Judgment and Decision Making, Vol. 13, No. 2, March 2018, pp. XX–XX

Title

Zhenbo Cheng* Gang Xiao‡

Abstract

In recent years, the increasing demand for seamless switching of HD streams and the widespread adoption of adaptive streams based on the DASH standard have become the main driving force for the research of rate adaptive algorithms. Most of the existing bit rate adaptive algorithms are hard-coded and cannot effectively cope with changes in the network environment. In addition, some scholars have proposed to use the Q learning method that can obtain higher QoE to learn the optimized code rate adaptive algorithm. However, Q learning has the problem that it is difficult to encode continuous state values and the algorithm converges slowly in large state spaces. To this end, this paper combines the nearest neighbor algorithm and proposes a KNN-Q learning algorithm that can handle continuous state. The experimental results show that in the rate adaptive scenario, KNN-Q learning can achieve higher QoE and faster convergence speed than ordinary Q learning.

Keywords: DASH, HTTP adaptive streaming, reinforcement learning,Q learning,KNN

1 Introduction

At present, video data is becoming the mainstream data of the traditional Internet, especially the mobile Internet.Video data requires a higher bit rate than other data transmissions, and the resulting increase in mobile traffic is more pronounced. According to the Cisco Global Mobile Data Traffic Forecast Update white paper(2016- 2021)(Index, n.d.), mobile video flow will grow 8.7 times by 2021, accounting for 78% of total mobile traffic. Users have more requirements for video quality and interactive features.

Thousands of videos are transmitted over HTTP every day. Video data can be easily transmitted through HTTP and NAT devices. Based on HAS (HTTP Adaptive Streaming) technology, major vendors have developed adaptive streaming media technologies for their respective devices or clients, such as Apple's HLS (HTTP Live Streaming) , Microsoft's MSS (Microsoft Silverlight Smooth Streaming) and Adobe's HDS (HTTP Dynamic Streaming). The MPEG organization developed the DASH (Dynamic Adaptive Streaming over HTTP) standard in 2011 to unify the video technology standards for transmitting dynamic bitrates between different device and server manufacturers(Sodagar, 2011).Compared with HLS and MSS, the DASH standard not only supports multiple encoding methods, but also supports CDN docking from multiple manufacturers.

The design of the DASH standard is that the server divides the original video into video clips with the same content and different bit rates, and then generates an index file MPD (Media Presentation Description) containing all the video clip information and their download addresses. After the client downloads and parses the MDP file, it switches the video clip of the appropriate bit rate according to the current video quality and network conditions during playback(Xu, Zhou, & Chiu, 2014). The DASH standard opens up the design of the bit rate adaptive algorithm, enabling developers to implement their own rate adaptive algorithms to provide higher QoE (Quality of Experience).

The existing bit rate adaptive algorithms are mostly based on heuristic algorithms. The algorithm usually uses hard coding to select the appropriate bit rate according to the network conditions, which may result in the inability to effectively cope with the frequent fluctuations of the network environment(Claeys et al., 2013). Dana et al. proposed a Q-Learning algorithm in reinforcement learning to obtain an optimized bit rate adaptive strategy(Li et al., 2014). Compared with the heuristic algorithm, this method can obtain higher QoE, but when the state is divided too much, the method has the difficulty of learning continuous state values and slow learning convergence. Aiming at this problem, this paper proposes a KNN-Q learning algorithm as a strategy for obtaining video clips in combination with the nearest neighbor algorithm.The experimental results show that the KNN-Q learning algorithm can achieve higher QoE.

This work is supported by Public Projects of Zhejiang Province (2016C31G2020069) and the 3rd Level in Zhejiang Province âĂIJ151 talents projectâĂİ to Zhenbo Cheng. We thank Liwen Bianji, Edanz Group China (www.liwenbianji.cn/ac), for editing the English text of a draft of this manuscript. The authors declare no competing financial or nonfinancial interests.

*Department of Computer Science and Technology, Zhejiang University of Technology, Hangzhou, China

†Email: xg@zjut.edu.cn

2 Related work

The client downloads streaming media and must be able to adaptively select the bit rate according to the network situation, client state and user experience, in order to play video reliably and efficiently in different environments.

Studies(Zink, Schmitt, & Steinmetz, 2003) have shown that users are not sensitive to the improvement of image quality, but often can detect the decline in image quality. The algorithm should try to avoid changing the image quality frequently. Even if the image quality change cannot be avoided, algorithm need to control changes within a small range. QoE-related adaptive strategies often need to consider multiple aspects such as available bandwidth, buffer, quality, etc. to accurately reflect the QoE perceived by the user. Mok et al(Mok, Luo, Chan, & Chang, 2012). proposed a heuristic-based bit rate adaptive algorithm combined with QoE. The QDASH-abw module is located in the proxy of the server to measure the highest video quality that can be supported under the current bandwidth. The client selects the most suitable video quality rate based on the QDASH-qoe algorithm. Their experimental results indicate that users prefer a smooth transition between the best and worst video quality, rather than a sudden switch. Zink et al(Zink et al., 2003).'s NOVA adaptive strategy proposes that "cross-layer" is designed to weigh resource allocation and rate adaptation, using the variance of the quality and time of the previous S video segments as a measure of QoE. The heuristic-based rate adaptive algorithm cannot easily establish a predictable mathematical model. And,the flexibility of the bit rate selection strategy is low. In the complex network environment, the stability of the this algorithm is poor.

Using machine learning methods, the client can learn how to adjust the bit rate of the requested video block based on changes in the context before and after intervention. Using machine learning methods can enhance the adaptability and flexibility of the client's rate adaptive strategy. Based on the random forest decision tree model, Chien et al(Chien, Lin, & Chen, 2015). map network state eigenvalues (such as the bandwidth of the latest slice, the average bandwidth of the session, and the fluctuation of the bandwidth) to the bit rate label obtained by the other excellent bit rate adaptive algorithm. Finally, the trained classification model is used to predict the bit rate size of the video segment. One disadvantage based on supervised learning is the need to prepare training data. Although Chien's algorithm can collect user data through online learning methods, it is difficult to train a robust model for all scenes due to the environment, user requirements and the diversity of terminal devices.

Introducing reinforcement learning into the DASH bit rate adaptive algorithm, the HAS client can continuously interact with the environment, train its own strategy according to the feedback of the current network environment, and dynamically learn to select the best action. Compared to supervised learning, reinforcement learning does not require a large amount of training data to be prepared, and only a reasonable action reward function needs to be defined during learning. Marinca et al(Marinca, Barth, & Vleeschauwer, 2013). implemented a HAS self-learning system on the MAC layer. The system uses a partially visible Markov model to model a bit rate adaptive algorithm based on Q learning. Zhu et al(Zhu, Lany, & Schaarz, 2013). proposed an infinite time Markov model, using the Virutal Experience (VR) strategy to update the Q table during training. The experimental results of they proposed method show that the VR strategy accelerates the process of model convergence. Compared with the approximate optimal heuristic algorithm, thier overall reward is higher and the delay is significantly reduced in the environment with changing network state. Zhang et al(Zhang, Zheng, Hua, Huang, & Yang, 2018). positioned QoE criteria as total cumulative bit rate, packet loss rate and video starvation duration, and constructed a deep self-transfer learning network (STL) considering QoE. The STL algorithm integrates a series of pseudo-rewards to train the virtual decision-making center. Then the virtual decision-making center continuously passes the learned adaptive decision to the original decision-making center. The experimental results show that the STL algorithm can improve the training efficiency and generalization performance. After combining the characteristics of deep learning and reinforcement learning, Gadaleta et al(Gadaleta, Chiariotti, Rossi, & Zanella, 2017). used the advantages of feedforward neural network and recurrent neural network to design a D-DASH framework. The D-Dash framework uses video clip quality and video replay event evaluation performance to achieve a fairly high QoE in both simulated and real-world environments.

In the context of using reinforcement learning to solve rate adaptation, it is necessary to weigh the algorithm convergence speed and state partition granularity. Too many states can enhance the robustness of the system, but it will cause the convergence speed to be too slow in the iteration; If the state division is too discrete, and the problem situation cannot be drawn to the ideal state, but the convergence speed is accelerated. In the adaptive rate scene, the states are continuous values. This paper proposes an algorithm combining k-nearest neighbor and Q-learning. Considering the video quality of streaming media, three aspects related to QoE are comprehensively selected: the quality of video clips, the quality of video frames before and after, and the risk of buffer overflow construction to construct a reward function. In this paper, an algorithm based on KNN Q-learing for bit rate adaptive action of continuous state is implemented, and a simulation experiment is designed to compare the performance of ordinary Q learning and KNN-Q learning.

3 KNN-Q algorithm

A normal timing-based DASH client will use the discovery strategy to select the appropriate bitrate when requesting the next video segment. As used herein, an agent based on reinforcement learning which help DASH client to make decisions to complete this process of exploration.

Reinforcement learning is a kind of machine learning. Because the environment is partially visible to the agent, the uncertainty of the agent's response to the environment is determined. But interacting with the environment is the only way for an agent to learn and grow. The agent receives rewards from the environment after each step of the action, and the agent understands the reward and makes the next decision. The goal of the agent is to pick the best action among the given environments to maximize the cumulative rewards(Kaelbling, Littman, & Moore, 1996).

In the bit rate adaptive algorithm, the DASH client is defined as an agent, and the environment value observed by the agent is defined as the state s. The DASH client requests the download of the video segment of a certain bit rate as action a at a certain time. When agent start exploring, it updates the state of the current environment. In Q learning, the Q table is equivalent to the brain of the agent to help the agent make decisions. The Q function is used to evaluate the quality of each action in the current state. As shown in equation 1, the agent selects action a in state s, and how to update the Q value after obtaining the reward r given by the environment. Where $\eta \in [0;1]$ is the learning rate, representing the extent to which the new value replaces the old value; $\lambda \in [0;1]$as a discount factor, measure the importance of future rewards. In the case of the state of the environment division meticulous, using this method to update Q table will bring convergence speed is too slow. The environment state in the bit rate adaptive context is a continuous value. Based on the KNN algorithm, this paper updates several consecutive states of the current environmental state according to certain strategies, which can achieve the effect of speeding up the iteration rate and obtaining higher cumulative rewards(see section 3.4).

$$\begin{aligned}&(1-\eta)\,Q_{old}\,(s_t,a_t)+\eta\,[r_{t+1}+\lambda maxQ\,(s_{t+1},a)]\\&\Rightarrow Q_{new}\,(s_t,a_t)\end{aligned} \tag{1}$$

In each training iteration agent selects the appropriate bit rate video clip to download according to the previously learned Q value. For the specific selection strategy(see section 3.3), the reward given by the current action of the environment is calculated based on the reward function (see section 3.2). After multiple iterations of training, a convergent Q table is generated. When the DASH client selects the appropriate bit rate to download the video according to the final generated Q table . After downloading the current video clip, the video clip will be added to the buffer queue to deal with special cases such as network anomalies, and finally the video queue will be played.

4 Environment model

The DASH client needs to collect the currently visible environment state to build the corresponding reward function and act according to the corresponding policy. The following describes the environment state and defines the action behavior of the DASH client.

4.0.1 Environment status

This article uses three state components to define the state of the environment: the available bandwidth, the total length of the buffer, and the video quality of the previous video clip. The available bandwidth is measured by the DASH client; the total duration of the buffer represents the padding status of the video clip bufferd by the DASH client. The state of the final synthesis of these three state components defines $s_t = (BW_t, B_t, q_t - 1)$. The value range and discrete level of the specific state component are shown in Table 1.

Table 1: Environment state definition

Status element	Range	Level
Bandwidth	$[0; BW_{max}]\,bps$	$N+1$
Buffer filling	$[0; B_{max}]\,sec$	$\frac{B_{max}}{T_{seg}}$
Quality Level(SSIM)	$[-1; 1]$	N

As shown in Table 1, BW_{max} represents the highest available bandwidth, B_{max} represents the maximum video length that can be stored in the buffer area, and N represents the total number of bit rate levels. The calculation method of the quality of the previous video clip is evaluated by SSIM (for the calculation method, see section 3.2.1).

4.0.2 Selection action

The action of the DASH client is defined to request downloading of a video clip of a certain bit rate. The download rate of the video clip that can be selected by the DASH client is determined in advance by the transcoding module of the DASH client.

4.1 Reward function

In reinforcement learning, the reward function is the basic guide to the experience of the agent's learning strategy. In the streaming adaptive context, the reward function is generally used to measure QoE, but the evaluation of QoE is often subjective. For example, it is impossible to compare a video that fluctuates between high and low quality and a video that is stable in medium quality. So it is necessary to use

mathematical models to accurately measure QoE. This paper constructs a reward function from three aspects: current video clip quality, video clip quality fluctuation and buffer overflow risk.

4.1.1 Video quality

This article uses SSIM, which is widely used to evaluate video quality, as an indicator for evaluating video quality(Hore & Ziou, 2010; Wallendael, Leuven, Cock, & Lambert, 2013; Pasqualini, Fioretti, Andreoli, & Pierleoni, 2009). SSIM is an indicator that measures the similarity of two images and uses structural information to evaluate image quality. The specific calculation of SSIM is as shown in equation 2. For a given image X and Y, the SSIM value is calculated using the mean, variance and covariance of the pixel brightness in the two images. Where μ_X and μ_Y are the average brightness of the pixels of the images X and Y, σ_X^2 and σ_Y^2 is the variance of the pixel brightness values of the images X and Y, respectively, and σ_{XY} is the covariance between the pixel brightness values of the images X and Y, c_1 and c_2 are constants set to avoid the denominator being zero. SSIM$\in [-1, 1]$, the closer value of SSIM is to 1, the higher the similarity between the two images.

$$SSIM\,(X,Y) = \frac{(2\mu_X\mu_Y + c_1)\,(2\sigma_{XY} + c_2)}{\left(\mu_X^2 + \mu_Y^2 + c_1\right)\left(\sigma_X^2 + \sigma_Y^2 + c_2\right)} \tag{2}$$

Using equation 2, the SSIM mean of each video frame of the video clip to be downloaded and the original video clip is calculated to measure the quality of the video clip. As shown in equation 3, the quality of the video clip is represented by the average SSIM of the video clip, where m indicates that the video clip is composed of $m$frame pictures, and q_n indicates the video quality of the nth clip.

$$q_n = \frac{\sum_{i=0}^{m} SSIM(download_{seg(i)}, origin_{seg(i)})}{m} \tag{3}$$

As shown in equation 4, the video quality of the nth segment is used to represent the reward value brought by the DASH client downloading the video segment of the video quality.

$$R_{quality} = q_n \tag{4}$$

4.1.2 Video quality oscillation

When the network bandwidth changes, the video quality requested by the DASH client also changes, and the video quality oscillation will bring about a drop in QoE. As shown in equation5, the reward loss due to quality switching during video playback is recorded as $R_{quality-change}$, which is mapped to a penalty reward based on the oscillating step size α.

$$R_{quality-change} = -\alpha\,|q_n - q_{n-1}| \tag{5}$$

4.1.3 Buffer security

The penalty item is set in this paper to measure the buffer security. As shown in equations 6, 7, and 8, the penalty item consists of a secure buffer level penalty and a low buffer penalty. Where B_{max} represents the maximum buffer size of the buffer, and B_n represents the length of the video clip owned by the buffer when the nth video clip is downloaded. β and γ represent the focus on buffer security and low buffer.

$$R_{buffer-filling} = R_{buffer-safe} + R_{buffer-low} \tag{6}$$

$$R_{buffer-safe} = -min\left\{\beta\left(max\left[0, T_{download} - B_n\right]\right), 1\right\} \tag{7}$$

$$R_{buffer-low} = -\gamma\left(max\left[B_{max} - B_{n+1}, 0\right]\right)^2 \tag{8}$$

4.1.4 Total reward function

The total reward function will drive the DASH client to explore higher bit rate levels, fewer rate switches, and more reasonable behavioral decisions for bufferd data volumes. Since QoE is often subjective, it is necessary to set adjustable parameters. As shown in equation 9, this paper sets C_1, C_2, and C_3 to correspond to the weights of $R_{quality}$, $R_{quality-change}$, and $R_{buffer-filling}$, respectively, and sets specific values according to the QoE aspect of the focus.

$$R = C_1 R_{quality} + C_2 R_{quality-change} + C_3 R_{buffer-filling} \tag{9}$$

4.2 Exploration strategy

The choice of actions in the process of reinforcement learning needs to pay attention to the balance of exploration and exploitation. In order to obtain a higher cumulative maximum reward, the agent often selects an instant reward that has been executed before, and on the other hand, the agent needs to try an action that has not been executed before and may return a larger instant reward. Focusing on exploration may have an impact on short-term gains, and focusing on exploitation can maximize long-term returns, but often it is not optimal. Commonly used solutions are the ε-greedy method and the Softmax method(Tokic & Palm, 2011).

This article uses the ε-greedy algorithm to explore the balance of exploration and exploitation. During the exploration, the next step is randomly selected with the probability of $\varepsilon(0 \le \varepsilon \le 1)$, and the probability of $1 - \varepsilon$ is chosen to bring the maximum long-term return behavior.

4.3 KNN-Q algorithm implementation

4.3.1 State division

In traditional Q learning, continuous state types need to be divided into several states according to their desirable range. Let the network status be BD_t, buffer B_t, and the previous

video clip quality q_{t-1} be divided into M, N, and L states, as shown in the figure 1. However, the traditional Q-learning division has a state problem: if the state division of fine enough, model can be more accurately describe the problem, but the convergence is slow; if the state is divided size is too large, even though it can speed up the iterative speed, but not so for environmental modeling accurate. How to accurately divide the state becomes a difficult point.

As shown in Figure 2, the KNN-Q learning proposed in this paper is different from the traditional state division. When the value is exactly the intermediate value of each interval, the state is found. Otherwise, the Q value of the state is calculated according to the Q value of the neighboring state. The distance d between the total states is calculated from the Euclidean distance of each state component, as shown in equation 10.

$$d_i = \sqrt{(d_i^{bandwidth})^2 + (d_i^{buffer})^2 + (d_i^{previous-ssim})^2} \tag{10}$$

After the state s_t is obtained from the proximity state of the state in each table, the K minimum distance state is selected therefrom. Use their Q-value array to compute an array of Q values for s_t, as shown in equation 11.

$$Q(s_t,:) = \begin{cases} Q(s_t,:), s_t \in S \\ \sum_{i=1}^{K} w_i Q(s^{(i)}, a), s_t \notin S \end{cases} \tag{11}$$

$s_t \in S$ stands for s_t in the presence table, $s_t \notin S$ stands for state s_t not available in the status table. $s^{(i)}$ is a state adjacent to the state s_t. w_i represents the weight of the neighboring state $s^{(i)}$. If $s^{(i)}$ is less than the distance d_i from s_t, the larger the w_i. Conversely, if the distance of $s^{(i)}$ from s_t is larger, the smaller the w_i is, the specific calculation method is shown in equations 12 and 13.

$$w_i = \frac{\rho_i}{\sum \rho_i} \tag{12}$$

$$\rho_i = \frac{1}{d_i} \tag{13}$$

4.3.2 Q table update

After the execution of the action is completed, the Q table needs to be updated based on the returned reward and the new state. If the current state can be found in the state partition table, the Q table is updated using equation 1. If the current state cannot be found in the state partition table, the Q values of the K adjacency states in the Q table are updated,as shown in equation 14.

$$Q(s^{(i)}, a)_{t+1} \leftarrow Q(s^{(i)}, a)_t + \lambda\theta w_i \tag{14}$$

Among them, the calculation method of θ is shown in equation 15. $s^{(i)'}, i = 1, ..., K$ is the K state adjacent to the state s_t next state. The pseudo code of the bir rate adaptive algorithm based on KNN-Q learning in the training stage is shown in Table 2.

$$\theta = R_{total}^{(s,a)} + \gamma max x_{a'} \sum w_i' Q(s^{(i)'}, a') - \sum w_i Q(s^{(i)}, a) \tag{15}$$

5 Simulation

In this paper, a series of simulation experiments are carried out through Matlab to evaluate and compare the performance of KNN-Q learning and traditional Q learning algorithms proposed in this paper. We use Matlab to write a program to simulate the process of the DASH client requesting to download and play the video to the server. This article sets the DASH client to simulate different scenes for different experiments under different bandwidth conditions, the server has different rate video. The K-value sensitivity of KNN-Q learning proposed in this paper is compared and analyzed.

5.1 Video evaluation indicator

It will cost lots of time to calculate the SSIM value frame by frame. For convenience, this paper approximates the SSIM value of the video quality evaluation index to the polynomial (Zanforlin, Munaretto, Zanella, & Zorzi, 2014) with the relative value of the video relative bit rate as the independent variable. The SSIM values of video segments of different bit rates can be obtained from equations 16, 17.

$$\rho_i = log\left(\frac{R_i}{R_1}\right) \tag{16}$$

$$SSIM_i \simeq 1 + d_{(1,v)}\rho_i + d_{(2,v)}\rho^2 + d_{(3,v)}\rho^3 + d_{(4,v)}\rho^4 \tag{17}$$

Where R_1 is the source bit rate of the video and R_i is the bit rate compressed by the server. In equation 17, the vector $\left[1, d_{(1,v)}, d_{(2,v)}, d_{(3,v)}, d_{(4,v)}\right]$ represents the complexity of the video.

5.2 Test video

In the simulation experiment, the test experiment video consists of multiple video scenes. The video complexity of a single video scene is constant, and each scene duration satisfies a random exponential distribution. The test video consists of 800 video clips, each of which has a content duration of 2 seconds.

The video material used to compose the test video in the simulation experiment is derived from the video set (*Xiph.org Video Test Media*, n.d.) provided by EvalVid CIF. The relationship between ρ_i and SSIM values for each video clip in the video clip is shown in Figure 3.

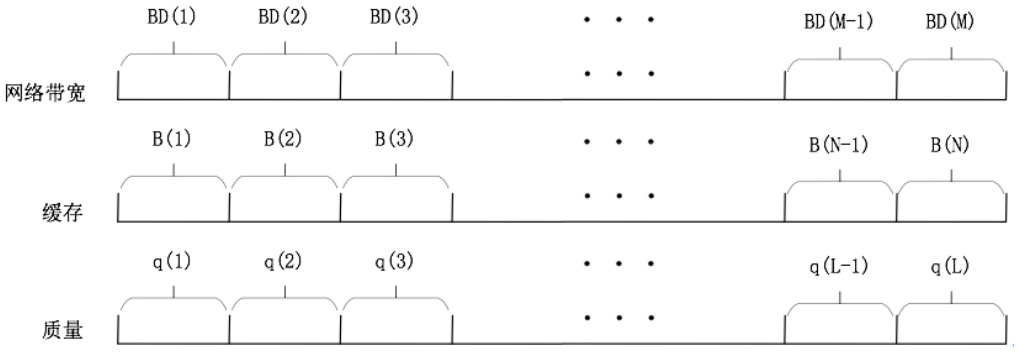

FIGURE 1: Q learning state division

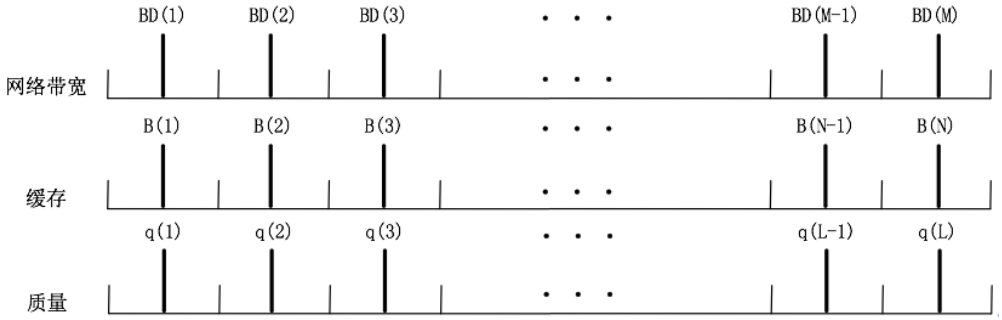

FIGURE 2: KNN-Q learning state division

In this paper, five representative video materials (marked with thicker lines in the 3) are selected from the video material library, and the whole test video is spliced by random combination. The five video materials are Brutta, News, Bridge (far), Harbour and Husky. The complexity coefficients of the five video materials obtained by the maximum likelihood estimation are shown in Table 3.

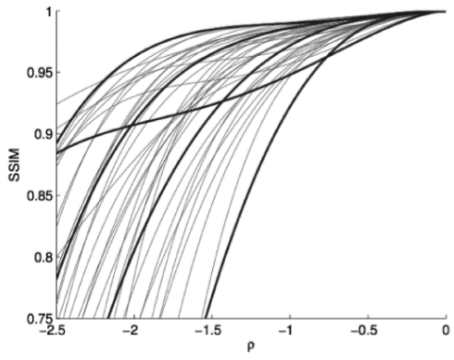

FIGURE 3: Video material bit rate - quality curve (Zanforlin et al., 2014)

5.3 Simulation experiment parameter

The values of the parameters of each scene in the simulation experiment are shown in the Table 5. Where η, λ are the learning rate and reward discount factor for equation 1, respectively. K is the number of adjacency states in KNN-Q learning, and B_{max} is the upper limit of the buffer of the DASH client.

To avoid accidental phenomena, 10 replicate experiments were performed. Each set of experiments include 50 iterations of training and one test with 150 iterations. Each iteration includes 800 steps of training or testing.

5.4 Experimental results and analysis

During the download of the video by the DASH client, the buffer size will have an impact on QoE, and the SSIM value is also an important indicator for measuring video quality. When there is too much buffer, the video buffer will overflow the buffer area; when the buffer is too small, the buffer starvation will occur in the environment with poor network conditions, then the video picture will freeze.

5.4.1 Simple scene

The simple scene simulates a situation where the network bandwidth is relatively stable, but the video content is rel-

TABLE 2: KNN-Q training algorithm pseudo code

Input:The current environment information
Output:The learned Q table

1.**Initialize**: **SET** Q(S,A) arbitrarily;
2.**REPEAT**(for each episode)
3. s_t = Observe current state;
4. **REPEAT**(for each step of episode)
5. Choose Q value array Q(s_t,:) from Q(S,A) with formula (11)
6. Confirm bitrate to download with Q(,:) using ε-greedy policy
7. Request to download the video segment of the bitrate above;
8. Update Buffer $buffer_t$;
9. Calculate the Reward R_t with formula (9);
10. IF $s_t \in S$
11. Update Q(S,A) with formula (1);
12. ELSE
13. Update Q(S,A) with formula (15);
14. END
15. s_{t+1} = Observe next state;
16. $s_t = s_{t+1}$;
17. **UNTILE** reach the end of an epiode
18.**UNTILE** run out the episodes

TABLE 3: Complexity coefficient matrix of 5 video materials

Video name	Complexity coefficient matrix $\left[1, d_{(1,v)}, d_{(2,v)}, d_{(3,v)}, d_{(4,v)}\right]$
Brutta	[-0.0101529,-0.0288832,-0.0242726,0.0041539]
News	[-0.0106444,-0.0229079, -0.0253096, 0.0007417]
Bridge(far)	[-0.01050829,-0.0538481, -0.0821086, 0.0136133]
Harbour	[-0.0050534,0.0055396, -0.01726018, 0.0002203]
Husky	[0.0099785,0.0759046,-0.0113807,0.0003986]

TABLE 4: SSIM value at each bit rate of the video clip 5

Video \ Bit rate	10000	6000	4000	3000	2000	1000	500	300
Brutta	1	0.99765	0.99554	0.99403	0.99215	0.98977	0.98750	0.98425
News	1	0.99851	0.99657	0.99487	0.99209	0.98591	0.97584	0.96352
Bridge(far)	1	0.99382	0.98504	0.97767	0.966578	0.94795	0.93211	0.92284
Harbour	1	0.99880	0.99647	0.99376	0.98808	0.97169	0.94359	0.91266
Husky	1	0.99838	0.99334	0.98641	0.97046	0.92216	0.84148	0.758424

TABLE 5: Simulation experiment general parameter

Parameter name	Value
η	0.3
λ	0.95
ε	0.3
K	2
B_{max}	20sec
TrainEpisodes	50
TestEpisodes	150
StepsOfEpisode	800

atively simple. For example, the current user is playing a video of the endless blue sky, and the scene in the video is almost unchanged. Therefore, in the simulation experiment, only one video material News is used in this scene to generate a test video with a single complexity.The range of network bandwidth changes in a simple scene is $[5000, 6000]$ kb/s.

The comparison between the buffer and video quality of the DASH client in a simple scene is shown in 4 and 5. In a simple scene, the rate-adaptive algorithm based on KNN-Q learning has the same growth trend as the adaptive algorithm of traditional Q-learning, but the average buffer of 10 experiments is not as large as the adaptive algorithm based on Q-learning.

In a simple scene, as shown in 6 and 7, the average video quality of the code rate adaptive algorithm based on KNN-Q learning is higher than that of the traditional Q learning.

5.4.2 Regular scene

The regular scene simulation shows the playing video scene in life, that is, the user bandwidth is relatively stable, and the content played by the user is relatively rich.In the regular scene, the video video with the complexity of the video is generated using the five video clips in Table 4. The network bandwidth varies from $[5000, 6000]$ kb/s.

It can be seen from the Figure 10 and 11 that in the regular scene, the buffer growth trend of KNN-Q learning and traditional Q learning is the same. But the 10 average experiments are not bufferd based on Q learning, and the difference between the two is nearly 7s.

It can be seen from the Figure 10 and 11 that in the regulr scene, the average video quality of the bit rate adaptive algorithm based on KNN-Q learning is higher than that of the traditional Q learning.

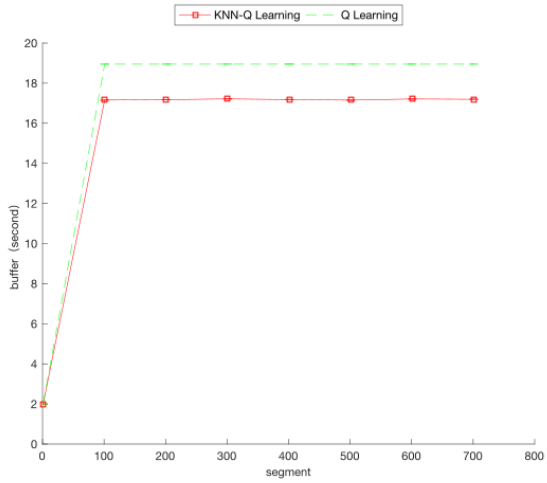

FIGURE 4: Buffer trend of simple scene

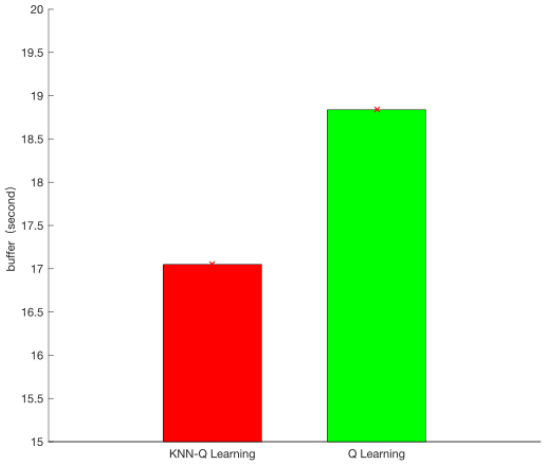

FIGURE 5: Average buffer of simple scene

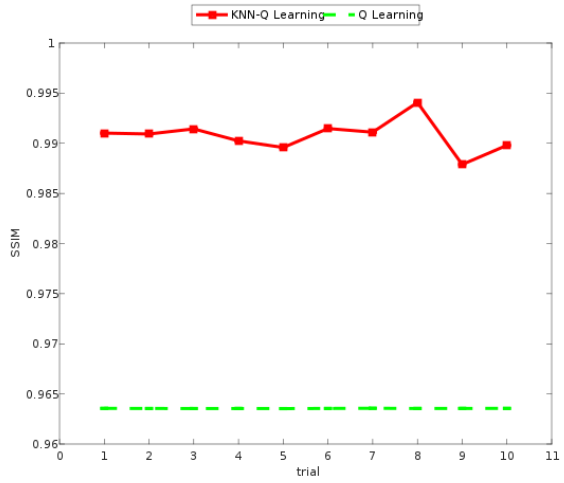

FIGURE 6: Quality comparison of simple scene

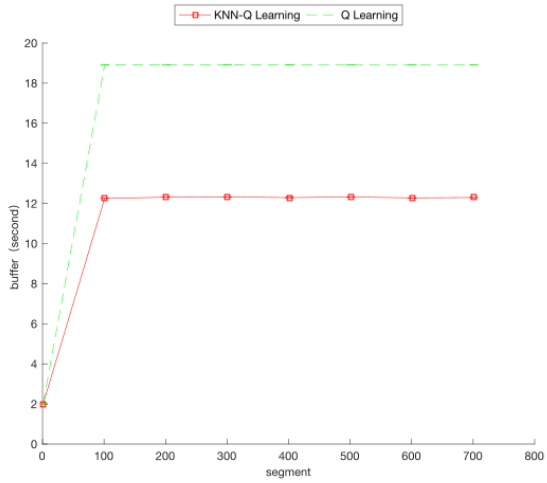

FIGURE 8: Buffer trend of regular scene

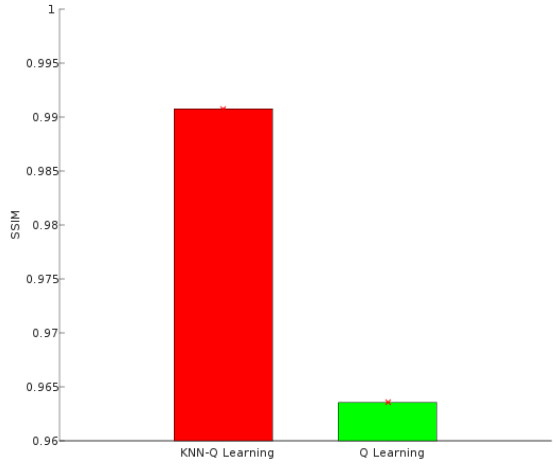

FIGURE 7: Average quality of simple scene

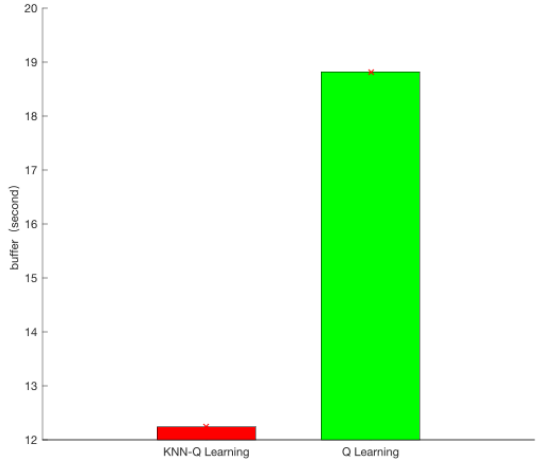

FIGURE 9: Average buffer of regular scene

5.4.3 Complex scene

The complex scene simulates the situation in which real-life network bandwidth fluctuations are large and video content is rich. In the simulation experiment, the range of the simulated bandwidth in the complex scene is $[400, 12500]$kb/s, and the test video is randomly synthesized from the five video materials in Table 4.

It can be seen from Figure 12 and 15 that in a complex scene, the buffer growth trend of KNN-Q learning and traditional Q learning is the same, but the 10 average experiments are not as large as the Q-based buffer.

It can be seen from the Figure 14 and 15 that in the complex scene, the average video quality of the bit rate adaptive algorithm based on KNN-Q learning is higher than that of the traditional Q learning, and the average SSIM value is close to 0.08.

5.4.4 The sensitivity of K

In the regular scene, set the distance formula to the Euclidean distance, and take K = 2, 3, 4, 5, and 6 to determine the influence of the K value in KNN-Q on the experimental results. The test video was randomly synthesized from the five videos in Table 4. As can be seen from Figure 16, the larger the K value, the larger the buffer. As can be seen from Figure 17, the larger the K value, the smaller the average value of the SSIM and the worse the video quality. Especially when K=6, the average SSIM value drops rapidly to below 0.91.

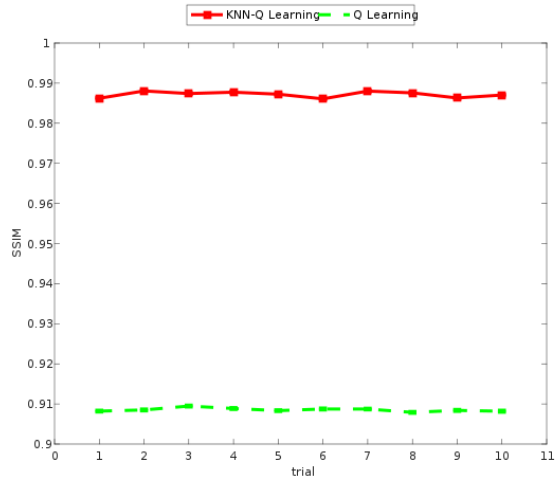

FIGURE 10: Quality comparison of regular scene

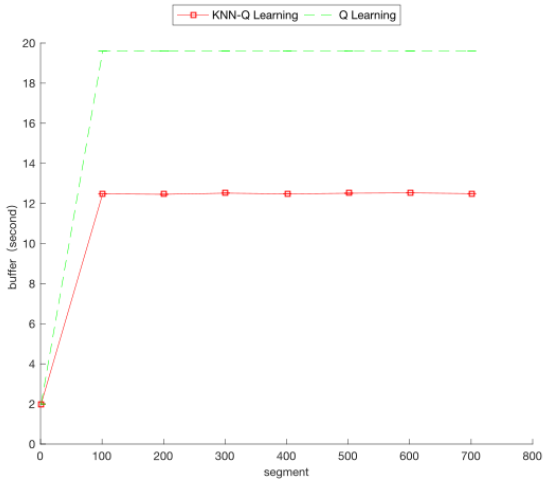

FIGURE 12: Buffer trend of complex scene

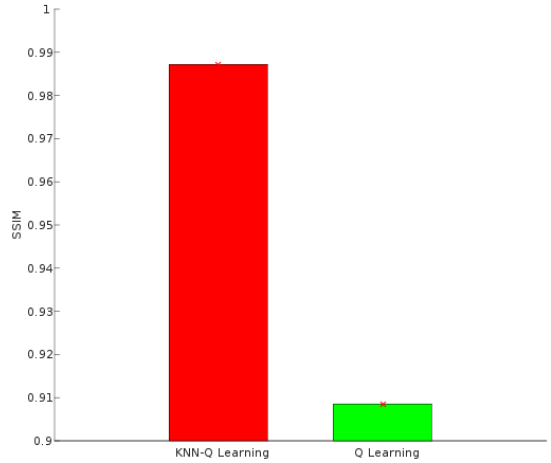

FIGURE 11: Average quality of regular scene

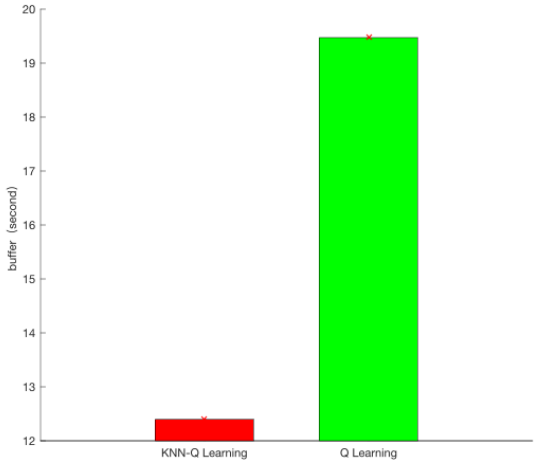

FIGURE 13: Average buffer of complex scene

5.4.5 Distance formula sensitivity

In normal scenario, we set 2 as distance. Euclidean distance formula, Manhattan distance formula and Chebyshev distance formula are used as distance formula in KNN-Q learning to judge the influence of distance formula on experimental results.

As shown in 18, the three distance formulas have little effect on the buffer. As shown in the diagram 19, when the distance formula is the Chebyshev formula, the SSIM value is the lowest, and the SSIM value is the highest when using the Euclidean distance, followed by the Manhattan distance.

6 Conclusion

The Matlab simulation results show that the KNN-Q learning rate adaptive algorithm proposed in this paper can obtain higher video quality than the traditional Q learning rate adaptive algorithm in simple scenes, regular scenes and complex scenes. In the process of training DASH client adaptive strategy, the convergence speed of KNN-Q learning is significantly better than traditional Q learning. In the process of distance and formula sensitivity test, when the Euclidean distance is used and the distance K is 2, the algorithm can obtain higher QoE.

Subsequent attempts will be made to introduce neural networks to replace discrete Q tables to characterize continuous state values.

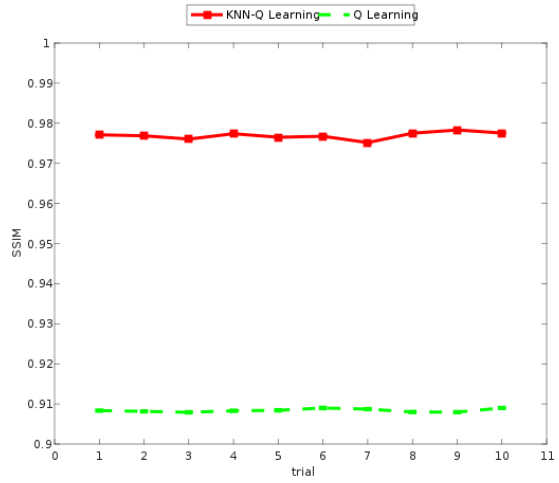

FIGURE 14: Quality comparison of Complex scene

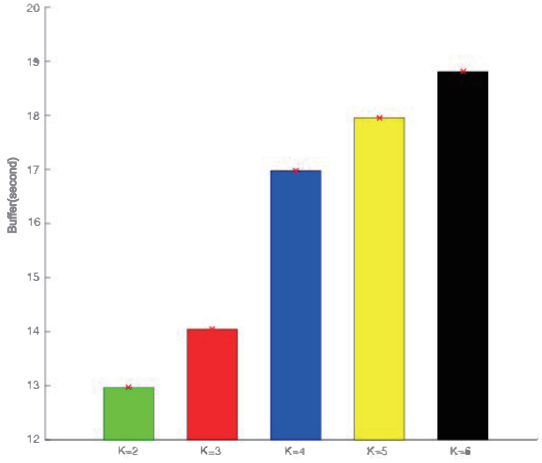

FIGURE 16: The effect of K value on the buffer

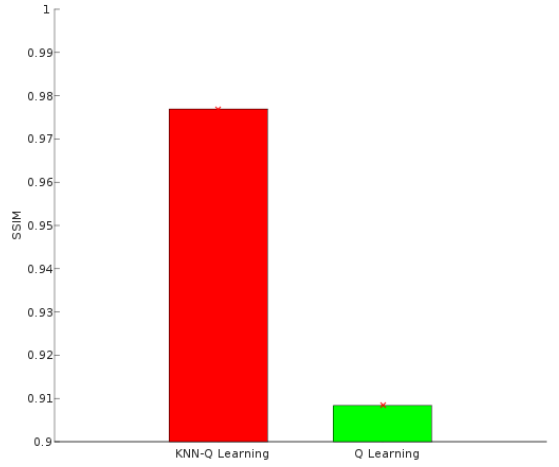

FIGURE 15: Average quality of complex scene

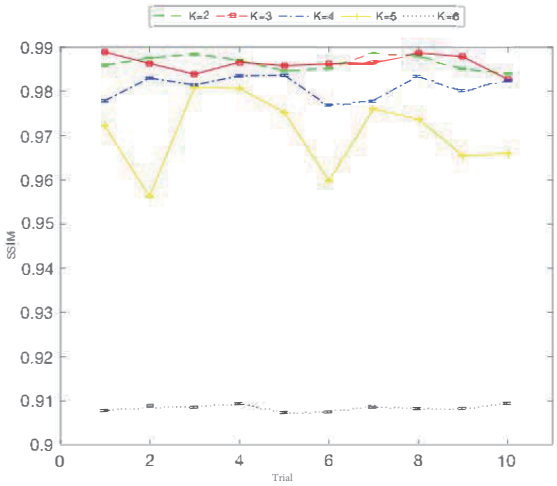

FIGURE 17: The effect of K value on video quality

References

Chien, Y. L., Lin, C. J., & Chen, M. S. (2015). Machine learning based rate adaptation with elastic feature selection for http-based streaming. In *Ieee international conference on multimedia and expo* (p. 1-6).

Claeys, M., Latré, S., Famaey, J., Wu, T., Van Leekwijck, W., & De Turck, F. (2013, 05). *Design of a q-learning based client quality selection algorithm for http adaptive video streaming*.

Gadaleta, M., Chiariotti, F., Rossi, M., & Zanella, A. (2017). D-dash: a deep q-learning framework for dash video streaming. *IEEE Transactions on Cognitive Communications & Networking*, *PP*(99), 1-1.

Hore, A., & Ziou, D. (2010). Image quality metrics: Psnr vs. ssim. In *International conference on pattern recognition* (p. 2366-2369).

Index, V. N. (n.d.). Cisco visual networking index: Global mobile data traffic forecast update, 20162021 white paper.

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *J Artificial Intelligence Research*, *4*(1), 237–285.

Li, Z., Zhu, X., Gahm, J., Pan, R., Hu, H., Begen, A. C., & Oran, D. (2014). Probe and adapt: Rate adaptation for http video streaming at scale. *IEEE Journal on Selected Areas in Communications*, *32*(4), 719-733.

Marinca, D., Barth, D., & Vleeschauwer, D. D. (2013). A q-learning solution for adaptive video streaming. In *International conference on selected topics in mobile and wireless networking* (p. 120-126).

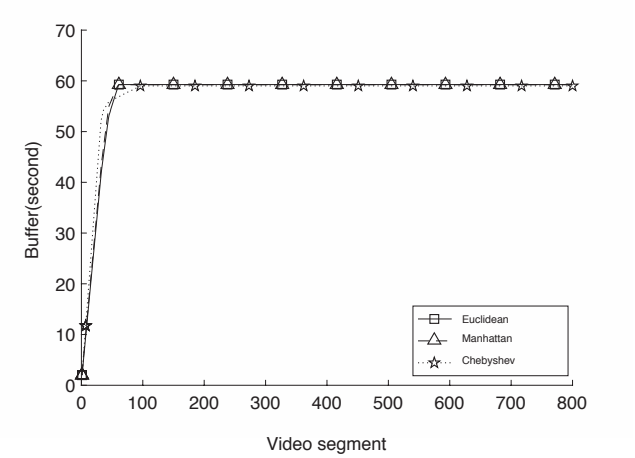

FIGURE 18: Distance formula affects the buffer

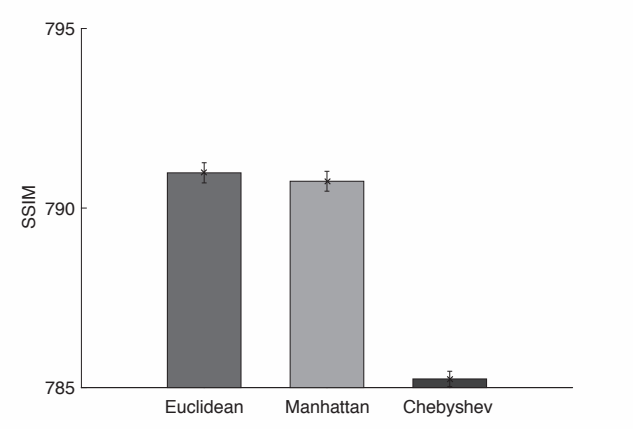

FIGURE 19: Distance formula affects video quality

Mok, R. K. P., Luo, X., Chan, E. W. W., & Chang, R. K. C. (2012). Qdash:a qoe-aware dash system. In *Multimedia systems conference* (p. 11-22).

Pasqualini, S., Fioretti, F., Andreoli, A., & Pierleoni, P. (2009). Comparison of h.264/avc, h.264 with aif, and avs based on different video quality metrics. In *International conference on telecommunications* (p. 190-195).

Sodagar, I. (2011). The mpeg-dash standard for multimedia streaming over the internet. *IEEE Multimedia*, *18*(4), 62-67.

Tokic, M., & Palm, G. (2011). Value-difference based exploration: Adaptive control between epsilon-greedy and softmax. In *Ki 2011: Advances in artificial intelligence, german conference on ai, berlin, germany, october 4-7,2011. proceedings* (p. 335-346).

Wallendael, G. V., Leuven, S. V., Cock, J. D., & Lambert, P. (2013). Evaluation of full-reference objective video quality metrics on high efficiency video coding. In *Ifip/ieee international symposium on integrated network management* (p. 1294-1299).

Xiph.org video test media. (n.d.). `https://media.xiph.org/video/derf/`. (Accessed February 15, 2018)

Xu, Y., Zhou, Y., & Chiu, D. M. (2014). Analytical qoe models for bit-rate switching in dynamic adaptive streaming systems. *IEEE Transactions on Mobile Computing*, *13*(12), 2734-2748.

Zanforlin, M., Munaretto, D., Zanella, A., & Zorzi, M. (2014). Ssim-based video admission control and resource allocation algorithms. In *International symposium on modeling and optimization in mobile, ad hoc, and wireless networks* (p. 656-661).

Zhang, Z., Zheng, Y., Hua, M., Huang, Y., & Yang, L. (2018). Cache-enabled dynamic rate allocation via deep self-transfer reinforcement learning.

Zhu, X., Lany, C., & Schaarz, M. V. D. (2013). Low-complexity reinforcement learning for delay-sensitive compression in networked video stream mining. In *Ieee international conference on multimedia and expo* (p. 1-6).

Zink, M., Schmitt, J., & Steinmetz, R. (2003). Subjective impression of variations in layer encoded videos. In *International conference on quality of service* (p. 137-154).