DNNStream: Deep-learning based Content Adaptive Real-time Streaming

Satish Kumar Suman
Dept. of Computer Science
DRIEMS Cuttack
satishkmrsuman@gmail.com

Aniket Dhok
IEEE member
Bangalore, India
aniketdhok01@gmail.com

Swapnil Bhole
IEEE member
Bangalore, India
swapnil.rajendra@gmail.com

***Abstract*—With the advent of modern smartphones, AR, VR services and advancement in display resolution of mobile devices coupled with real-time streaming services, the demand for high-resolution video has boomed. To fulfill this requirement, a variety of Adaptive Bit-Rate Streaming methods for Video-on-Demand applications are employed. However, the use of multi-pass encoding in the aforementioned methods renders them obsolete when it comes to real-time video streaming due to latency restrictions. In this work, we bypass the conventional multiple-encoding used in Video-on-Demand applications and present a novel machine-learning-based approach that estimates the optimal video resolution for a given content at a particular bit-rate for ultra low latency applications. A new feature that captures temporal as well as spatial correlation in video sequence has been used to train the Deep Neural Network (DNN) model. A python-based testbed is designed to evaluate the proposed scheme. Experiment results corroborate the viability and effectiveness of the proposed method for real-time mobile video streaming applications.**

I. Introduction

Video streaming has become ubiquitous in this smartphone world, thanks to the developments in the network and data services. In the Cisco Visual Networking Index, it is reported that 82 percent of global IP traffic will be video by 2022, up from 75 percent in 2017 [1]. AR-VR traffic and global internet gaming traffic will grow twelve-fold and nine-fold between 2017 and 2022 respectively. The growth of real-time video streaming is evident from the increasing popularity of services such as Twitch.tv. To address the challenges of limited resources in our devices, huge traffic on the network and at the same time, providing the best video streaming experience to the user, it is imperative to develop and deploy robust as well as low latency bit rate adaptation algorithms. To overcome these challenges, Netflix has developed SVT-AV1 video encoder [2]

Since its introduction in 2005, HTTP Adaptive Streaming (HAS) has become the dominant approach for video streaming owing to its easy integration, cheaper deployment, and simplicity over other proprietary streaming solutions [3]. HAS provides network resilience to improve QoE. Before publishing on the HTTP server, the original file is prepared for streaming by dividing it into chunks of equal playback time and encoding each of these chunks at all supported resolutions and bitrates. The server then prepares a manifest file for the clients which contains all the metadata, viz. available resolution and bitrate for the video chunks. With the help of this manifest file, the current network bandwidth, and the media playback buffer status, the client repeatedly requests the best possible video chunk. Apple's development of HTTP Live Streaming (HLS) platform [4] and several other commercial HAS based solutions brought HAS into limelight. Subsequently, it was standardized under the name Dynamic Adaptive Streaming over HTTP (DASH) [5], thanks to the joint efforts of Moving Pictures Experts Group (MPEG) and 3rd Generation Partnership Project (3GPP).

DASH is further improved in terms of QoE and video smoothness in [6] where a Markov Decision-based Rate Adaptation Approach for Dynamic HTTP Streaming was proposed. The reward function is tailored to satisfy required QoE, playback parameters, etc. exploiting a sub-optimal greedy algorithm. In [7], a similar approach is followed where video streaming is modeled as a Markov Decision Process (MDP) and reinforcement learning is used to find the sub-optimal solution of MDP in real-time. An approach combining deep learning and reinforcement learning known as D-DASH [8] achieves significant improvements over most of the state-of-the-art rate adaptation approaches, in terms of higher video quality, faster convergence, better policy optimality, and more stability. A still better approach is to use Content of Interest (CoI) based rate adaptation streaming [9], where, a deep neural network first estimates the interestingness of the video content, and then a Deep-Q-network (DQN) assigns higher bit rate budget for more interesting content.

One of the most important applications of real-time adaptive bit-rate streaming is active gaming platforms like Google Stadia and Microsoft xCloud. Implementation of DASH in live game streaming platforms lowers the infrastructure costs by around 40 percent [10]. 5 dB improvement in viewing and close to 50 Gbps bandwidth reduction was achieved in [11] by implementing Segment of Interest (SoI) resource allocation algorithm on their testbed. SoI segments are extracted based on features such as webcam images from streamer and viewer, CPU and GPU utilization and keyboard/mouse inputs from the viewer. Resources are allocated based on SoI detector output. A video classification approach is adopted in [12] using features such as texture details, camera movements with the aim to construct a decision tree to map these features

978-1-7281-8895-9/20/$31.00 © 2020 IEEE

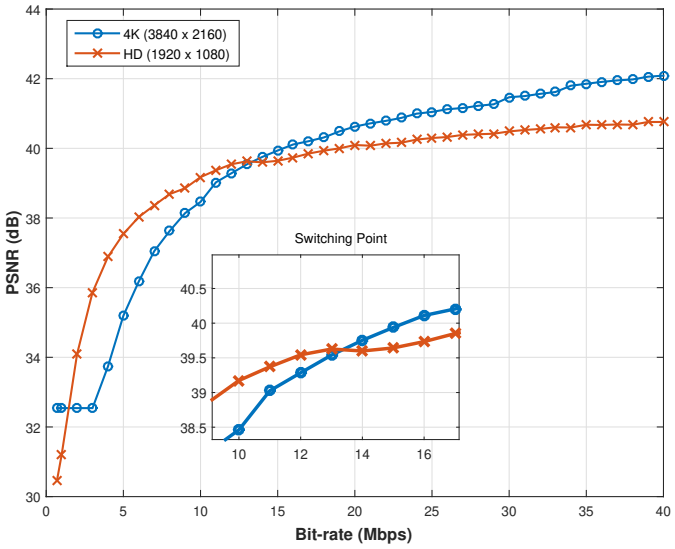

Fig. 1. BD Rate Curve showing Switching Point for a single GoP

to their quality classes, and exploiting Video Multi-Method Assessment Fusion (VMAF) as quality assessment metric.

In this work, we propose to dynamically adapt the resolution of current video segment based on predictions generated by DNN model which is trained to generate rate at which two Bjontegaard Function (BD) rate curves for the immediately previous Group of Pictures (usually GoP consists of 9 frames) to the current video segment intersect (also referred as *switching point*) where the blue curve corresponds to video being encoded at its original resolution while the red curve corresponds to the video being down-scaled and encoded at lower resolution and then again up-scaled to original resolution. This predicted rate is then compared with the current bit-rate to choose the optimal resolution for the current segment. The novel feature that computes homogeneity of the block formed by the co-located blocks of size 384 x 384 from all the frames in a single GoP is used to train the DNN. A testbed is also developed to validate our DNNStream approach and the results obtained are promising. In QARC (video Quality aware rate control algorithm) [13] future bitrate is predicted from the past video frames using DRL(Deep Reinforcement learning) whereas in this study we predict the best resolution for video sequence for given client bitrate from the past video frames.

The structure of the paper is as follows. Section II provides details of the novel features used and the proposed design of DNNStream. Section III describes the TestBed. Results and comparison of DNNStream with conventional encoding techniques are presented in Section IV. Section V gives an insight into the possible future extensions and finally, concludes this paper.

II. Preliminaries & Proposed Design

A. Motivation

The behavior of individual video segments of 9 frames was studied by encoding the segments using libvpx (VP9 encoder) for two different resolutions namely 4K and HD at bit rates ranging from 0.7 Mbps to 40 Mbps. The original resolution of uncompressed video content used during this study was 4K. Fig. 1 shows BD rate curves for one such video segment. It can be observed from figure that these two curves intersect at approximately 13 Mbps. Let loss due to downscaling from 4K to HD be L_D, loss due to encoding at HD be L_{EHD} , loss due to upscaling be L_U and loss due to encoding at 4K be L_{E4K}, then at the intersection point:

$$L_D(R) + L_{EHD}(R) + L_U(R) = L_{E4K}(R) \tag{1}$$

This *switching point* is content dependent and can have significant variations across different video segments. The knowledge of this intersection point is necessary for the optimal adaptation of resolution because to the left of this point, always encoding the video segment by first downscaling the video segment at lower resolution (here it is HD), encoding at HD and ultimately upscaling to the original resolution at the time of display will result in better QoE than encoding the video segment at 4K (original resolution).

B. Feature Extraction

DNN model is used to predict the switching point for a given video segment consisting of 9 frames. The feature used to train DNN is extracted from the original uncompressed video segment by first constructing the concatenated frame, computing Gray-Level Co-Occurrence Matrix (GLCM) of concatenated frame and finally calculating the homogeneity of computed GLCM matrix. Homogeneity *H* indicates by how much the non-diagonal elements vary from the diagonal elements in GLCM. From [14], GLCM is calculated on *w*x*h* of frame *F* for distance *r* and orientation θ,

$$C_M(r,\theta) = \sum_{u=1}^{w}\sum_{v=1}^{h} \begin{cases} 1, & \text{if } F(u,v) = m \text{ and } F(u',v') = n \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$H = \sum_{m,n} \frac{C_M(m,n)}{1+|m-n|} \tag{3}$$

where *u'* and *v'* are computed from *u*, *v*, *d*, θ.
$1 <= u' <= u$,
$1 <= v' <= v$ and $1 <= m,n <= q-1$
where q is the number of quantization levels. In our case, $q = 2^8$.

The concatenated frame is generated by combining co-located blocks of size 384 x 384 from each frame in GoP as shown in Fig. 2. The block-size of 384 x 384 was chosen for feature extraction as it gives better results than block-sizes of 128 x 128 and 256 x 256 for the video contents used in this study. Similarly, this process of constructing concatenated frames and homogeneity computation is repeated over all 384 x 384 blocks in frame. The feature vector of all such homogeneity values are given as input to DNN. The feature extracted by the proposed method takes into account

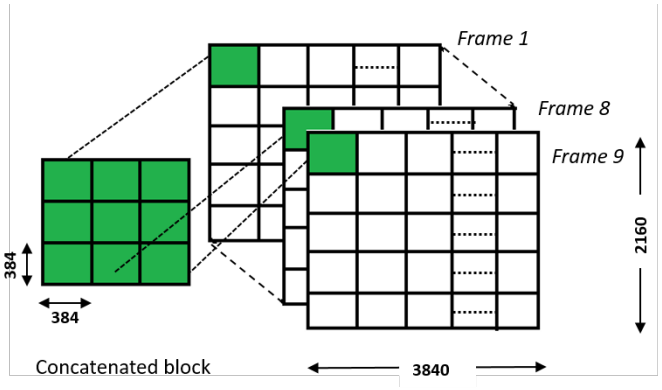

Fig. 2. Homogeneity extraction for 4K resolution

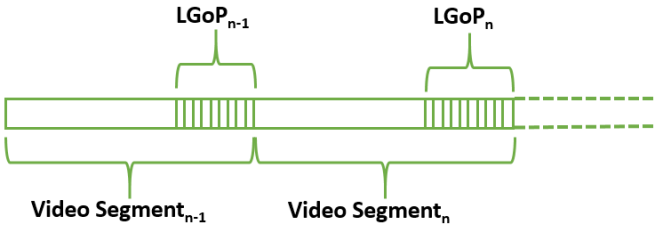

Fig. 3. Decision Mechanism in DNNStream

the spatial and temporal characteristics of video segment because homogeneity is calculated spatially over blocks of different frames. GLCM is used as a feature in this study rather than features based on video encoding parameters to make the DNNStream framework independent of the video encoder used.

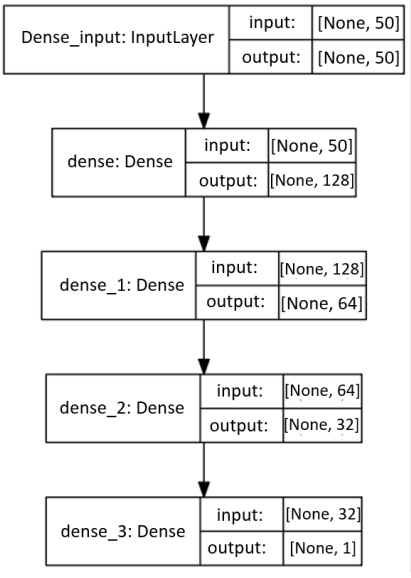

Fig. 4. Training setup for DNNStream

C. Proposed Design

For streaming Video Segment n, the feature vector is generated using frames of last GoP of the previous segment referred to $LGoP_{n-1}$ in Fig. 3. If C_1 is the first homogeneity calculated on the concatenated frame constructed using the first 384 x 384 blocks of every frame, then the feature vector can be represented as $x = [C_1, C_2, ..., C_n]$ where

$$n = (W * H)/(384 * 384)$$

W, H are the width and height of the frame respectively. This feature vector is fed to the DNN model which predicts the switching point for the input GoP. In this study DNN is employed for prediction instead of simple feature extraction and comparison because the threshold which will be used for comparison in latter will vary significantly across different video contents whereas DNN will be able to generalize well across different video contents. The DNN model consists of three layers 128-64-32 as shown in Fig. 4.

None dimension denotes that the training batch size is user-configurable. As shown in Fig. 5, the predicted rate R_p is compared with the current rate R_a to decide if the resolution needs to be adapted or not. If $R_p < R_a$, the video segment is encoded at original resolution, otherwise the video is downscaled, encoded at lower resolution by the streaming server. In Fig. 6, for encoding segment 10 since the current bitrate Ra (15 Mbps) is less than predicted birate Rp (29 Mbps) the segment would be downscaled before encoding. The client needs to take into account the same decision that is taken on server-side whether to upscale the video segment after decoding or not. In both cases, the segment is encoded at current bit-rate R_a which the client needs to communicate to the server. The proposed method does not introduce latency because the optimal resolution for video segment n is decided based on the last GoP of the previous segment which is already encoded, whereas conventional methods buffer the input frames and perform multiple encoding of the same video segment to decide optimal parameters which result in latency, rendering them obsolete for ultra low latency streaming applications like interactive cloud gaming. Moreover, no multiple instances of the encoder are employed to generate all possible representations of the same video segment and let the client pick the best possible configuration amongst them.

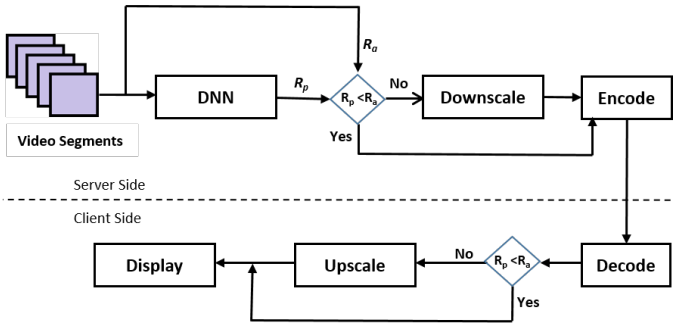

Fig. 5. System Design of DNNStream

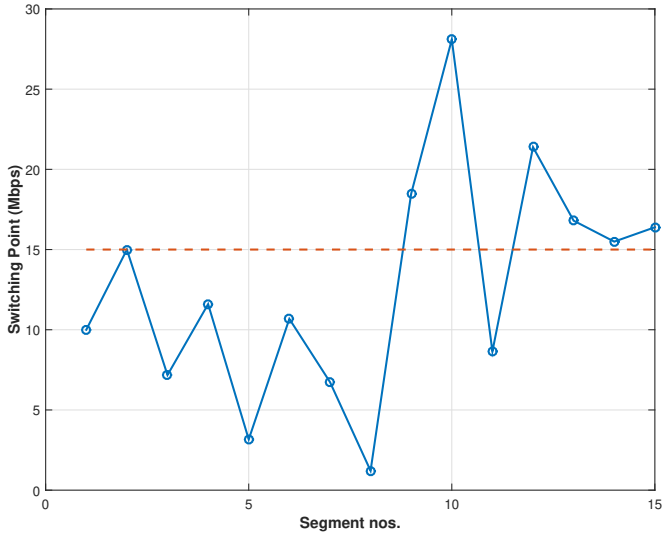

Fig. 6. Switching Point vs Segments

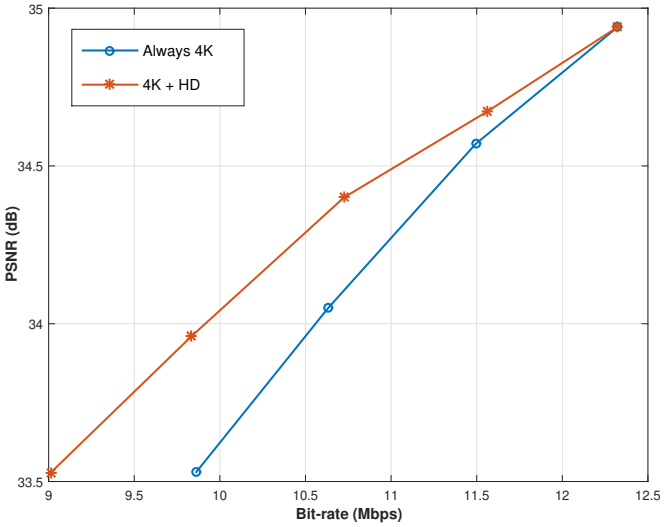

Fig. 7. BD rate curve for Dota 2

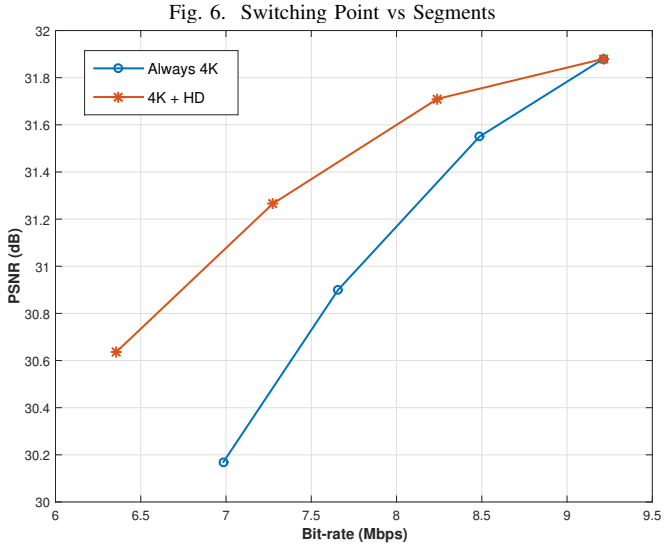

Fig. 8. BD rate curve for PUBG

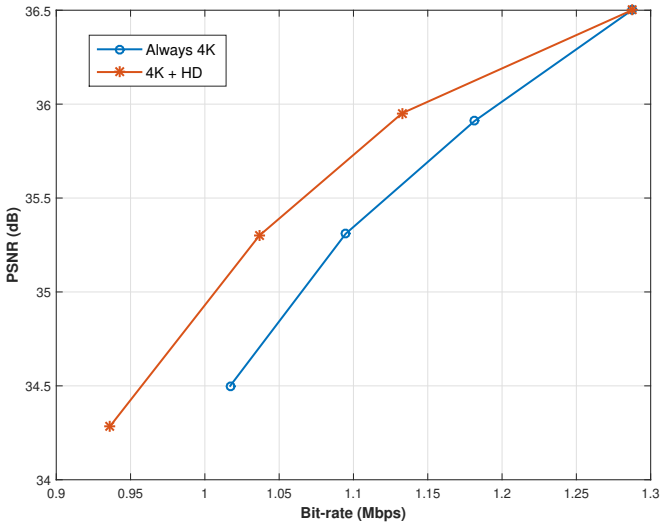

Fig. 9. BD rate curve for Assassins Creed

III. Test Bed

As cloud gaming is one of the most prevalent examples of real-time streaming, the video contents used for training and testing are gaming contents. For evaluating DNNStream, a sample testbed is created in Python which tries to simulate both server and client functionality. To generate training labels for DNN, the uncompressed video content is divided into segments of 9 frames using *--segment_time* option in ffmpeg. The number of video segments for every gaming content used is listed in table I. For each of these segments, the switching point is determined by plotting BD rate curves for bit rates ranging from 0.7 Mbps to 40 Mbps similar to Fig. 1.

The following encoding recipe is used:

./vpxenc --codec=vp9 --skip=0 --cpu-used=5 --end-usage=cbr --min-q=0 --max-q=63 --auto-alt-ref=0 --kf-max-dist=60 --rt --frame-parallel=0 --buf-sz=400 --buf-initial-sz=250 --buf-optimal-sz=300 --psnr -o output.ivf --target-bitrate=700 --tile-columns=4 --tile-rows=0 -w 1920 -h 1080 --fps=60000/1000 input.yuv

The following ffmpeg recipe with lanczos filter is used for upscaling and downscaling:

ffmpeg -s:v 3840x2160 -i input.yuv -vf scale=1920x1080 -sws_flags lanczos -c:v rawvideo -pix_fmt yuv420p downscaled.yuv

The training features are also generated for these segments as illustrated in the above sections out of which 50 features are used. Keras API is used as a front-end to design our DNN which runs on top of Google's machine learning library TensorFlow. The video segments of playback time equal to 1 second are fed to the testbed as input. The contents used in this study are 60 fps so each segment will consist of 60 frames. Testbed then uses trained DNN model and bitrate requested by the client to decide the optimal resolution for the current video segment based on the last 9 frames of the previous segment. If

TABLE I
TRAINING SET

Gaming Contents	No. of Video Segments
The Witcher 3	694
Call of Duty	724
Jump Force	634
For Honor	1442
Horizon Zero Dawn	1188
Metro Last Night	1892
Total	6574

the determined resolution is lower than the original resolution, the video segment is downscaled before encoding, otherwise it is directly encoded at the client bitrate. The client then decodes the video segment and based on the same decision taken on the server-side, the video segment is upscaled before displaying. The client bit-rate is varied randomly in steps to simulate the real-world scenario within range the of 0.7 Mbps to 40 Mbps, same as the training set.

IV. RESULTS

DNNStream achieves significant BD rate gains compared to fixed resolution encoding because it attempts to select best resolution at particular bitrate. BD rate for DNNStream will always be better than or equal to the BD rate when the gaming video is encoded at the original resolution. Table II shows BD rate gains achieved by DNNStream for three different test gaming contents using testbed developed. For evaluation, 15 video segments of varying complexity each of 1 second were selected from all three test games. BD rate curves for DOTA2, PUBG and Assassins Creed have been plotted in Fig. 7 to 9 respectively. The BD rate curves for DNNStream and fixed resolution encoding at 4K meet after certain bitrate this is because at higher bitrates encoding at 4K resolution always results in better QoE than encoding at HD which is correctly predicted by DNN model. This intersection point is heavily dependent on the video content and can vary significantly across different portions of the same gaming content. In case of scene changes where the consecutive video segments are very different in terms of complexity, there is a high probability that the optimal resolution is not selected for the current video segment, as the decision is based on the previously encoded frames. However from the next video segment there would be immediate correction in this decision because DNN model can better predict switching point for subsequent video segments because of temporal similarity.

TABLE II
BD RATE GAINS FOR TESTING SET

Gaming Contents	BD Rate Y	BD Rate Avg
Dota 2	-0.8228338375518174	-2.372812915587341
PUBG	-11.779742628355528	-14.262849204988392
Assassins Creed	-2.011169291687598	-3.7875902575107645

V. CONCLUSION & FUTURE SCOPE

In this work, we proposed ultra low latency resolution adaptation framework that provides best possible resolution for a given bitrate. Novel feature that captures both temporal as well as spatial characteristics of video segment is used to train DNN which coupled with client bitrate provides the resolution that results in best possible QoE. DNNStream can be used for applications where latency is of prime importance such as active cloud gaming because only previously encoded frames are used to take decision rather than buffering of frames. Testbed is developed to comprehensively evaluate DNNStream framework in real-world scenarios and a variety of test video contents. Simulation results show on an average 6 percent BD rate improvement on test vectors.

This work can be extended by integrating DNNStream in the end-to-end system to measure latency compared with the conventional ABR algorithms. Switching between multiple resolutions can be studied contrary to only two resolutions 4K and HD considered in the current study. Moreover, the testbed can be used to evaluate the performance of DNNStream for more gaming contents.

REFERENCES

[1] Cisco, "Cisco visual networking index: Forecast and methodology, 2017-2022," *White Paper, June*, 2017.

[2] Netflix, "Svt-av1: open-source av1 encoder and decoder," 2020.

[3] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer, and R. Zimmermann, "A survey on bitrate adaptation schemes for streaming media over http," *IEEE Communications Surveys Tutorials*, vol. 21, no. 1, pp. 562–585, 2019.

[4] A. Developer, "Apple http live streaming," 2016.

[5] T. C. Thang, Q.-D. Ho, J. W. Kang, and A. T. Pham, "Adaptive streaming of audiovisual content using mpeg dash," *IEEE Transactions on Consumer Electronics*, vol. 58, no. 1, pp. 78–85, 2012.

[6] C. Zhou, C.-W. Lin, and Z. Guo, "mdash: A markov decision-based rate adaptation approach for dynamic http streaming," *IEEE Transactions on Multimedia*, vol. 18, no. 4, pp. 738–751, 2016.

[7] M. Xing, S. Xiang, and L. Cai, "A real-time adaptive algorithm for video streaming over multiple wireless access networks," *IEEE Journal on Selected Areas in communications*, vol. 32, no. 4, pp. 795–805, 2014.

[8] M. Gadaleta, F. Chiariotti, M. Rossi, and A. Zanella, "D-dash: A deep q-learning framework for dash video streaming," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 703–718, 2017.

[9] G. Gao, L. Dong, H. Zhang, Y. Wen, and W. Zeng, "Content-aware personalised rate adaptation for adaptive streaming via deep video analysis," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–8.

[10] K. Pires and G. Simon, "Dash in twitch: Adaptive bitrate streaming in live game streaming platforms," in *Proceedings of the 2014 Workshop on Design, Quality and Deployment of Adaptive Video Streaming*, 2014, pp. 13–18.

[11] T.-Y. Fan-Chiang, H.-J. Hong, and C.-H. Hsu, "Segment-of-interest driven live game streaming: saving bandwidth without degrading experience," in *2015 International Workshop on Network and Systems Support for Games (NetGames)*. IEEE, 2015, pp. 1–6.

[12] S. Zadtootaghaj, S. Schmidt, N. Barman, S. Möller, and M. G. Martini, "A classification of video games based on game characteristics linked to video coding complexity," in *2018 16th Annual Workshop on Network and Systems Support for Games (NetGames)*. IEEE, 2018, pp. 1–6.

[13] T. Huang, R.-X. Zhang, C. Zhou, and L. Sun, "Qarc: Video quality aware rate control for real-time video streaming based on deep reinforcement learning," in *Proceedings of the 26th ACM international conference on Multimedia*, 2018, pp. 1208–1216.

[14] R. M. Haralick, K. Shanmugam, and I. H. Dinstein, "Textural features for image classification," *IEEE Transactions on systems, man, and cybernetics*, no. 6, pp. 610–621, 1973.