计 算 机 工 程

Computer Engineering

**中图分类号：**

基于 KNN-Q 学习的ABR优化方法研究

肖刚 周华康 程振波

（浙江工业大学 计算机系，杭州 310023）

**摘 要：**在符合DASH（Dynamic Adaptive Streaming over HTTP）标准的视频流媒体播放系统中，客户端获取视频片段的策略对于开发者是开放的，DASH标准对媒体内容格式、传输方式、业务控制等方面都规定了具体的技术指标，但没有规定DASH中对于视频片段获取的策略,因即开放了客户端的ABR(Adaptive Bitrate)算法给开发者。现阶段提出的基于启发式算法和强化学习算法的视频片段获取策略，往往可以再进一步优化。比如基于Q学习的视频片段获取策略，为了加速得到Q表，减少Q表的大小，在训练阶段以比测试训练阶段更大的离散尺度得到Q表，则在测试阶段往往查不到需要的状态值和Q值，这将导致算法无法有效执行下去。本文提出基于KNN-Q学习的视频片段获取策略。实验结果表明，本文提出的KNN-Q学习算法，在训练阶段可以加速Q表的生成，且解决了部署测试阶段无法查找Q表的问题，且具有更高的SSIM值。

**关键词：**HTTP自适应流媒体；基于HTTP的动态自适应流媒；Q学习；KNN-Q学习;结构相似性

英文文章名（须与中文题目意思吻合）

作者名（拼音）

（单位名称，单位所在地 邮编）（英文）

**【Abstract】**（英文摘要）

**【Key words】**HAS;DASH; Q Learning;KNN-Q Learing;SSIM

# 1 引言

# 当前，视频数据正在成为当前传统互联网，尤其是移动互联网的主流数据。根据思科全球移动数据流量预测（2016- 2021）白皮书［1］显示，到2021年，全球移动数据流量的四分之三（78％）将是视频流量。与此同时，用户对视频质量和支持特性提出了越来越多的要求，由此视频传输技术经历了三个阶段，分别是：渐进式下载技术，RTMP(Real Time Messaging Protocol)/RTSP(Real time streaming protocol)流式传输技术以及自适应流式传输技术[2]。

# 在自适应流式传输技术中，HTTP自适应流（HTTP Adaptive Streaming，HAS）技术是其主流技术框架。知名公司也都研发了具体的HAS的视频传输技术，如Apple的 HLS（HTTP Live Streaming），Microsoft的MSS（Microsoft Silverlight Smooth Streaming）以及Adobe的HDS（HTTP Dynamic Streaming）。这些技术的具体实现细节由相应技术厂家规定，存在诸多差异性，为方便规范开发，MPEG国际组织制定了HAS的国际标准DASH（Dynamic Adaptive Streaming over HTTP）。DASH支持“按需播放”、“直播”等不同的视频服务以及支持“码率自适应”、“多国语言字幕”、“广告插入”、“窍门模式”等多种特性[3]，满足用户提出的诸多需求。

# DASH标准对视频内容格式、传输方式、业务控制等方面都规定了具体的技术标准，但与大多数HAS技术一样，DASH标准中并没有规定客户端对于视频片段的获取策略,即开放了客户端的ABR(Adaptive Bitrate)算法[3]给开发者。客户端的ABR算法也一直是近些年的研究热点。

客户端的ABR算法根据其所依据的反馈信号的不同可分为基于吞吐量的ABR算法,基于缓存的ABR算法和基于混合/控制理论的ABR算法[4]。

# C. Liu等人指出客户端请求的视频码率的大小取决于视频片段的获取时间，其增减控制方式类似于TCP的拥塞控制策略－急剧减小和步进式增大[5]。除了考虑视频片段的获取时间，K. Evensen等人提出还应该同时考虑缓存大小［6］。

# 。 于是，Maxim Claeys等人提出利用强化学习中的Q学习算法作为视频片段的获取策略［7］。但由于涉及的状态数太多，存在算法收敛速度慢的问题。为此，Maxim Claeys等人修改相关的函数模型，减少了涉及的状态数，提出FAQ学习算法作为视频片段的获取策略[8]。

当Q学习中涉及的问题的状态是连续值时，需要先将连续的状态值离散化（比如视频片段获取策略问题中涉及到的网络带宽状态），由于状态值的连续特性以及离散程度的不同，往往无法在策略矩阵Q表中直接查询到所需要的Q值数组，**前人的研究没有提出针对此问题的有效解决方案。**针对上述问题，本文提出的KNN-Q学习算法作为视频片段的获取策略。本文通过**设计并进行了对比实验以验证提出的算法性能。**

# 2 背景知识

## 2.1 MPEG-DASH与ABR

****

图1 DASH标准的/系统架构图

图1展示了DASH标准的系统架构图。DASH标准的设计思想是[2]：首先在服务器端利用视频编码模块，对原始高清的视频进行编码处理，生成视频内容相同但是码率不同的视频。接着，利用视频切分模块将拥有不同码率的各个视频都切分成播放时长很短的视频片段（一般其时长占2秒）。随后，服务器端将生成一个描述视频片段的内容、URL地址、编码和其他相关信息的清单文件，称为MPD（Media Presentation Description）文件。然后，DASH客户端发起请求下载MPD文件到本地，传送MPD解析器。DASH客户端可以从MPD文件得知当前视频流数据的时长、类型、分辨率、可选码率、URL等后续请求下载视频片段所需知道的全部信息。DASH客户端可利用这些信息，通过HTTP Get方法，选择请求下载合适码率的视频片段到本地，然后通过媒体播放器逐个播放本地已缓存的视频片段，直至视频内容播放完毕。

## 2.2 MPD与Q学习

## 2.1 数学模型

在DASH自适应算法问题中，定义DASH客户端为智能体Agent，让其中的算法控制器执行KNN-Q学习算法以向HTTP服务器请求下载视频片段。KNN-Q学习算法的模型涉及的要素主要有状态、动作、回报函数和策略矩阵Q表等，下面分别对其进行介绍。

2.1.1状态

将Agent观测到的环境值(如网络、缓存、前一视频片段的SSIM值)的函数值定义为Agent的状态State，简写为S。其中，是网络带宽bandwidth的状态值,是缓存buffer的状态值,是前一视频片段的SSIM的状态值。形式化定义KNN-Q学习算法的状态State的数学模型如公式（1）所示：

(1)

## 2.1.2动作

将Agent的动作Action定义为“对某一码率的视频片段进行请求下载”。其中，可供DASH客户端选择的视频片段的码率，由DASH服务器的转码模块事先确定。动作的选择策略是基于Softmax策略，在第t个视频片段处选择动作a的概率见公式

(2)

其中，e是自然对数， 是温度系数。

## 2.2.3回报函数

定义回报函数模型如公式（3）所示：

(3)

其中，代表当Agent处于状态S时，采取动作a后,评估当前所请求下载的视频片段的质量指标SSIM的回报值。代表当Agent处于状态s，采取动作a后, 评估“所请求下载的前后相邻视频片段的质量指标SSIM差值”的回报值。代表当Agent处于状态s时，采取动作a后，评估“当前Agent所拥有缓存值”的回报值。

## 2.1.4策略矩阵Q表

定义策略矩阵Q(S,A)如公式（4）所示：

(4)

策略矩阵Q表的行代表Agent的状态集合[]，列代表Agent的动作集合[]。策略矩阵Q表中每一个值（即Q值）对应一对状态和动作（即策略）所对应的效用值。Q值更新模型如公式（5）所示：

(5)

其中，是学习速率， 是折扣系数。

## 2.1.5 KNN-Q学习

为了解决由于状态值的连续特性以及离散程度的不同而往往无法在策略矩阵Q表中直接查询到所需要的Q值数组的问题。本文提出KNN-Q学习算法，当Agent处于在Q值矩阵中无法查到所需Q值数组的新状态时，取新状态的 K个最邻近的状态的Q值数组的均值,作为新状态的Q值数组, 再根据计算出的Q值数组确定在状态下应该采取的动作。

# 2.2 算法描述

本文提出的KNN-Q学习算法分为“训练阶段”和“部署测试阶段”。

“训练阶段”主要是为了得到策略矩阵Q(S,A)，供Agent在“部署测试阶段”查询策略。本文中以请求的视频片段的序列号作为时间度量。“训练阶段”的具体的计算步骤见算法1：

算法1: KNN-Q学习算法“训练阶段”算法

|  |
| --- |
| 输入：网络带宽bandwidth,当前缓存buffer,前视频片段的SSIM值  输出：策略矩阵Q表 |
| 1. 初始化segment = 0,round = 0; 2. 初始化网络带宽，缓存; 3. REPEAT 4. 根据网络带宽、缓存、前一视频片段的SSIM值计 5. 算当前时刻agent的状态currentState; 6. REPEAT 7. 根据currentState查询Q表，得到对应的Q值数组； 8. 基于Softmax策略确定下载的视频码率； 9. 发起请求，下载相应码率的视频片段； 10. 更新缓存； 11. 计算即使回报； 12. 计算agent下一个状态nextState； 13. 按公式（？）更新Q表 14. segment = segment + 1； 15. currentState = nextState； 16. UNTIL segment = 800; 17. round = round + 1; 18. UNTIL round = 100; 19. END |

得到策略矩阵Q(S,A)后，Agent将进入“部署测试阶段”。该阶段主要是根据当前Agent所处的状态，查询策略矩阵得到具体的策略，即“Agent处于当前状态时Agent应请求下载何种码率的视频片段”。KNN-Q学习算法的“部署测试阶段”的算法如算法2所示：

|  |
| --- |
| 输入：策略矩阵Q表  输出：请求下载每个视频片段时的码率和SSIM值 |
| 1. BEGIN 2. 初始化*segment* = 0,*round* = 0; 3. 初始化网络带宽，缓存; 4. REPEAT 5. 根据网络带宽、缓存、前一视频片段的SSIM值计 6. 算当前时刻agent的状态*currentState*; 7. REPEAT 8. IF 状态*currentState*在策略矩阵Q表中存在 THEN 9. 根据状态state查询Q表，得到Q值数组； 10. END-IF 11. ELSE 12. 结合KNN算法查询Q表，得到Q值数组； 13. END-ELSE 14. 在Q值数组中找最大Q值对应的下标*actionIndex*； 15. 请求下载*actionIndex*相应码率的视频片段； 16. 更新缓存； 17. 计算SSIM值； 18. *segment* = *segment* + 1 19. 计算agent下一个状态*nextState*； 20. *currentState* = *nextState*； 21. UNTIL *segment* = 800 22. END |

# 2.3 视频质量评价指标

本文采用广泛应用于评估视频质量[9-11]的结构相似性指数 （Structural Similarity Index, SSIM），作为本次实验中视频播放质量的评价指标。SSIM是衡量两张图片相似度的指标，其中一张图是没有经过压缩的无失真的原始图像，另外一张图是原始图像经过压缩后的有失真的图像。SSIM与传统度量指标PSNR（Peak Signal to Noise Ratio）和MSE（mean square error）相比，改善了感知到的视频质量的表现形式。PSNR和MSE等评估指标是基于逐个像素的分析比较来估计感知到的失真。与此不同的是，SSIM利用结构信息来评估图像质量，因此SSIM考虑了位置上互相紧靠的元素之间的互依赖性。

SSIM计算公式如（6）所示，计算方法为：给定图像X和Y，利用两个图像中的像素点亮度值的均值、方差和协方差来计算SSIM值。其中，和分别是图像X和Y像素点亮度值的均值，和分别是图像X和Y像素点亮度值的方差，是图像X和Y像素点亮度值之间的协方差，和是为了避免分母为0的而设置的常数。SSIM的取值范围是-1到1，当SSIM值越接近1时，代表两个图像越相似，即图像失真程度越低；反之，则代表图像失真程度越高。

（6）

# 3.仿真实验

## 3.1 实验说明

本节通过在Matlab进行的一系列仿真实验来评估本文提出的KNN-Q学习算法的性能，即编写Matlab程序模拟DASH客户端向HTTP服务器请求下载视频并播放视频的过程。本文通过设置DASH客户端处于不同的网络带宽条件、DASH客户端请求拥有不同码率的视频等操作来模拟不同的实验所需的场景。本文提出“简单场景”、“常规场景”和“复杂场景”等三种播放场景，用以对比KNN-Q学习算法和Q学习算法的性能。最后在常规场景下，分别对KNN-Q学习算法公式中涉及到的距离公式和K值进行灵敏度分析的对比仿真实验。

为方便计算，SSIM值可近似为以图片或视频的相对码率的对数值为自变量的4维多项式。不同码率的视频片段的SSIM可以利用下面的公式（7）、（8）近似计算得到[13]：

(7)

+ (8)

其中，是视频的最高清的码率，即视频的原始码率，是经过视频服务器上的“编码模块”压缩后的视频编码率。(1, ,,,)是代表视频片段的复杂度系数。

仿真实验中用以组成测试视频的视频素材源于EvalVid CIF视频数据（https://media.xiph.org/video/derf/）。

视频素材库中各个视频素材的与SSIM值的关系如（2）所示[?]：

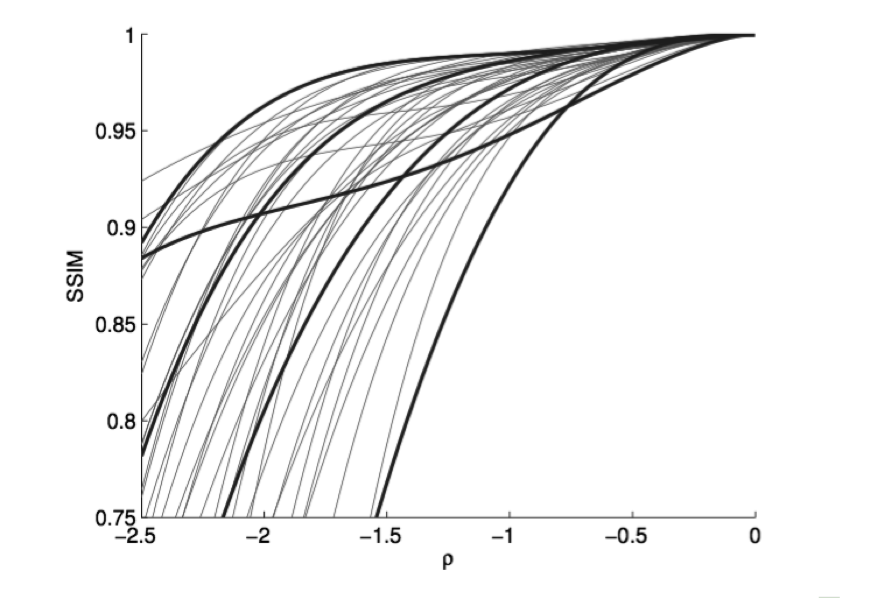


图2视频素材与SSIM的关系

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（2）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2所示[?]：

表 2 5个视频素材的复杂度系数

|  |  |
| --- | --- |
| 视频 | 素材复杂度系数  ,，， |
| 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 |

实验中涉及的相关参数如表3所示：

表3 实验中的相关参数

|  |  |
| --- | --- |
| 参数名 | 值 |
|  | 0.05 |
|  | 0.09 |
|  | 0.02 |
|  | 800 |
|  | 800 |
| K | 3 |
|  | 60s |

分别是公式学习(5)中出现的学习速率和折扣系数， 是公式（2）中的温度系数。， 分别是Q学习算法和KNN-Q学习算法一轮迭代训练中的视频片段的总个数。 是缓存的上限值。

## 3.2 实验结果

为保证策略矩阵Q表能够达到收敛，每次实验的“训练阶段”有100 轮迭代训练，每轮迭代共经历800个视频片段的训练。每个播放场景都会进行10组重复实验,最终得出的可视化实验结果都是基于每次10组实验的缓存、SSIM的均值及标准差。

“简单场景”下，“训练阶段”和“测试阶段”网络带宽都比较稳定，网络带宽取值范围是[5500，6000]，单位 kb/s,且只用1个视频素材News生成 “训练阶段”和“部署测试阶段”的测试视频，该测试视频复杂度比较单一。

KNN-Q学习与Q学习在“简单场景”下的DASH客户端的缓存和SSIM对比情况如图3所示；由图3可知，采用KNN-Q学习算法的DASH客户端的缓存在播放大约第50个视频片段之后达到最大值（约53s），而Q学习算法采用Q学习算法的DASH客户端的缓存在播放大约第70个视频片段之后达到最大值（约60s）。



“常规场景”的“训练阶段”和“部署测试阶段”网络带宽大小都比较稳定，与“简单场景”一样，网络带宽取值范围是[5500，6000]，单位 kb/s。但“常规场景”使用表2中的5个视频素材，随机组合生成“测试阶段”和“测试阶段”的测试视频。

“复杂场景”下，“训练阶段”和“部署测试阶段”的网络带宽大小变化比较大，可取值为[400 750 1500 2500 3500 4500 5750 7250 9000 12500],单位 kb/s。同“常规场景”，“复杂场景”下也是用表1 中的5个视频素材，随机生成的“训练阶段”和“部署测试阶段”的测试视频。



图3简单场景下缓存对比图

KNN-Q学习与Q学习在“简单场景”下的DASH客户端的SSIM值比较情况如图4、图5所示，KNN-Q学习算法的SSIM值比 Q学习算法的SSIM值显著高（ p = 0.0090）。

图4简单场景下SSIM值对比图



图5简单场景下SSIM值对比图

KNN-Q学习与Q学习在“常规场景”下的DASH客户端的缓存对比情况如图所示，由图可知，KNN-Q学习和Q学习的缓存值，都大概在50个视频片段之后近似达到缓存上限60秒。SSIM值对比情况如图7、图8所示，由图7、8可知，KNN-Q学习算法的SSIM值比Q学习算法显著高（p= 1.0711\*10^-07）。



图6 常规场景下缓存对比图

图7常规场景下SSIM值对比图



图8常规场景下SSIM值对比图

KNN-Q学习与Q学习在“复杂场景”下的DASH客户端的缓存对比情况如图9所示，由图可知，Q学习缓存值在100个视频片段之后近似达到缓存上限60s，而KNN-Q学习在100个视频片段之后的缓存值一直在40s上下波动。SSIM值对比情况如图10、图11所示，由图10、11可知，KNN-Q学习算法的SSIM值比Q学习算法显著高（p= 1.0711\*10^-07）。



图9复杂场景下缓存值对比图

图10复杂场景下SSIM值对比图



图11复杂场景下SSIM值对比图

## 6.4距离公式灵敏度分析

6.4.1实验设置

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。

## 6.4.2 实验结果

缓存对比结果如图12所示, 可见在“常规场景”中，设置K=2，采用欧式距离、曼哈顿距离和切比雪夫距离三种距离公式的缓存情况非常接近，难分伯仲。但是由图13可知，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。



图12距离公式灵敏度分析-缓存



图13距离公式灵敏度分析-SSIM

## 6.5 K值灵敏度分析

6.5.1实验设置

在“常规场景”中，设置距离公式为欧式距离，分别取K = 2，3，4来判断KNN-Q中K值的大小对于实验效果的影响。

## 6.5.2 实验结果

缓存对比结果如图14所示, 可见在“常规场景”中，设置默认距离公式为欧式距离，设置K=2时，缓存值对中停留在50秒左右，而K=3,4时，缓存值停留在60秒左右。然而K=2时，其SSIM值拥有最大值，即视频播放质量最好。



图14 K值灵敏度分析-缓存



图15 K值灵敏度分析-SSIM

# 结束语

Matlab仿真实验结果显示，在三种场景对比实验中，采用KNN-Q学习算法作为视频片段获取策略，都比采用Q学习算法作为视频片段的获取策略，DASH客户端播放的视频的QoE指标SSIM值将更高，表明采用KNN-Q学习作为视频片段获取策略将会有更好的视频播放质量。在参数灵敏度分析中，采用欧式距离和K=2，会有更高的SSIM值。

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。

参考文献：

1. Cisco, “Global Mobile Data Traffic Forecast Update, 2016–2021 White Paper,” Cisco Public Information, 2017.
2. Suzen Saju Kallungal,"A Survey on Adaptive Video Streaming Technologies",International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 6, Issue 3, March 2017, ISSN: 2278 – 1323
3. T. Stockhammer, “Dynamic Adaptive Streaming Over HTTP: Standards and Design Principles,” in Proceedings of the second annual ACM conference on Multimedia systems (MMSys), San Jose, USA, 23-25 February 2011.
4. Jonathan Kua,Grenville Armitage,Philip Branch,"A Survey of Rate Adaptation Techniques for Dynamic Adaptive Streaming Over HTTP",IEEE COMMUNICATIONS SURVEYS & TUTORIALS, VOL. 19, NO. 3, THIRD QUARTER 2017
5. C. Liu, I. Bouazizi, M. Gabbouj, “Rate adaptation for adaptive HTTP streaming,” in Proc. of ACM MMSys2011, California, Feb. 2011.
6. L. De Cicco, S. Mascolo, “An experimental investigation of the Akamai adaptive video streaming,” in Proc. of USAB2010, pp. 447-464, Klagenfurt, Austria Nov. 2010.
7. S. Akhshabi, A.C. Begen, C. Dovrolis, “An experimental evaluation of rate-adaptation algorithms in adaptive streaming over HTTP,” in Proc. ACM MMSys2011, California, Feb. 2011.
8. K. Evensen, et al. “Improving the performance of quality-adaptive video streaming over multiple heterogeneous access networks,” in Proc. ACM MMSys2011, California, Feb. 2011.
9. J. F. T. W. W. V. L. Maxim Claeys, Steven Latre ́ and F. D. Turck, “Design of a Q-Learning-based client quality selection algorithm for HTTP adaptive video streaming,” in Adaptive and Learning Agents Workshop, part of AAMAS2013 (ALA-2013), 2013, pp. 30–37.
10. M. Claeys, S. Latre ́, J. Famaey, T. Wu, W. Van Leekwijck, and F. De Turck, “Design and optimisation of a (FA) Q-learning-based HTTP adaptive streaming client,” Connection Science, vol. 26, pp. 25–43, 2014
11. G. V. Wallendael, S. V. Leuven, J. D. Cock, P. Lambert, R. V. D. Walle, N. Staelens, and P. Demeester, “Evaluation of full-reference objective video quality metrics on high efficiency video coding,” in IFIP/IEEE In- ternational Symposium on Integrated Network Management (IM 2013). IEEE, 2013, pp. 1294–1299.
12. T. A. Le, H. Nguyen, and H. Zhang, “Scalable video transmission on overlay networks,” in IEEE Second International Conferences on Advances in Multimedia (MMEDIA). IEEE, 2010, pp. 180–184.
13. S. Pasqualini, F. Fioretti, A. Andreoli, and P. Pierleoni, “Comparison of h.264/avc, h.264 with aif, and avs based on different video quality metrics,” in IEEE International Conference on Telecommunications. IEEE, 2009, pp. 190–195.
14. Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” IEEE Trans. Image Process., vol. 13, no. 4, pp. 600–612, Apr. 2004.
15. M. Zanforlin, D. Munaretto, A. Zanella, and M. Zorzi, “SSIM-based video admis- sion control and resource allocation algorithms,” in 12th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), pp. 656–661, IEEE, 2014.