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

题目

肖刚　　周华康　　陈振波

Abstract

近年来，用户对无缝切换高清流需求的增长和基于 DASH 标准的自适应流的广泛采用，成为码率自适应算法研究的主要驱动力。现有的大部分码率自适应算法通过硬编码实现，无法有效应对网络环境的变化。此外也有学者提出使用可以获得更高 QoE 的 Q 学习的方法来学习得到优化的码率自适应算法。但 Q 学习存在难以编码连续的状态值和在大状态空间下算法收敛速度慢的问题。为此，本文结合了最近邻算法，提出了一个可以处理连续状态的 KNN-Q 学习算法。实验结果表明，在码率自适应情境中，KNN-Q 学习相比普通 Q 学习可以获得更高的 QoE 和更快的收敛速度。关键词:

1　前言

目前，视频数据正在成为传统互联网，尤其是移动互联网的主流数据。视频数据相比于其它数据传输需要更高的比特率，因此带来的移动流量增长更为明显。根据思科全球移动数据流量预测 (2016-2021) 白皮书(Index, n.d.) 显示，到 2021 年，移动视频将增长 8.7 倍，占总移动流量的 78％. 用户对视频质量和交互特性有了更多的要求。

每天都有成千上万的视频通过 HTTP 传输。视频数据通过 HTTP 传输可以轻易通过防火墙和 NAT 设备。各大厂商基于 HAS(HTTP Adaptive Streaming) 技术，也都研发出了适用于各自设备或者客户端的自适应流媒体传输技术，如 Apple 的 HLS(HTTP Live Streaming)，Microsoft 的 MSS (Microsoft Silverlight Smooth Streaming) 以及 Adobe 的 HDS(HTTP Dynamic Streaming)。MPEG 组织在 2011 年制定了 DASH(Dynamic Adaptive Streaming over HTTP) 标准，旨在统一在不同设备和服务器制造商之间传送动态码率的视频技术标准(Sodagar, 2011)。相比于 HLS 和 MSS，DASH 标准不仅支持多种编码方式，还支持多种厂家的 CDN 对接。

DASH 标准的设计思想是：服务器将原始视频切分成内容相同而码率不同的视频片段，然后生成一个包含所有视频片段信息和他们下载地址的索引文件 MPD(Media Presentation Description)。客户端下载并解析 MDP 文件后，在播放过程中根据当前的视频质量和网络情况切换合适码率的视频片段下载(Xu, Zhou, & Chiu, 2014)。DASH 标准开放了码率自适应算法的设计，使开发者可以实现自己的码率自适应算法来提供更高的 QoE(Quality of Experience)。

现有的码率自适应算法大都基于启发式算法，该算法通常使用硬编码的方式根据网络情况选择合适的码率，会导致无法有效应对网络环境频繁波动的情况(Claeys et al., 2013)。Dana 等人提出了采用强化学习中 Q-Learning 算法来得到优化过后的码率自适应策略(Li et al., 2014)。该方法相比于启发式算法可以得到更高的 QoE，但在状态划分过于多时，该方法存在难以学习连续的状态值和学习收敛速度慢的不足。针对该问题本文结合最近邻算法提出了一种 KNN-Q 学习算法作为获取获取视频片段的策略。实验结果表明 KNN-Q 学习算法可以获得更高的 QoE。

本文的组织结构:.....

2　相关工作

客户端接受流媒体必须能够针对网络情况，客户端状态和用户体验自适应选择码率，才能在不同

环境下可靠高效地接受视频。

有研究(Zink, Schmitt, & Steinmetz, 2003) 表明用户对画质的提升不敏感，但是常常能够察觉到画质的下降。应该尽量避免频繁改变画质，即使不能避免画质变化，也需要控制在较小的幅度内变化。与 QoE 相关的自适应策略往往需要考虑多个方面如可用带宽、缓存、画质等，以准确反应用户所感知到的 QoE。Mok(Mok, Luo, Chan, & Chang, 2012) 等人提出了结合 QoE 且基于启发式的码率自适应算法，QDASH-abw 模块位于服务器的代理中用以测量当前带宽下可以支持的最高视频质量，客户端基于 QDASH-qoe 算法选择最合适的视频质量等级。他们的的实验结果表明用户更喜欢在最好和最坏的视频质量之间平缓过度，而不是突然切换。Zink(Zink et al., 2003) 等人的 NOVA 自适应策略提出了"cross-layer" 旨在权衡资源分配和码率自适应，使用之前 S 个视频片段的质量和时间的方差作为衡量 QoE 的标准。上述基于启发式的码率自适应算法，无法方便地建立可预测的数学模型，码率选择策略的灵活度较低，在网络情况复杂的环境中，码率选择算法的稳定性较差。

引入机器学习方法，客户端可以在无人干预的情况下根据前后环境的变化，学习如何调整所请求的视频分块的比特率。使用机器学习的方法可以增强客户端码率自适应策略的自适应性和灵活性。Chien(Chien, Lin, & Chen, 2015) 等人基于随机森林决策树模型将网络状态特征值 (如最新分片的带宽，会话的平均带宽和带宽的波动等) 映射到优秀的码率自适应算法所得到的码率标签，最后使用训练好的分类模型预测视频片段的比特率大小。基于有监督学习的一个缺点在于需要准备训练数据，虽然 Chien 的算法可以通过在线学习的方法收集用户的数据，但是由于环境、用户需求和终端设备的多样性，训练出一个针对所有情景鲁棒性较强的模型还是比较困难的。

将强化学习引入 DASH 的码率自适应算法中，可以使 HAS 客户端不断与环境交互，根据当前网络环境的反馈训练自身的策略，动态学习选择最佳动作。相比于有监督学习，强化学习不需要准备大量训练数据，只需要在学习时定义合理的动作回报函数。Marinca(Marinca, Barth, & Vleeschauwer, 2013) 等人在 MAC 层上实现了一个 HAS 自学习系统，该系统使用部分可见的马尔可夫模型建模，基于 Q 学习设计了一个在线学习的码率自适应算法。Zhu 等人(Zhu, Lany, & Schaarz, 2013) 提出了一个无限时间的马尔可夫模型，在训练过程中使用 Virutal Experience(VR) 策略更新 Q 表。该方法的实验结果表明 VR 策略加速了模型收敛的过程，相比于近似最优的启发式算法在网络状态多变的环境中总体上获得了更高的奖励和显著降低了延迟。Zhang(Zhang, Zheng, Hua, Huang, & Yang, 2018) 等人将 QoE 的准则定位成总累计比特率、丢包率和视频饥饿时长，构建了一个考虑 QoE 的深度自转移强化学习网络 (Self-Transfer learning,STL)。STL 算法综合了一系列伪奖励用以训练虚拟决策中心，然后虚拟决策中心不断将学习到的自适应决策传递给原始决策中心，实验结果表明 STL 算法可以提高训练效率和泛化性能。在结合深度学习和强化学习的特性之后，Gadaleta 等人(Gadaleta, Chiariotti, Rossi, & Zanella, 2017) 利用了前馈神经网络和递归神经网络的优点设计了一个 D-DASH 框架。D-Dash 框架使用视频片段质量和视频重缓事件存评价性能在模拟和真实的环境之下获得了相当高的 QoE。

在使用强化学习解决码率自适应的情境之中，需要权衡算法收敛速度和状态划分粒度。状态划分过多，可以增强系统的健壮性，但会导致迭代时收敛速度过慢；状态划分过于离散，对问题情境的刻画达不到理想状态，这时收敛速度加快。针对码率自适应情境中，状态都是连续的值，本文提出了一种结合最近邻和 Q-learning 的算法。从流媒体的视频质量考虑，综合选择与 QoE 相关的三个方面：当时视频片段质量、前后帧视频质量变化和缓存区溢出危险性构建回报函数。实现了基于 KNN Q-learing 的可以对连续状态进行码率自适应动作的算法。并且设计了一个仿真实验用于比较普通 Q 学习和本文所提出的 KNN-Q 学习的性能表现。

3 基于 KNN-Q 的码率自适应算法

普通基于时序的 DASH 客户端会在请求下一个视频片段时使用探索策略来选择合适的码率。本文使用了基于强化学习的智能体来完成这个探索的过程，帮助 DASH 客户端来进行决策。

强化学习是机器学习的一种，由于对智能体来说环境部分可见，导致智能体应对环境做决策的时候的不确定性。但是与环境互动是智能体唯一学习和成长的方法。智能体在每一步动作之后都会接收到来自环境的奖励，智能体理解了奖励之后进行下一步的决策。智能体的目标是在给定环境之中选取最优的行动以最大化累积奖励。(Kaelbling, Littman, & Moore, 1996)

在码率自适应算法中，定义 DASH 客户端为智能体，智能体观察到的环境值定义为状态 s，DASH 客户端在某一时刻对某一码率的视频片段请求下载记为动作 a。当开始探索时，智能体会更新当前环境的状态。在 Q 学习中，Q 表相当于智能体的大脑来帮助智能体做出决策。Q 函数用于评价当前状态下每个动作的好坏。如等式 1 所示智能体在状态 s 选择动作 a，并且获得环境所给的奖励 r 之后如何更新 Q 值。其中 $\eta \in [0;1]$ 为学习率，代表新值取代旧值的程度；$\lambda \in [0;1]$ 为折扣因子，衡量未来奖励的重要程度。在环境状态划分过细的情况下，使用该方法更新 Q 表，会带来收敛速度过慢的问题。码率自适应情境下环境状态都是连续值，本文基于 KNN 算法，将当前环境状态前后连续的若干个状态按一定策略一起更新，可以达到加快迭代速率和获得更高累积奖励的效果。(参见 3.4)

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

每次迭代训练智能体根据之前学习到的 Q 值，选择合适码率视频片段下载，具体的选择策略参见 3.3。环境当前动作所给的奖励根据奖励函数计算 (参见 3.2)。在多次迭代训练之后，会生成一个收敛的 Q 表。DASH 客户端根据当前所处环境选择合适码率下载视频的时候，查询最终生成的 Q 表进行决策。下载当前视频片段后会将视频片段加入缓存队列，用以应对网络异常等特殊情况，最后播放视频队列。

3.1 环境建模

DASH 客户端需要收集当前可见的环境状态来构建相应的回报函数，并根据相应的策略做出动作。下面将对环境状态建模和定义 DASH 客户端的动作行为。

3.1.1 环境状态

本文使用三个状态分量来定义环境状态：可用带宽，缓存区总时长和前一个视频片段的视频质量。可用带宽为 DASH 客户端测量出来的可用带宽；缓存区总时长代表 DASH 客户端缓存视频片段的填充状态。这三个状态分量最终合成的状态定义为 $s_t = (BW_t, B_t, q_t - 1)$。具体的状态分量的取值范围和离散级别如表1所示。

Table 1: 环境状态定义

状态元素	范围	级别
Bandwidth	$[0; BW_{max}]\ bps$	$N+1$
Buffer filling	$[0; B_{max}]\ sec$	$\frac{B_{max}}{T_{seg}}$
Quality Level(SSIM)	$[-1; 1]$	N

如表 1 所示，BW_{max} 代表最高可用带宽，B_{max} 表示缓存区最多可以储存的视频时长，N 表示码率等级总个数。前一个视频片段质量的计算方法采用 SSIM 评价 (计算方式参见 3.2.1 小节)。

3.1.2 动作

将 DASH 客户端的动作定义为对某一码率的视频片段请求下载。可供 DASH 客户端选择视频片段的下载码率，由 DASH 客户端的转码模块事先确定。

3.2 回报函数

在强化学习中，回报函数是智能体学习策略经验的基本指南。在流媒体自适应情境中，回报函数一般用来衡量 QoE，但是评价 QoE 的高低往往带有主观性。比如，无法比较一个在高低质量之间波动的视

频和一个稳定在中等质量视频的好坏。所以使用数学模型来准确衡量 QoE 很有必要。本文从当前视频片段质量，视频片段质量波动和缓存区溢出危险性三个方面来构建回报函数。

3.2.1 视频质量

本文使用广泛应用于评价视频质量的SSIM作为评价视频质量的指标(Hore & Ziou, 2010; Wallendael, Leuven, Cock, & Lambert, 2013; Pasqualini, Fioretti, Andreoli, & Pierleoni, 2009)。SSIM 是衡量两张图片相似度的指标，利用结构信息来评估图像质量。SSIM 的具体计算方式如等式 2 所示，对于给定的图像 X 和 Y, 利用两个图像中的像素点亮度的均值、方差和协方差来计算 SSIM 值。其中 μ_X 和 μ_Y 分别是图像 X 和 Y 的像素点亮度均值,σ_X^2 和 σ_Y^2 分别是图像 X 和 Y 的像素点亮度值的方差,σ_{XY} 是图像 X 和 Y 的像素点亮度值之间的协方差,c_1 和 c_2 是为了避免分母为零而设置的常数。SSIM$\in [-1,1]$ ，当 SSIM 越接近 1 时表示两个图像相似度越高。

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

利用等式 2，计算待下载码率的视频片段和原视频片段各帧的 SSIM 均值用来衡量该视频片段的质量。如等式 3 所示，用视频片段的平均 SSIM 来表示视频片段的质量，其中 m 表示视频片段由 m 帧图片构成,q_n 表示第 n 个片段的视频质量。

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

如等式 4 所示，用第 n 个片段的视频质量来表示 DASH 客户端下载该视频质量的视频片段所带来的回报值。

$$R_{quality} = q_n \quad (4)$$

3.2.2 视频质量振荡

在网络带宽变化时，DASH 客户端请求的视频质量也会发生变化，而视频质量的振荡会带来 QoE 的下降。如等式 5 所示，视频播放期间质量切换带来的回报损失记为 $R_{quality-change}$, 根据振荡的步长 α 映射成惩罚回报。

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

3.2.3 缓存区安全性

本文设置该惩罚项目的在于度量缓存区安全性，如等式 6、7 和 8 所示，该惩罚项由安全缓存水平惩罚项和低缓存惩罚项组成。其中 B_{max} 表示缓存区最大缓存容量，B_n 表示在下载第 n 个视频片段时缓存区拥有的视频片段的时长, β 和 γ 代表了对缓存安全和缓存安全的关注程度。在下一个下载动作会引起视频冻结时，$R_{buffer-safe}$ 的值会明显下降，即惩罚会引起缓存饥饿的行为。DASH 客户端缓存区时长越少的时候，$R_{buffer-low}$ 的值越小，即惩罚低缓存的行为。

$$R_{buffer-filling} = R_{buffer-safe} + R_{buffer-low} \quad (6)$$

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

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

3.2.4 总回报函数

总回报函数会推动 DASH 客户端探索较高的码率等级、较少的码率切换和较合理的缓存数据量的行为决策。由于 QoE 往往带有主观性，设置可调的参数很有必要。如等式 9 所示，本文设置了 C_1、C_2 和 C_3 分别对应 $R_{quality}$、$R_{quality-change}$ 和 $R_{buffer-filling}$ 的权重，根据所侧重的 QoE 方面来设置具体的值。

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

3.3 探索策略

强化学习的过程中的动作选择需要注意探索和利用的平衡。智能体为了获得更高的累积最大奖励，往往会选择之前执行过的即时奖励比较大动作；另一方面，智能体需要尝试之前没有被执行过有可能返回更大即时奖励的动作。专注于探索可能会对短期收益产生影响，专注于利用可以使长期收益最大

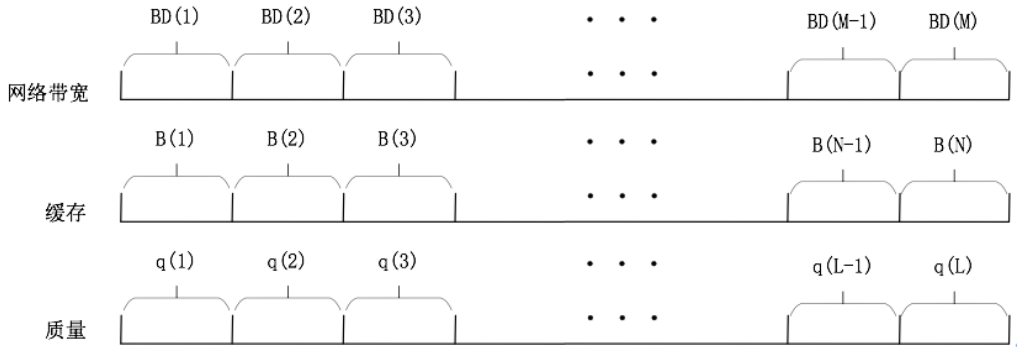

FIGURE 1: Q 学习状态划分

化，但往往可能得到的不是最优解。常用的解决方法有 ε-greedy 方法和 Softmax 方法(Tokic & Palm, 2011)。

本文采用 ε-greedy 算法进行探索和利用的平衡。探索过程中以 $\varepsilon(0 \le \varepsilon \le 1)$ 的概率随机选择下一步的行为，以 $1-\varepsilon$ 的概率选择会带来最大长期回报的行为。

3.4 KNN-Q 算法实现

3.4.1 状态划分

在传统 Q 学习中，需要将连续的状态类型依据其可取范围划分成若干个状态。设网络状态为 BD_t、缓存 B_t、前一个视频片段质量 q_{t-1} 各自划分成 M、N、L 个状态，如图1所示。但传统的 Q 学习存在状态划分问题：如果状态划分足够细，可以更精确地描述问题，但算法收敛速度过慢；如果状态划分粒度太大，虽然能够加快迭代速度，但使对环境建模不准确导致算法表现较差。如何准确的划分状态成为一个难点。

如图2所示，本文所提出的 KNN-Q 学习与传统状态划分不同的是，当取值正好为每个区间的中间值时才认定找到状态，否者该状态的 Q 值根据邻近状态的 Q 值计算得到。总状态之间的距离 d_i 是根据各个状态分量的欧式距离计算得出来的，如公式 10 所示。

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

状态 s_t 与各个表中状态的邻近状态距离得到之后，从中选出 K 个距离最小的状态，用他们的 Q 值数组计算 s_t 的 Q 值数组，如公式 11 所示。

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

$s_t \in S$ 代表 s_t 存在状态表中，$s_t \notin S$ 代表状态 s_t 在状态表查询不到。$s^{(i)}$ 是与状态 s_t 相邻近的状态。w_i 代表了邻近状态 $s^{(i)}$ 其 Q 值的比重。若 $s^{(i)}$ 离 s_t 的距离 d_i 越小，w_i 越大；反之，若 $s^{(i)}$ 离 s_t 的距离越大，w_i 越小，具体计算方式见公式 12、13。

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

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

3.4.2 Q 表更新

动作执行完成之后，需要根据返回的即使奖励和新的状态更新 Q 表。如果当前状态可以在状态划分表中找到，则使用公式 1 更新 Q 表。若当前状态无法在状态划分表中找到，则更新 K 个邻接状态在 Q 表中的 Q 值。

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

其中，θ 的计算方式见公式 15。$s^{(i)'}, i = 1,.....,K$ 是与状态 s_t 下一状态相邻近的 K 个状态。基于 KNN-Q 学习的码率自适应算法在训练阶段的伪代码见表2

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

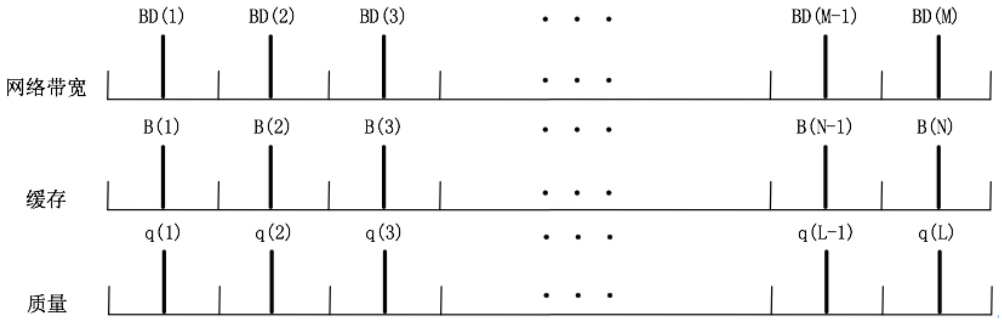

FIGURE 2: KNN-Q 学习状态划分

4 仿真实验

本文通过 Matlab 来进行一系列仿真实验来评估对比本文提出的 KNN-Q 学习与传统 Q 学习算法性能。使用 Matlab 编写程序来模拟 DASH 客户端向服务器请求下载视频并播放的过程。本文设置 DASH 客户端在不同带宽条件下，服务器拥有不同码率视频的条件下来模拟不同的实验所需的场景。并对本文所提出的 KNN-Q 学习的 K 值灵敏度做了对比分析。

4.1 视频评价指标

逐帧计算 SSIM 值非常耗时，为方便起见，本文把视频质量的评价指标 SSIM 值近似为以视频相对码率对数值为自变量的多项式(Zanforlin, Munaretto, Zanella, & Zorzi, 2014)。不同码率的视频片段的 SSIM 值可以由等式 16、17 得到。

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

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

其中，R_1 是视频的原始码率，R_i 是经过服务器压缩的码率。等式 17 中向量 $\left[1, d_{(1,v)}, d_{(2,v)}, d_{(3,v)}, d_{(4,v)}\right]$ 代表了视频的复杂度。

4.2 测试视频

仿真实验中，测试实验视频由多个视频场景组成，单个视频场景的视频复杂度是恒定的, 每个场景时长满足随机指数分布。测试视频包括 800 个视频片段，每个视频片段的内容时长为 2 秒。

仿真实验中用以组成测试视频的视频素材源于 EvalVid CIF 提供的视频数据集(*Xiph.org Video Test Media*, n.d.)。视频素材库中各个视频素材的 ρ_i 与 SSIM 值的关系如图3所示。

本文从视频素材库中选取了具有代表性的 5 个视频素材（在图3中用较粗的线条标出）, 通过随机组合拼接成完整的测试视频。这 5 个视频素材分别是 Brutta, News,Bridge(far),Harbour 和 Husky。通过最大似然估计发得到 5 个视频素材的复杂度系数如表3所示。

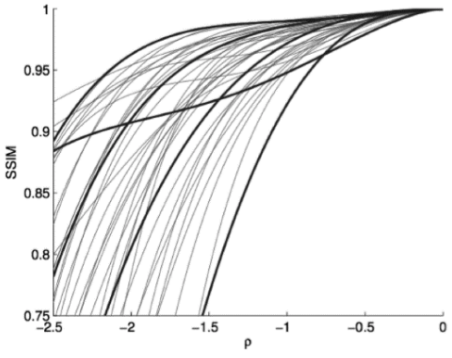

FIGURE 3: 视频素材比特率-质量曲线 (Zanforlin et al., 2014)

TABLE 2: KNN-Q 训练算法流程伪代码

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: 5 个视频素材的复杂度系数矩阵

视频名称	复杂度系数矩阵 $[1, d_{(1,v)}, d_{(2,v)}, d_{(3,v)}, d_{(4,v)}]$
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: 5 个视频素材各个比特率下的 SSIM 值

视频 \ 比特率	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

4.3 仿真实验参数设置

TABLE 5: 仿真实验通用参数表

参数名	取值
η	0.3
λ	0.95
ε	0.3
K	2
B_{max}	20sec
TrainEpisodes	50
TestEpisodes	150
StepsOfEpisode	800

仿真实验各场景的参数取值如表5所示。其中 η, λ 分别是公式 1 的学习速率和奖励折扣系数。K 是 KNN-Q 学习中邻接状态的个数，B_{max} 是 DASH 客户端的缓存上限。

为避免偶然现象，进行 10 组重复实验。每组实验包括 50 个迭代训练和一次包含 150 次迭代的测试。每个迭代进行 800 次步长的训练或测试。

4.4 实验结果与分析

DASH 客户端在下载视频的过程中，缓存大小会对 QoE 产生一定影响，同时 SSIM 值也是衡量视频质量的一个重要指标。缓存过多时，视频缓存会溢出缓存区；缓存过少时，在网络情况较差的环境下会引发缓存饥饿，视频画面冻结。

4.4.1 简单场景

简单场景模拟了网络带宽比较稳定，但视频内容比较单一的情形。比如当前用户正在播放一望无际的蓝色天空的视频，此时视频中的场景几乎无变化。故在仿真实验中，该场景下只利用 1 个视频素材 News 生成复杂度单一的测试视频。简单场景中网络带宽变化的范围为 $[5000, 6000]$，单位 kb/s。

简单场景下 DASH 客户端的缓存和视频质量对比情况如图4、5所示。在简单场景下，基于 KNN-Q 学习的码率自适应算法与传统 Q 学习的自适应算法缓存增长趋势相同，但是 10 次实验的平均缓存没有基于 Q 学习的自适应算法大。

如图6、7所示，简单场景下，基于 KNN-Q 学习的码率自适应算法平均的视频质量比传统 Q 学习的码率自适应算法高。

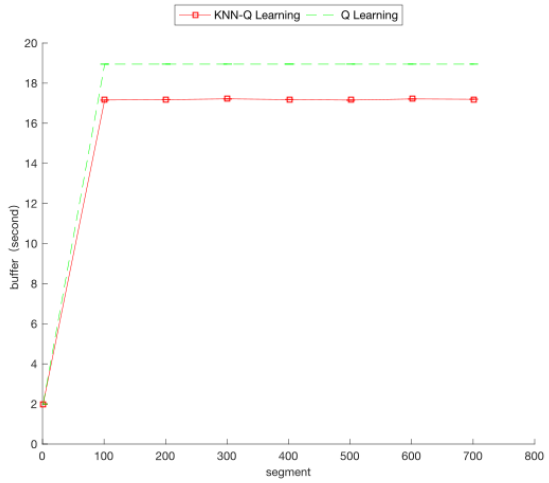

FIGURE 4: 简单场景 buffer 趋势图

4.4.2 常规场景

常规场景模拟显示生活中的播放视频场景，即用户带宽比较稳定，用户播放的内容比较丰富。常规

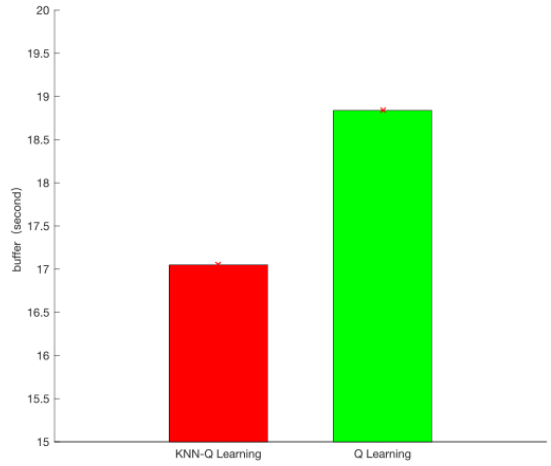

FIGURE 5: 简单场景 buffer 柱状图

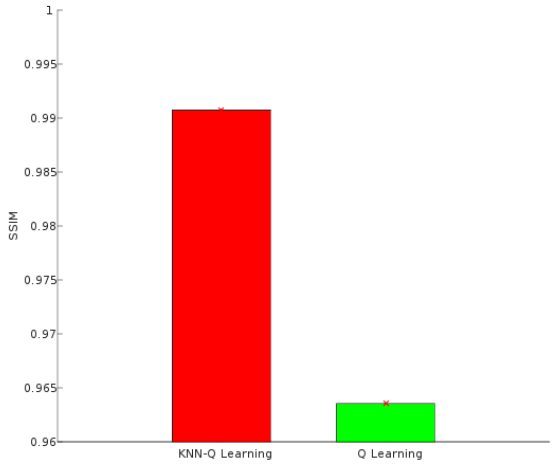

FIGURE 7: 简单场景平均视频质量

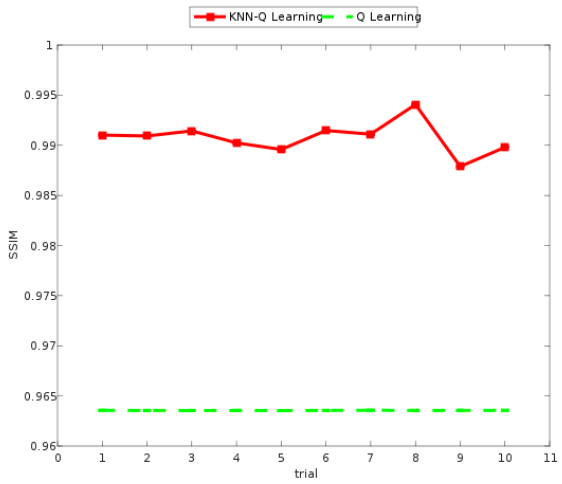

FIGURE 6: 简单场景视频质量对比图

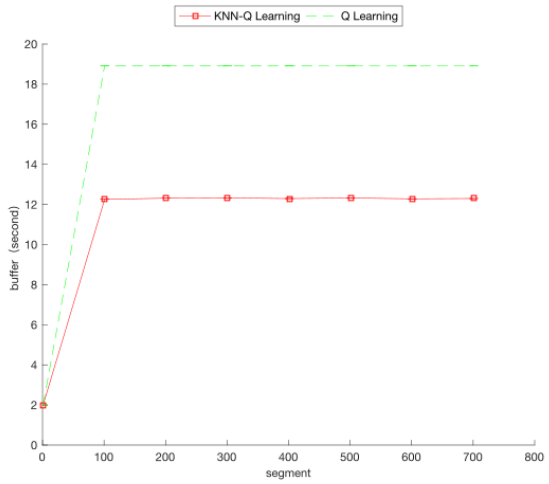

FIGURE 8: 常规场景 buffer 趋势图

场景下利用表4 中 5 个视频素材生成视频复杂度在波动的测试视频，网络带宽变化的范围为 [5000, 6000]，单位 kb/s。

由图8、9可知，在常规场景下，KNN-Q 学习和传统 Q 学习的缓存增长趋势相同，但 10 次平均实验没有基于 Q 学习的缓存大，且两者相差近 7s 的缓存。

由图10、11可知，常规场景下，基于 KNN-Q 学习的码率自适应算法平均的视频质量比传统 Q 学习的码率自适应算法高。

4.4.3 复杂场景

复杂场景模拟了现实生活中网络带宽波动较大，视频内容丰富的情境。在仿真实验中，复杂场景下模拟带宽的范围是 [400, 12500]kb/s，测试视频由表4 中 5 个视频素材随机合成。

由图12、15可知，在复杂场景下，KNN-Q 学习和传统 Q 学习的缓存增长趋势相同，但 10 次平均实验没有基于 Q 学习的缓存大。

由图14、15可知，复杂场景下，基于 KNN-Q 学习的码率自适应算法平均的视频质量比传统 Q 学习

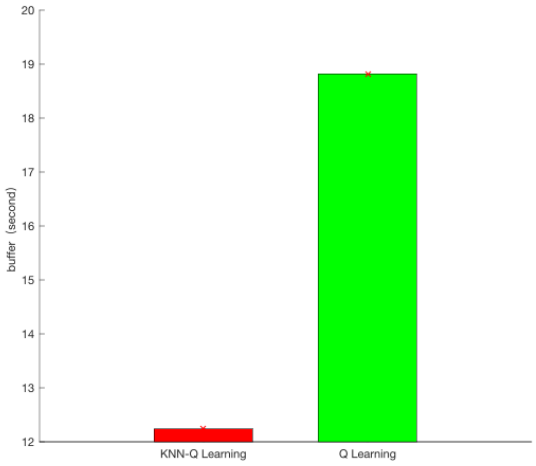

FIGURE 9: 常规场景 buffer 柱状图

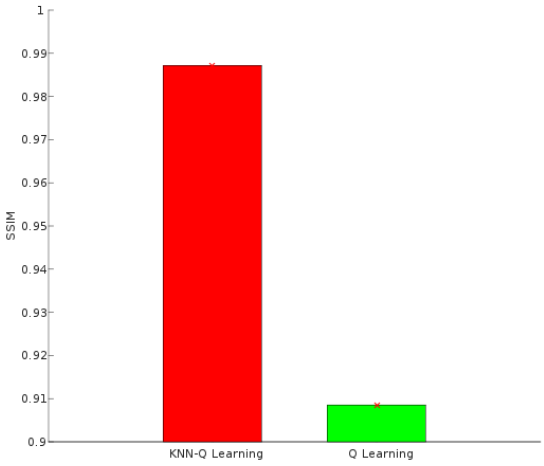

FIGURE 11: 常规场景平均视频质量

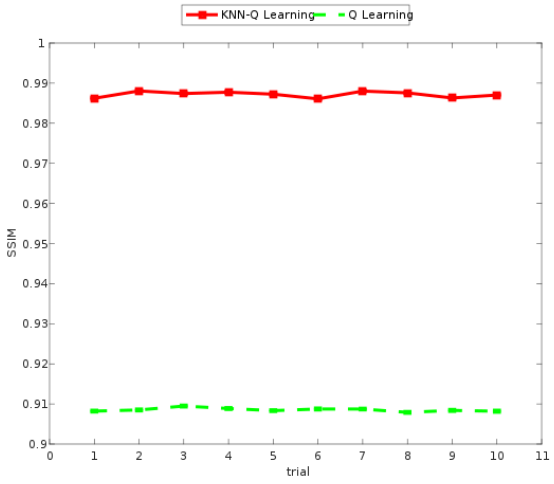

FIGURE 10: 常规场景视频质量对比图

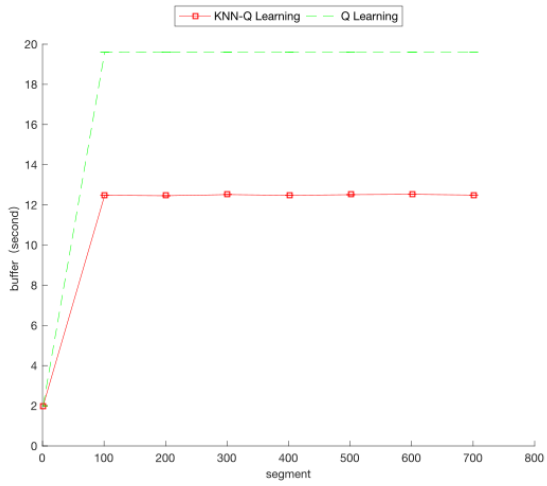

FIGURE 12: 复杂场景 buffer 趋势图

的码率自适应算法高, 平均 SSIM 值相差近 0.08。

4.4.4 K 值灵敏度

在常规场景中，设置距离公式为欧式距离，分别取 K = 2，3，4，5，6 来判断 KNN-Q 中 K 值的大小对于实验效果的影响。测试视频由表4 中 5 个视频随机合成。由图16可知，K 值越大，缓存也越来越大。由图17可知，K 值越大，SSIM 平均值越小，视频质量越差。尤其是当 K=6 时，平均 SSIM 值迅速下降到 0.91 以下。

4.4.5 距离公式灵敏度

在常规场景中，设置 K=2。分别采用欧式距离，曼哈顿距离公式和切比雪夫距离公式作为 KNN-Q 学习中的距离公式来判断 KNN-Q 中距离公式的选择对于实验效果的影响。

如图18所示，三种距离公式对缓存几乎没有影响。如图19所示，当距离公式是切比雪夫公式时，SSIM 值最低，采用欧式距离时的 SSIM 值最高，其次是曼哈顿距离。

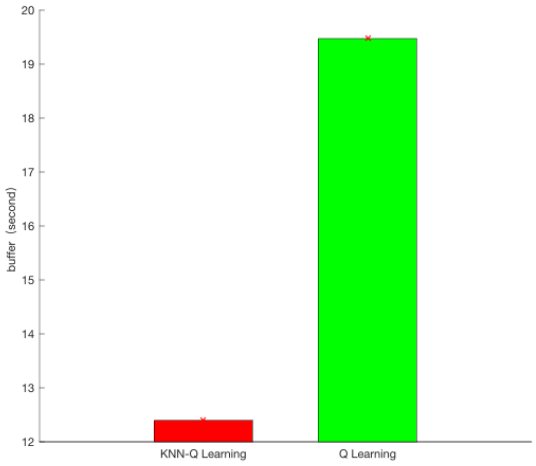

FIGURE 13: 复杂场景 buffer 柱状图

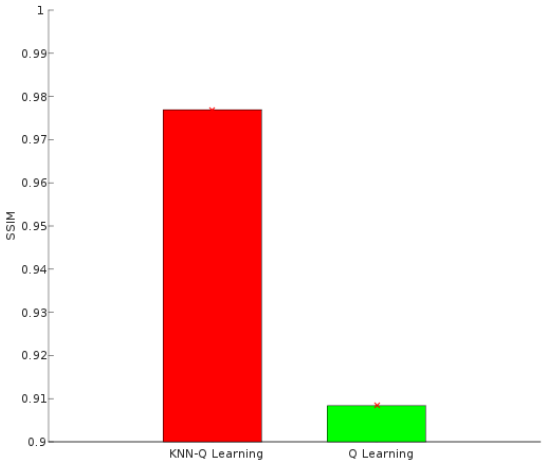

FIGURE 15: 复杂场景平均视频质量

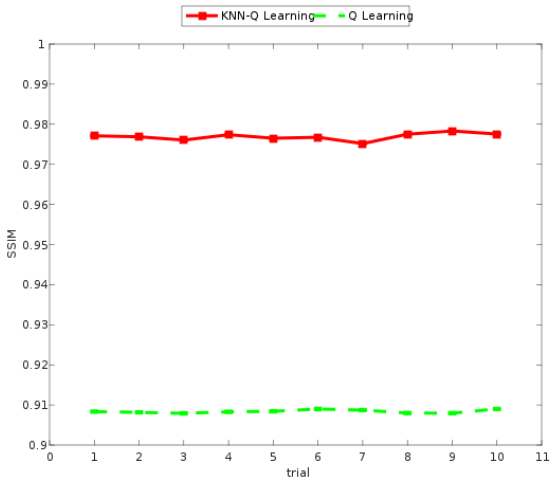

FIGURE 14: 复杂场景视频质量对比图

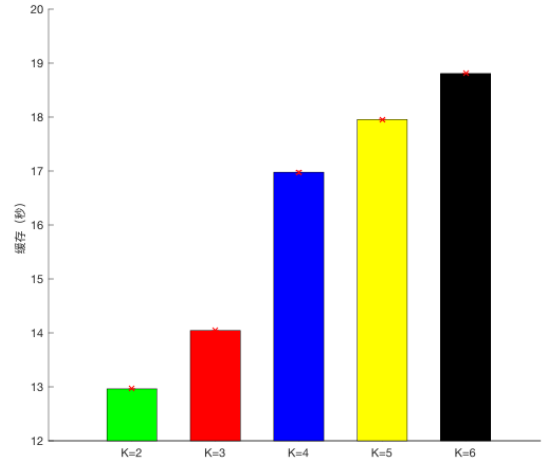

FIGURE 16: K 值对缓存的影响

5 结束语

Matlab 仿真实验结果显示，在简单场景、常规场景和复杂场景中，本文所提出的 KNN-Q 学习码率自适应算法比传统的 Q 学习码率自适应算法上可以获得更高的视频质量。在训练 DASH 客户端自适应策略的过程中，KNN-Q 学习的收敛速度明显优于传统 Q 学习。在距离和公式灵敏度测试的过程中，使用欧式距离，且 K=2 时，算法可以获得更高的 QoE。

后续将尝试引入神经网络以替代离散的 Q 表来刻画连续的状态值。

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.

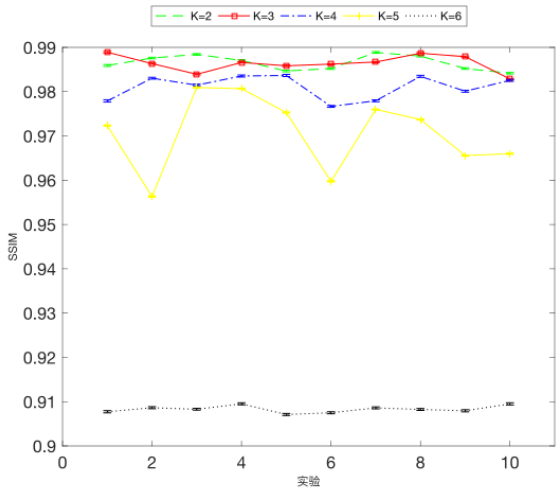

FIGURE 17: K 值对视频质量的影响

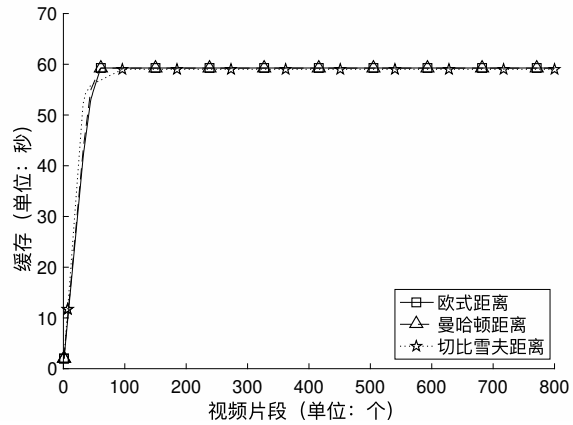

FIGURE 18: 距离公式对缓存影响

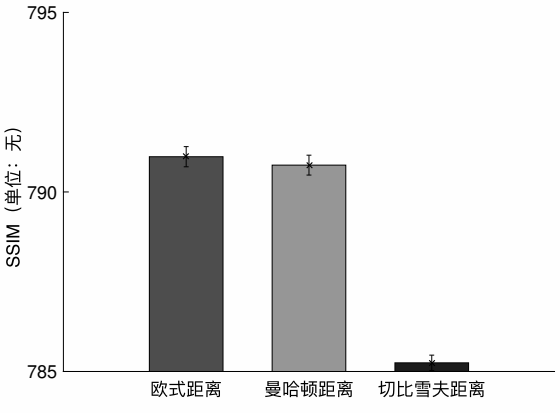

FIGURE 19: 距离公式对视频质量影响

(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, 2016–2021 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).

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 net-*

work 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).