansen, and P. Halvorsen, "Efficient live and on-demand tiled HEVC 360 VR video streaming," in *Proc. of IEEE Int. Symp. on Multimedia, ISM'18*, Taichung, Taiwan, Dec. 2018.

[5] R. Aksu, J. Chakareski, and V. Swaminathan, "Viewport-driven rate-distortion optimized scalable live 360^o video network multicast," in *Proc. of IEEE Int. Conf. on Multimedia Expo Workshops, ICMEW'18*, San Diego, CA, USA, Jul. 2018.

[6] Y. Hu, S. Xie, Y. Xu, and J. Sun, "Dynamic VR live streaming over MMT," in *Proc. of IEEE Int. Symp. on Broadband Multimedia Systems and Broadcasting, BMSB'17*, Cagliari, Italy, Jun. 2017.

[7] C. Ge, N. Wang, W. K. Chai, and H. Hellwagner, "QoE-assured 4K HTTP live streaming via transient segment holding at mobile edge," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 8, pp. 1816–1830, Aug. 2018.

[8] P. Maniotis, E. Bourtsoulatze, and N. Thomos, "Tile-based joint caching and delivery of 360^o videos in heterogeneous networks," in *Proc. of IEEE 21st Int. Workshop on Multimedia Signal Processing, MMSP'19*, Kuala Lumpur, Malaysia, Sep. 2019.

[9] ——, "Tile-based joint caching and delivery of 360^o videos in heterogeneous networks," *IEEE Trans. on Multimedia*, in press.

[10] G. Papaioannou and I. Koutsopoulos, "Tile-based caching optimization for 360^o videos," in *Proc. of the 20th ACM Int. Symp. on Mobile Ad Hoc Networking and Computing, Mobihoc '19*, Catania, Italy, Jul. 2019.

[11] [Online]. Available: https://engineering.fb.com/ios/under-the-hood-broadcasting-live-video-to-millions/

[12] L. Sun, F. Duanmu, Y. Liu, Y. Wang, Y. Ye, H. Shi, and D. Dai, "A two-tier system for on-demand streaming of 360 degree video over dynamic networks," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 1, pp. 43–57, Mar. 2019.

[13] L. Alexander, R. Johnson, and J. Weiss, "Exploring zipf's law," *Teaching Mathematics and Its Applications: International Journal of the IMA*, vol. 17, no. 4, pp. 155–158, Dec 1998.

[14] E. Baccour, A. Erbad, A. Mohamed, K. Bilal, and M. Guizani, "Proactive video chunks caching and processing for latency and cost minimization in edge networks," in *Proc. of IEEE Wireless Communications and Networking Conference, WCNC'19*, Marrakesh, Morocco, 2019.

[15] F. Duanmu, Y. Mao, S. Liu, S. Srinivasan, and Y. Wang, "A subjective study of viewer navigation behaviors when watching 360-degree videos on computers," in *Proc. of IEEE Int. Conf. on Multimedia and Expo, ICME'18*, San Diego, CA, USA, July 2018.