硕士学位论文

视频用户 QoE 的感知和预测

Video User QoE Perception and Prediction

作者：张加林

导师：郑宏云

北京交通大学

2019 年 6 月

学位论文版权使用授权书

本学位论文作者完全了解北京交通大学有关保留、使用学位论文的规定。特授权北京交通大学可以将学位论文的全部或部分内容编入有关数据库进行检索，提供阅览服务，并采用影印、缩印或扫描等复制手段保存、汇编以供查阅和借阅。同意学校向国家有关部门或机构送交论文的复印件和磁盘。学校可以为存在馆际合作关系的兄弟高校用户提供文献传递服务和交换服务。

（保密的学位论文在解密后适用本授权说明）

学位论文作者签名：张加林　　导师签名：郑宏云

签字日期：2019 年 6 月 3 日　　签字日期：2019 年 6 月 3 日

学校代码：10004 密级：公开

北京交通大学

硕士学位论文

视频用户 QoE 的感知和预测

Video User QoE Perception and Prediction

作者姓名：张加林 学　　号：16120164

导师姓名：郑宏云 职　　称：副教授

学位类别：工学 学位级别：硕士

学科专业：通信与信息系统 研究方向：信息网络

北京交通大学

2019 年 6 月

致谢

“春光荏苒休孤负，相对原宜惜寸阴”，三年的研究生生涯临近尾声，两个月“漫长”的毕业论文书写工作，也终于到了写这个“致谢”的时候。这三年，在交大、在有线组，结识了很多良师益友，感恩与他们在这里相识相知。在毕业论文完成之际，我想要在这个部分，向所有帮助过和支持过我的人表示衷心的感谢。

首先，感谢我的导师郑宏云老师。郑老师严谨的治学态度和科学的工作方法给了我极大的帮助和影响。这三年，由于我薄弱的个人能力，让郑老师时常为我能否顺利毕业的问题而担忧，对此，我感到非常抱歉和愧疚。“雷霆雨露，俱是师恩”，感谢郑老师在这三年来，对我的鼓励、批评、指导、训斥、赞赏和期盼。

然后，感谢赵永祥老师、李纯喜老师、郭宇春老师和陈一帅老师。各位老师一直在坚持的“每周论文研讨会”制度，给予了我，就某一个问题，与教授们直接面对面思辩的机会，这不仅锻炼了我的口头表述能力，也开拓了我的学术视野，重新培养了我思考问题、提出问题的能力。陈一帅老师丰富多彩的兴趣爱好、热情洋溢的个人魅力、温暖细腻的家国情怀，在许多方面给予我启发，告诉我生活的多样性，让我更加珍视生活，用心地去感受生活。同时，感谢陈老师领着我接触了人工智能，这个未来无限可能的领域。

此外，感谢加拿大滑铁卢大学 zhengfang duanmu 博士、东南大学计算机科学与工程学院潘吴斌博士，在我毕设研究期间，对我的答疑和帮助，感谢你们不介意我一次次的在邮件和微信上叨扰。

另外，好像规定还要感谢那些“协助完成研究工作和提供便利条件的组织和个人”，所以，在这里，我“不得不”还要感谢赵红娜的电脑、张琛玥的零食、宋云鹏的“叨扰”。感谢黄楠栖、王璐、陈滨等与我一同学习、生活两年多的实验室同学，与你们一起学习、科研、八卦、哈啤酒、找工作的经历，是我一生中最宝贵的财富。本论文的顺利完成，也离不开你们的关心和帮助。

最后，非常感谢我的家人，我的爸爸、妈妈、姐姐，是你们的辛勤工作、无私付出，让我得以在良好的环境中学习并顺利毕业。感恩你们一直以来的理解和支持！

最后的最后，还要感谢一下自己，感谢自己一直以来的坚持，感谢自己的努力和付出！感谢那个虽然一路跌跌撞撞，但是却始终不曾放弃、砥砺前行的自己！

此外，感谢国家自然科学基金（No.61872031 基于熵理论的信息匹配网络测量与建模）对我研究工作的资助。

摘要

视频流媒体已经成为目前因特网的主要应用，为了给视频用户提供更好的网络服务，运营商有必要了解并保障用户的观看体验。用户的观看体验，即体验质量QoE(Quality of Experience)，是用户对视频服务质量的主观感受。如果没有终端用户的配合，运营商无法得知用户的主观 QoE，即便是应用层的客观 QoE 参数（视频帧率、码率、初始时延、卡顿等），运营商也难以获得。因此，运营商通常采用监测网络流量来推测视频用户的 QoE。

然而，目前这种解决方法也因 HTTP 自适应流媒体传输技术(HAS)的使用和加密视频流的逐渐普及，而面临新的挑战。HAS 机制中码率自适应请求算法（ABR）能够自适应网络变化，而选取匹配网络状况的视频块进行传输，从而会模糊掉网络流量波动所带来的用户 QoE 信息。这样会使得运营商无法判断用户 QoE 得以改善的原因是 ABR 自适应的结果还是网络状况改善带来的结果。与此同时，目前越来越多的 HAS 视频流开始采用加密传输服务，使得运营商无法继续采用深度解析包的方法进行网络流量分析。

本文工作注意到视频客户端 ABR 算法所造成的流量特征模糊的问题，同时也注意到 ABR 算法是以视频块为基本单位进行自适应码率调整。因此，与传统方法以“包”为颗粒度不同，本论文提出以视频“块”为颗粒度进行视频流量分析和挖掘，进而评估用户 QoE。具体地，论文包括以下两个部分工作。

第一部分工作，研究了如何从加密的 HAS 视频流量中重建视频块序列。本文首先基于网络测量的方法对 HAS 加密视频流的传输特性进行分析，然后基于视频流量传输模式的分析结果，提出一种 HAS 视频流的块序列重建算法。实验结果表明，采用我们算法从网络流量中重建的视频块序列，均方根误差不高于 0.132，具有很好的拟合精确度。

由于播放器的演播状态直接影响到用户 QoE，播放器的演播状态又可以采用应用层客观的 QoE 参数来表征，因此第二部分工作基于视频块序列的特征来推断播放器的演播状态。具体地，论文工作首先为此提出一种新的应用层客观 QoE 度量指标：缓冲区综合状态，用来表征播放器的演播状态。接着，利用 HAS 客户端与服务器之间的双向视频流量，重建视频块序列，使用时间序列数据挖掘技术，建立了网络流量特征与新的应用层客观 QoE 指标之间的预测模型，用于实时估测视频的演播状态，从而实现视频用户 QoE 的跨层感知和预测。实验结果表明，采用我们的方法能够获得较好的实时预测效果。

关键词：QoE；HAS 视频流；ABR；运营商；加密；时序数据挖掘；块序列

ABSTRACT

Video streaming has become the main application of the Internet. In order to provide better network services for video users, operators need to understand and guarantee the user's viewing experience. The user's viewing experience, QoE (Quality of Experience), is the user's subjective feeling of video service quality. If there is no cooperation with the end users, the operator cannot directly aware the user's subjective QoE. Even for objective QoE parameters of the application layer, such as video frame rate, bitrate, initial delay and stall, are difficult for operators to obtain. Therefore, operators often monitor network traffic to estimate the QoE of video users.

However, this solution is currently facing new challenges due to the use of HTTP Adaptive Streaming (HAS) and the increasing popularity of encrypted video streams. In the HAS mechanism, the Adaptive BitRate algorithms (ABR) can adapt to network changes and select the video chunks that match the network states for transmission, which will obscure the user QoE information brought by network traffic fluctuations. This will make it impossible for the operators to judge whether the user's QoE is improved because the result of ABR adaptation or the improvement of network conditions. In addition, more and more HAS video streamings are beginning to adopt encrypted transmission services, which makes operators unable to continuly use deep parsing methods for network traffic analysis.

This paper focuses on the problem of fuzzy traffic characteristics caused by the ABR algorithm on video client. At the same time, it is also noticed that the ABR algorithm uses the video chunk as the basic unit for adaptive rate adjustment. Therefore, different from the traditional methods use 'package' as the granularity, this paper proposes video traffic analyzing and mining at the granularity of video 'chunk', and then evaluates the user QoE. Specifically, the main work of this paper includes the following two parts:

The first part is how to reconstruct a video chunk series from encrypted HAS video traffic. Firstly, based on the network measurement method, the transmission characteristics of the HAS encrypted video streaming are analyzed. Then, based on the analysis result of the video traffic transmission mode, a chunk series reconstruction algorithm for the HAS video streaming is proposed. The experimental results show that the root mean square error of the video chunk series reconstructed from the network traffic by our algorithm is not higher than 0.132, which has good fitting accuracy.

Since the player's states directly affect the user QoE and it can be characterized by

the application layer's objective QoE parameters, the second part of the work is to infer the player's states based on the characteristics of the video chunk series. Specifically, the thesis first proposes a new application layer objective QoE metric, buffer comprehensive state, to characterize the player's states. Then, using the two-way video traffic between the HAS client and the server, the video chunk series is reconstructed, and the time series data mining technology is used to establish a prediction model between the network traffic characteristics and the new application layer objective QoE parameter for real-time estimation. The video playback state is predicted and measured in real time according to the established model, thereby achieving cross-layer preception and prediction of the video user's QoE. The experimental results show that our method can obtain better real-time prediction results.

KEYWORDS： QoE; HAS video streaming; ABR; operator; encryption; time series data mining; chunk series

目录

缩略词表

英文缩写	英文全称	中文全称
QoE	Quality of Experience	体验质量
QoS	Quality of Service	服务质量
HAS	HTTP Adaptive Streaming	HTTP 自适应流媒体
ABR	Adaptive Bitrate Algorithms	自适应码率
TSDM	Time Series Data Mining	时间序列数据挖掘
ANN	Artificial Neural Network	人工神经网络
LSTM	Long Short-Term Memory	长短期记忆网络
RNN	Recurrent Neural Network	循环神经网络
RMSE	Root Mean Squared Error	均方根误差
MPC	Model Predictive Control	模型预测控制
BOLA	Buffer Occupancy based Lyapunov Algorithm	基于缓冲区占用的李雅普诺夫算法
ARIMA	Auto Regressive Integrated Moving Average model	差分整合移动自回归模型
DASH	Dynamic Adaptive Streaming over HTTP	基于 HTTP 的动态自适应流媒体

1 引言

1.1 研究背景和意义

近年来，全球网络视频业务的发展势头迅猛，视频流媒体已经成为目前因特网的主要应用。数据表明，2018 年网络视频播放量年同比增长 52%，总观影时长年同比增长 63%，网络视频服务作为主导业务占据了因特网高达 58%的下行流量[1]。欣欣向荣的网络视频产业，给视频服务提供商们带来的不仅仅是机遇，还有责任。

为了给用户提供更好的网络服务，运营商（网络服务提供商）有必要更充分地了解并保障视频流的服务质量和视频用户的观看体验。用户的观看体验，即体验质量 QoE（Quality of Experience），是指用户对某项应用或服务整体的主观可接受程度，是从用户角度对业务的 QoS（Quality of Service，服务质量）机制进行评价的度量标准。传统的 QoS 机制关注业务架构与服务流程中技术因素的影响，直接将业务系统运行时网络参数的变化情况（例如网络延迟和阻塞）表征为该应用之于用户的服务质量，其评价主体是网络。与之不同，QoE 的评价主体是终端用户，它可以从用户的主观感受中反映视频流的服务质量。因此，在这个“用户至上、需求为王”的互联网时代，运营商可以通过评估 QoE，感知用户的视频观看体验，从而更合理地指导网络资源调度，保障流媒体的传输质量。

但是，实际中，运营商在感知用户 QoE 时，会在技术上面临很大的困难。首先，如前所述，QoE 是用户的主观感受，运营商能够观察的是网络侧的数据，如果没有终端用户的配合，运营商无法得知 QoE。虽然可以采用客观的 QoE 参数，例如视频播放的码率、帧率、初始时延、卡顿等来评估用户的 QoE，但这些参数依然位于客户端（即属于应用层参数），运营商仍然难以获取[2]。对运营商而言，通过监测网络流量来跨层推测视频用户的 QoE，不失为一种解决方案[3]。

但是，这种解决方法因视频流传输技术的发展和变化面临新的挑战。目前的视频流传输越来越多地采用 HTTP 自适应流媒体传输技术（HTTP Adaptive Streaming，HAS）。HAS 机制中码率自适应请求算法（Adaptive BitRate algorithms，ABR）的应用，会在一定程度上模糊掉网络中视频流量波动所带来的用户 QoE 信息[3]，甚至会带来具有“欺骗性”的网络流量模式，例如，在实际网络状况并不理想的情况下，客户端通过强大的 ABR 算法调控视频的播放质量，保证了视频的平稳演播和良好的用户 QoE，但是，此时运营商就有可能会误以为，用户良好的 QoE 是网络调节导致的，从而造成错误的判断和优化方向。并且，不同的 HAS 客户端会采用

不同的 ABR 算法，由此产生的实际流量模式也不尽相同[4]。所以说，HAS 机制中 ABR 算法的应用，对于客户端来说，是提升了其保障用户 QoE 的能力，但是对于运营商来说，却是被“欺骗”了！

此外，为了更好地保护用户隐私，越来越多的视频内容提供商开始采用 HTTPS 等加密协议对视频流进行加密传输，使得通过诸如深度包解析[5,6]、网络探针[7,8] 等方法进行流量信息提取越来越困难。在这种情况下，运营商若想更有效地根据网络流量推测应用层的客观 QoE 参数，进而实现用户 QoE 的评估，就需要一种更好地视角和方法。

遗憾地是，就我们目前掌握的资料来看，虽然现有的部分研究已经开始着眼于从运营商的角度基于加密视频流评估用户 QoE，但是却都没有注意到，在从网络流量中挖掘用户 QoE 时，客户端 ABR 算法所带来的流量特征动态模糊的问题。运营商若想要更好的感知用户 QoE，并有效的指导网络资源调度，首先需要能够从网络流量中挖掘出客户端的 ABR 调控信息，规避因为“视野狭窄和蒙蔽”所带来的 QoE 调节效果误判的问题。我们注意到，HAS 视频传输机制中，服务器端对视频内容按照“块”的粒度进行切分和存储，客户端依据 ABR 算法按照“块”的粒度对视频内容进行请求，但是，当视频块数据在网络中进行传输的时候，由于网络传输单元的容量限制，视频块会被分解成一个个 TCP 数据包进行传输，这种处理方式会在一定程度上模糊甚至破坏掉网络流量序列中本来包含的客户端 ABR 调控信息（例如视频块请求时刻、码率等级、码率切换水平等），影响运营商基于网络流量评估用户 QoE。因此，本文的第一部分工作，研究了如何从加密的 HAS 视频流量中重建视频块序列。我们首先基于网络测量的方法对 HAS 加密视频流的传输特性进行分析，然后基于视频流量传输模式的分析结果，提出一种 HAS 视频流的块序列重建算法。

此外，我们还注意到，在采用应用层客观的 QoE 参数来表征用户主观 QoE 的相关工作中，见参考文献[9-31]，部分研究虽然也意识到了用户在不同的播放状态下（例如卡顿、初始延迟）会有不同的视频观看体验，选择通过评估播放器状态间接地感知用户 QoE。但是，他们却都简单地将缓冲区占有量等价为播放器状态，没有考虑到客户端缓冲区的动态变化对播放器演播状态的影响。我们的研究工作发现，在视频演播过程中，缓冲区占有量会经历上升-下降的周期性振荡，由于缓冲区占有量升降变化的“惯性”，即便是具有相同的占有量，当缓冲区处于不同的变化过程中，即处于上升或是下降过程，播放器的演播状态是不同的，带来的视频观看体验也是不同的。基于此观察，本文的第二部分工作综合缓冲区占有量和它的变化趋势，提出一种新的应用层客观 QoE 度量指标：缓冲区综合状态，来表征播放器的演播状态。同时，我们利用 HAS 客户端与服务器之间的双向流量数据，基于重

建的视频块序列，使用时间序列数据挖掘技术，建立了网络流量特征与新的应用层客观 QoE 指标之间的预测模型，用于实时估测视频的演播状态，从而实现视频用户 QoE 的跨层感知和预测。与已有的方法不同，我们的方法只用到了视频流量的统计信息，在实际部署时不需要网络封包的明文数据和视频演播参数，因此，该方法在应用时既不需要客户端/服务器的参与，也可以工作于流量加密的情况下，具有客观、易用的特点。此外，我们的方法中使用了适用于时间序列的模型和训练技巧，充分利用了 HAS 视频流的时序特征，处理方式更贴近该数据类型的特点。

1.2 研究现状

本节主要从视频流量的特征分析和视频用户的 QoE 评估两个部分，详细介绍本文中相关工作的研究现状。

1.2.1 视频流量的特征分析

本小节主要概述视频流量特征分析的研究现状。

文献[32]中描述了 HAS 播放器在 Steady-State 状态时网络流量的典型特征。他们发现视频流在正常演播时会呈现出周期性的 ON-OFF 特征，在 ON 状态下，网络中维持有下载流量，此时的播放器缓冲区占有量会持续增长至最大阈值；然后进入 OFF 状态，此时客户端会中断数据流的请求和下载，直至缓存长度逐渐降低到最小阈值，播放器重新进入 ON 状态，如此往复。

文献[33]中从视频流量如何在服务器中生成的角度描述了 YouTube 的视频服务特征。他们发现 YouTube 会在每次视频服务的最开始，以网络最大可用传输带宽传送 40s 左右的视频内容，此时的服务器进入“初始突发阶段”；然后服务器会将数据发送速率限制为视频块码率的 1.25 倍，用于节制视频资源的传输速度，此时的服务器进入“节流阶段”，该阶段下视频流的生成速率在 250kbps-1.25Mbps 之间。

文献[34]中提出了一种非侵入式的流量分析解决方案，用于观察基站、边缘路由器或网关中的未加密 HAS 视频流。通过视频演播时网络侧 TCP 下载吞吐量与客户端码率之间差值曲线的斜率特征，将 HAS 服务的工作状态分成三个阶段：充盈、稳态、衰减，并在单一码率的情况下验证了这种差值曲线斜率特征的有效性。

文献[35]中关注视频流服务中的 TCP 控制策略，通过分析 YouTube 和 Netflix 的视频流量特征，发现流媒体的 TCP 传输控制策略因应用程序（例如 Web 浏览器、移动端应用程序）和容器（例如 Sliverlight、Flash、HTML5）类型的不同存在不同的 ON-OFF 特征。

文献[36]中分别在简单和复杂的网络环境下，对不同 YouTube 视频的流量特征进行了分析。通过对比播放器的实时演播日志发现，在复杂的网络环境下，YouTube 会对同一种内容的视频块采用多种码率质量版本并行传输，以适应网络情况的变化，这种行为会产生高达 33%的网络冗余流量。

文献[37]对校园网络环境下的 YouTube 视频的传输流量进行分析，挖掘其使用模式、文件属性、流行度和传输行为的时间特征。他们发现校园网络下的 YouTube 访问模式与用户行为活动密切相关，视频流量模式随着日历、周历和学术大事记等重要日程呈现出显著不同的变化。

文献[38]对视频点播和视频共享服务从会话请求到达率、会话请求时间间隔、下载流量峰值、视频流行度分布、视频持续时间分布等流量特征的角度进行分析。他们发现这些服务的会话请求到达率并不符合 Poisson 过程，其分布符合对数正态分布特性，会话请求时间间隔符合广延指数分布特性。

文献[39]中应用 MinMax K-means、OPTICS 和 AutoClass 等无监督聚类算法对 DASH 视频流量基于统计依赖性进行相似度分析，以区别不同视频播放策略下的 DASH 流量类型。

文献[40]和[41]中对不同播放策略下的 HAS 视频流量结构进行分析。他们发现 HAS 视频的流量与 HTTP 渐进式下载方式不同，呈现负相关、反持久性的特点，并且，采用相同或是类似播放策略的客户端请求的视频流量之间呈现正相关、同步的关系；而源自不同播放策略的 HAS 流量之间呈现负相关或是不相关[41]。

文献[42]中使用 GoP 分类、相位拟合、马尔可夫链等技术进行网络视频流量建模。该模型可以用高效地流体马尔可夫过程对任何类型、长度的视频数据在不同编码器下产生的视频流量进行人工建模，捕捉流量特征，用以指导运营商根据人造流量模型，对多种传输模式下的视频流量进行整形和控制的相关研究。

文献[43]中基于服务器对于 HAS 视频不同的分块切分逻辑，对无线网络视频的流量特性进行分析。他们发现大的视频块持续时间在略微提高视频分辨率的同时，会显著降低视频播放的平滑度，这与将视频块编码为高码率进行传输产生的影响相同。

文献[44]中分别对 Apple IOS 和 Andriod 终端中 YouTube 视频服务的移动网络流量特征进行了对比分析。他们发现 YouTube 服务器在每一次视频服务的开始都存在初始突发阶段和限制传输阶段，这两个阶段下，YouTube 会给予不同类型的客户端以不同的传输参数，导致不同的数据传输速度和体验。高端 Andriod 设备的 YouTube 服务采用双阈值缓冲策略，客户端可以根据缓冲区占用情况主动中断和恢复数据下载；IOS 设备中，视频分块下载不会被 YouTube 客户端中断，因此可能会在播放器缓冲区中累积大量数据；中端 Andriod 设备中，在播放器缓冲区充盈时，

TCP 接收窗口也会限制视频流的继续传输。

1.2.2 视频用户 QoE 的评估

为了提供更好的流媒体业务，视频服务提供商有必要对自己的服务质量进行追踪和评估，而 QoE 作为一种度量指标，可以直接地反映用户的主观感受。因此，视频服务提供商可以通过感知用户的 QoE，更合理高效地优化视频播控策略和网络资源调度。

一直以来，HAS 视频用户 QoE 评估的工作璀璨叠出，提出了许多具有代表性的研究方法。基于研究角度和数据收集位置的不同，我们把现有的相关研究分为两类：从客户端的角度评估用户 QoE，从网络侧（运营商）的角度感知用户 QoE。第一类研究根据 QoE 评估方法的不同，又可以分为两种：主观 QoE 测量和客观 QoE 测量。主观 QoE 测量是模拟用户的真实观看环境，直接通过受测人员对视频业务的主观打分，研究光线、界面、视频流行度等主观因素对用户 QoE 的影响[45]，这种方法可以准确的反映用户的主观感受，但是实施难度大、可移植性差，并且容易受到样本自身差异性的影响，说服力较弱；客观 QoE 测量是基于客户端的视频演播参数，构建客观 QoE 度量指标，如初始延迟、卡顿率、码率切换、缓冲区占用率、进度条拖拽行为等[45]，建立客观 QoE 指标与主观 QoE 分数之间的数学函数模型，侧面推测用户 QoE。

与第一类研究不同，第二类研究中由于运营商所处位置的局限性，无法直接获取到与用户 QoE 强相关的播放器侧度量指标，退而求其次，运营商只得通过挖掘网络流量特征获取视频播放的相关信息，分析网络 QoS 参数，跨层感知视频用户 QoE。根据对网络流量分析手段的不同，这类研究也可以分成两种：侵入式 QoS 测量法和非侵入式 QoS 测量法。侵入式 QoS 测量法是基于深度包检测[5-6]或网络探针[7-8]等方法，直接获取视频流量关键报文中记录的视频文件相关信息如码率、帧率、分辨率等，这类方法直截了当，但是却需要复杂的数据处理，无法流程化实时在线分析，并且有窃听用户隐私的风险[9]。非侵入式 QoS 测量法是通过对网络流量进行统计分析，在不破坏流量结构的前提下，获取诸如吞吐量、重传率、延迟等 QoS 参数[45]，建立网络 QoS 与用户 QoE 之间的跨层映射关系。 本文的研究角度属于第二类：从网络侧跨层感知视频用户 QoE。近几年，这个方向的研究工作越来越多的引入机器学习、深度学习、数据挖掘等人工智能的方法建模 QoS-QoE 映射关系，见参考文献[9-31]。

文献[14]中提出了一种在加密视频流下，使用机器学习技术估计视频块码率的算法。该算法中指出拆分自同一个视频块的网络报文的 ACK 号相同，并基于该发

现进行了视频流量聚合。但是该算法在计算码率时使用的视频块持续时间参数来源于他们对 YouTube 视频文件的一般统计结果，不完全适用于其他 HAS 视频。

文献[30]利用增强学习网络，将视频播放过程中网络 QoS 参数（延迟、下载吞吐量等）和播放器演播信息（缓冲区占用率、播放速率等）作为神经网络模型的输入信息，将 QoE 指标作为模型的输出信息和反馈信息，用以优化客户端的码率自适应算法。

文献[26]将视频用户 QoE 感知问题看作成时间序列预测问题，研究客户端的不良演播状态（例如卡顿、码率切换频繁等）对用户观看感受的长时间影响。作者们训练了一系列循环神经网络和非线性自回归模型预测用户 QoE 打分，并引入了记忆残留函数，代表用户在经历卡顿或码率切换等不良播放事件后，这种糟糕体验随着时间推移的消退程度。这种方法认识到并使用了视频演播参数在时序上的应用价值，综合近期客户端播放事件和用户主观感受，预测当前状态下用户的主观 QoE。但是，该文献并未讨论如何识别卡顿、码率切换等不良的播放状态或事件。与它不同的是，本论文研究的是如何识别播放状态。

本文在前期工作[10]中提出了一种视频流卡顿识别算法。该算法利用视频服务器和客户端之间的双向流量数据，使用决策树的方法，建立分类模型，根据网络流量的统计特征实时预测视频播放时是否发生卡顿。这篇文章中使用了特征提取技术将视频流量序列的时序信息建模在每一个分类样本中，有效地在同一个样本中引入了更多的流量传输特征。但是，在该前期工作中，我们依然使用了缓冲区的实时占用量来表征播放器的演播状态（是否发生卡顿）。此外，文章中使用的机器学习分类模型，把具有时序特征的 QoS 参数序列分割成一个个时刻样本点单独参与训练，破坏了视频流 QoS 参数序列在时间上的结构，忽略了其中所原本包含的时序变化信息。

1.3 论文工作和组织结构

具体地，本论文完成了三个方面的工作：

(1) 提出新的客观 QoE 指标。不再直接使用“缓冲区占有量”，而是综合缓冲区实时占有量和缓冲区占有量的变化趋势，提出一个全新的客观 QoE 度量指标：缓冲区综合状态，来表征播放器状态。

(2) 对颗粒度为“报文”的网络流量进行重建，构造出颗粒度为“块”的视频网络流。重建视频块序列可以在数据预处理的层面上，重构出网络视频流量中那些“隐藏”了的客户端 ABR 调控信息。

(3) 使用时间序列数据挖掘技术，建立视频网络流量特征与新的客观 QoE 度量

指标之间的预测模型，用于实时感知客户端播放器的演播状态。我们的方法中使用了适用于时间序列的模型和训练技巧，更充分地利用了 HAS 视频流的时序特征。

论文的结构安排如下：

第一章 论述本文的研究背景和研究意义，介绍了本文中相关工作的研究现状，并对现有工作与本文工作的不同之处进行了分析比较。

第二章 介绍本文研究过程中所应用到的技术知识和相关研究，包括 HTTP 自适应流媒体传输、时间序列数据挖掘和网络视频 QoE 的概念、技术概要。

第三章 介绍基于加密视频流特征分析的块序列重构。提出了一种基于 HAS 视频流的块序列重建算法。

第四章 介绍基于加密流量的 HAS 视频用户 QoE 的感知和预测。提出了一种全新的应用层客观 QoE 指标，并使用时间序列数据挖掘技术，实现从加密视频流量中实时预测 HAS 视频用户的 QoE。

第五章 总结本文工作，并指出了本文中现有工作的不足以及下一步研究的可行方向。

2 相关技术

本章介绍本文研究过程中所应用到的技术知识和相关研究，包括 HTTP 自适应流媒体传输、时间序列数据挖掘和网络视频 QoE 的相关概念、技术概要等内容。

2.1 HTTP 自适应流媒体传输技术

HTTP 自适应流媒体（HTTP Adaptive Streaming，HAS），是指基于 HTTP 协议的自适应传输流技术。该技术不仅继承了传统的流媒体技术和 HTTP 渐进式下载技术的特点，支持用户边下载边播放视频内容，而且引入了码率自适应调控逻辑，使用多码率编码源内容和动态码率请求算法，具有灵活的会话控制功能和智能的流量调节机制[4]。相比于传统的流式传输技术，HAS 给予了播放器更多的“自主权”，使得播放器可以根据当前的网络状态和用户服务参数，更合理地调控视频流的请求质量，在有限的网络资源下尽可能地满足用户的观看体验。

本节对 HAS 传输的工作机制、主流解决方案、码率自适应请求算法等相关内容进行介绍。

2.1.1 HAS 传输的工作机制

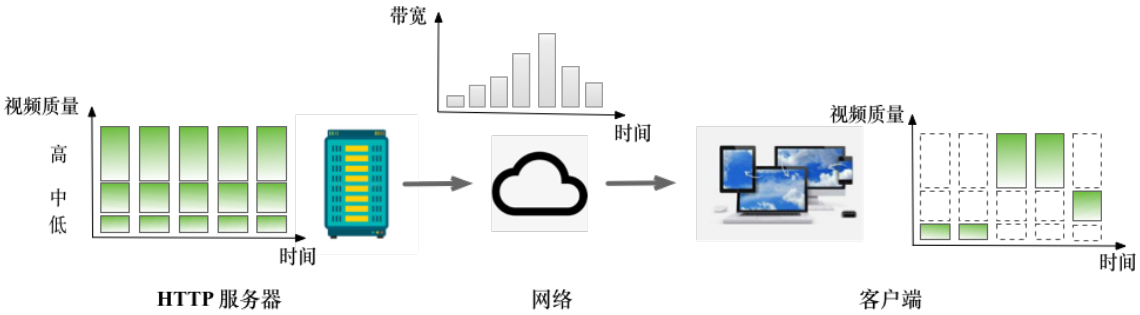


图 2-1 HAS 传输的工作机制[4]

Fig 2-1 General mechanism for HAS transmission[4]

图 2-1 是 HAS 传输机制的一般模型。在 HAS 中，视频会被编码为多种码率版本，每个版本的视频描述的媒体内容相同，但具有不同的质量等级[4]。每种版本的视频会被进一步分解为一个个持续时间相同的连续视频分块，播放器可以基于每个分块在版本间进行灵活切换。对于每一个媒体资源，服务器中都存储有该媒体的元信息描述文件，描述文件中包含了可支持切换的版本等级、块持续时间、URL 链接等媒体信息。描述文件的存在，就相当于给播放器提供了一份详细的媒体资料清

单，可以让播放器在每一次媒体文件正式传输之前就提前获知到该请求资源的所有详细信息，包括每一个视频块的分辨率和码率等，这样就允许播放器在视频演播过程中实时参考媒体的元数据参数，综合网络状态和演播参数及时合理地调整媒体分块的请求方案，保证视频流的平稳高质量播放。在一次会话开始时，播放器首先从服务器下载所请求媒体资源的元信息描述文件，参考描述文件内容，根据网络的实时条件，使用 HTTP/HTTPS 协议开始从服务器按照顺序依次请求媒体分块，每一个媒体分块都需要完整的被下载到本地播放器之后才可以进行播放。为了吸收网络带宽的波动性，播放器中设置有“缓冲区”，用于提前储备视频资源，以规避演播过程中因网络条件恶劣，媒体分块无法及时到达而导致的卡顿风险。在整个视频流式传输期间，播放器的自适应码率请求逻辑会综合网络带宽的估计情况和当前缓冲区的实时占有率，确定下一个媒体分块的请求码率，更智能地进行会话控制和流量调节。

2.1.2 HAS 传输的主流解决方案

目前，HAS 企业解决方案根据技术标准的不同可以分为三种：MSS（Microsoft Smooth Streaming）、DASH（Dynamic Adaptive Streaming over HTTP）和 HLS（HTTP Live Streaming）。这三种主流标准都是基于 HAS 传输技术的一般思想演化而来的，均采用了多版本分块存储和自适应码率请求的传输机制，但是在具体的媒体分块逻辑和传输控制策略上存在不同。相比于 MSS 技术，HLS 技术和 DASH 技术目前的商用范围更广、设备支持率更高，本节详细介绍这两种解决方案的区别和联系。

(1) HTTP Live Streaming（HLS）

HLS 是 Apple 公司设计的 HAS 整体解决方案，如图 2-2 所示。

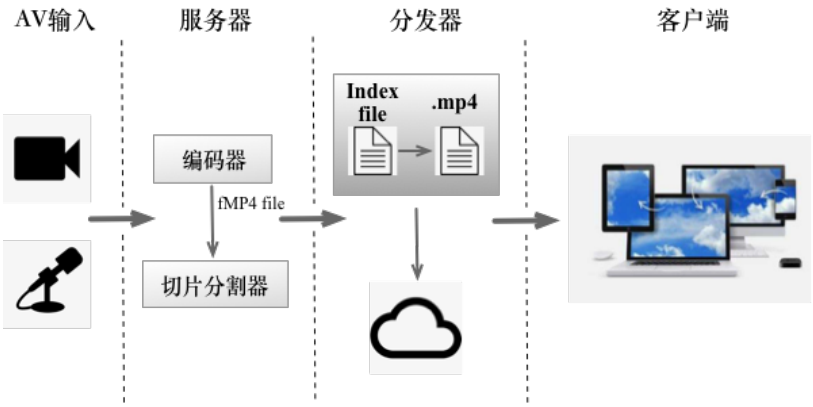


图 2-2 HLS 方案的基本框架

Fig 2-2 The basic framework of the HLS scheme

HLS 只请求基本的 HTTP 报文，与实时传输协议（RTP）不同，HLS 可以穿过任何允许 HTTP 数据通过的防火墙或者代理服务器[9]，此外，它也很容易使用内容分发网络来传输媒体流。目前，Apple 公司已经把 HLS 协议作为一个互联网草案提交至 IETF 组织讨论，成为正式的 RFC。

图 2-2 是 HLS 传输的基本框架，主要由三个部分组成：服务器组件、分发组件和客户端组件。在典型的 HLS 配置中，服务器中的媒体编码器接收到音频-视频的模拟信号输入，将其编码和封装为 HEVC 视频和 AC-3 音频，编码完成后的媒体文件会被流式处理为 MPEG-4 或 MPEG-2 格式的媒体流送给分块切割器，由分块切割器将媒体流分割成大量持续时间相同的 TS 分块文件存储在 Web 服务器上。与此同时，分发器会创建并维护一个包含媒体文件列表的 m3u8 索引文件，用于记录分块文件的 URL 等基本信息。客户端软件通过 HTTP 方式请求并读取索引文件，然后再按照顺序请求文件中列出的媒体分块并显示。

(2) Dynamic Adaptive Streaming over HTTP（DASH）

DASH 是第一个基于 HTTP 的自适应比特率流媒体解决方案，是一种正式标准。DASH 技术由 MPEG 主导开发，如图 2-3 所示。

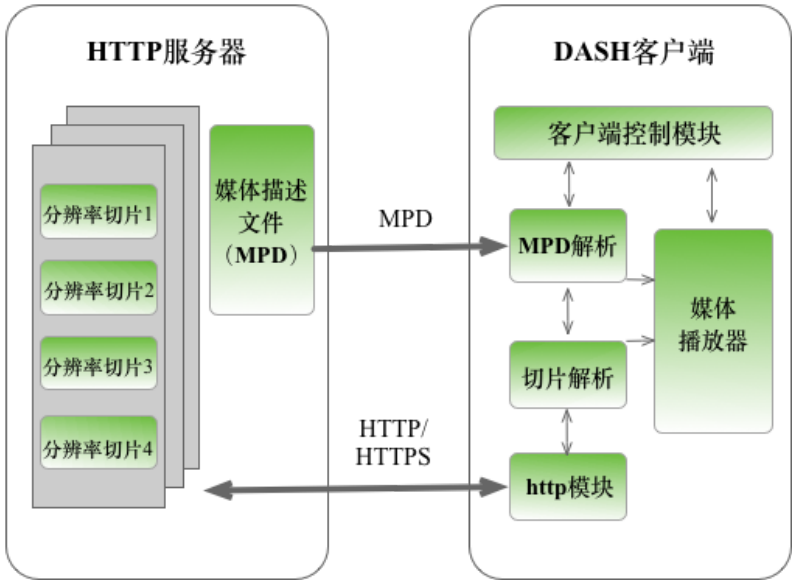


图 2-3 DASH 协议的系统框架
Fig 2-3 System framework of DASH protocol

图 2-3 是 DASH 协议的系统架构。在 DASH 的传输流程中，DASH 客户端首先获取的是 MPD 文件，MPD 文件是媒体文件的描述文件（Manifest），作用类似于 HLS 的 m3u8 文件，MPD 文件以 XML 格式组织。DASH 的工作原理和传输流程与 HLS 基本相同：即他们都是需要将视频流进行分块处理，并基于 HTTP 协议进行传输，客户端都具备自适应调控逻辑。与 HLS 不同的是，DASH 在分块时，

不是按照同一种分辨率或是编码格式进行分块，DASH 中同一条媒体文件会存在多种分辨率/编码的分块组合。因此，相比于 HLS，DASH 的灵活度更高，可以适应的应用场景也更加宽泛，能够无缝地适应不断变化的网络条件，并且以比较少的卡顿或重新缓冲事件提供高质量的视频播放。表格 2-1 中对 HLS 与 DASH 这两种 HAS 技术解决方案之前的区别和联系进行罗列。

表 2-1 HLS 与 DASH 的异同点
Table 2-1 Similarities and differences between HLS and DASH

项目	DASH	HLS
类型	开放的，基于标准	Apple 控制
源视频编码器	H.264 等	H.264
源音频编码器	AAC 等	AAC,MP3
分块格式	MP4 格式 MPEG-2 TS	MPEG-2 TS
音频视频复用	复用在一个分块或音视频分开	复用在一个分块
分片和发送	标准 HTTP 流媒体服务器	标准 HTTP 流媒体服务器
播放	3GPP Rel9/MPEG 客户端	苹果 ios/QT X
DRM 保护	灵活的（如 OMA 或 UV）	AES-128 加密
典型分块时长	灵活的	10s
自适应控制	客户端	客户端

2.1.3 HAS 传输的码率自适应请求算法

无论是 HLS、DASH 解决方案，还是一般的 HAS 传输机制，与传统的流媒体技术相比，最显著的区别就是支持客户端根据网络条件和用户服务参数动态地调整媒体资源的请求质量，这种智能化码率请求机制的应用，一方面归功于 HAS 中服务器端对媒体文件采用“多质量等级分块”的存储方式，另一方面，主要得益于 HAS 在客户端中引入的码率自适应请求算法（Adaptive BitRate algorithms，ABR）。

ABR 算法工作于 HAS 客户端，通过持续监控网络带宽、设备 CPU 性能、视频显示质量等终端设备参数来评估视频用户网络连接的状态，并依据这些参数实时的调整视频流的请求质量，从而自动地适应用户网络和播放条件的任何变化。ABR 算法采用“尽力而为、量入为出、最优规划”的设计思想，从用户体验出发，在现有资源的条件下尽可能的满足用户的观看体验，这就意味着客户端在应用 ABR 算法时请求的不一定总是最高质量的视频流，但是却一定是设备在当前硬件条件和网络资源下可以播放的最高质量的流。

现有的 ABR 算法的设计逻辑主要分为两大类：基于缓冲区占用率[47,48]；同时基于网络吞吐量预测和缓冲区占用率[49-54]。在这两类逻辑的基础上设计出的 ABR 算法中最具有代表性的解决方案是：MPC（Model Predictive Control）、BOLA（Buffer Occupancy based Lyapunov Algorithm）、HYB（HYBrid）。

(1) MPC[54]（基于吞吐量预测和缓冲区占用；通过求解最优化问题选择视频块的比特率）：MPC 以最近请求的五个视频块的下载吞吐量作为样本，预测下一个块的网络吞吐量，并基于该吞吐量的预测值选择视频块的请求码率。在这种思想的基础上，衍生出了两个 MPC 版本：FastMPC 和 RobustMPC。FastMPC 算法中引入了“调和平均预测器”估计吞吐量；RobustMPC 算法通过衰减因子$1+d$限制吞吐量的预测值，其中d是过去五个块下载时吞吐量的最大预测误差。

(2) BOLA[48]（基于缓冲区占用率；通过求解最优化函数来选择视频块的码率）：BOLA 是一种基于缓冲区占用率的 ABR 算法，目前已经成功部署于开源的 DASH 流媒体播放器 Dash.js[55,56]中。BOLA 在进行码率决策时不需要进行吞吐量预测，而是把不同的缓冲区占用率映射到不同的码率等级上来，从而将视频块的码率选择模型转化成一个最优化问题。BOLA 将缓冲区占有率的最低阈值与其想要维持的最理想阈值之间的比值γ作为整个模型的限制参数，用来决定模型在当前的网络条件和缓冲区水平下，应该选择多大码率的视频块，才能够尽可能的将缓冲区占用率维持在理想阈值之上并且不会发生重新缓冲。通俗地说，γ代表着 BOLA 想要避免重新缓冲事件发生的这种意愿的“强烈程度”，γ值越高，BOLA 的码率决策也越保守。

(3) HYB（基于吞吐量预测和缓冲区占用率；使用启发式算法选择码率）：HYB 是单词 hybrid 的简称，同时基于吞吐量的预测值和缓冲区占用率选择视频块的请求码率。对于每一个视频块的请求码率，HYB 在避免重新缓冲事件发生的前提下，选择能够获取到的最高码率视频块。具体地说，HYB 将$S_j(i)$定义为码率为i的第j个视频块的大小，B是根据过去块的下载流量对当前网络吞吐量的预测值，L是当前时刻的缓冲区占有率（存储的视频块时长）。HYB 在满足$S_j(i)/B < L \times B$的前提下选择最大的i值，也就是选择最高的视频块码率。这里的B是$(0,1)$之间的一个数值，B越大代表可以选择的i值也相对更大，此时的 HYB 码率决策也更“大胆”。通过调整比例因子B，可以在一定程度上抵消吞吐量的预测误差并且补偿该算法中贪婪求解的弊端，有效地应对意外的网络吞吐量骤降给未来的缓冲事件带来的影响。

2.2 时间序列数据挖掘

时间序列数据挖掘（Time Series Data Mining，TSDM），是指从大量的时间序列数据中通过算法挖掘隐藏于其中、与时间属性相关的信息，并将信息转换为人们可理解的结构，用于进一步指导人们社会、生活和经济等活动的过程。时间序列数据挖掘是一种涉及机器学习、统计学和数据库理论的交叉方法[57]。因这种数据类型分布领域的广泛性和丰富性，时间序列数据挖掘具有丰富的应用场景和研究意义，目前已经成为数据挖掘的研究热点之一。

本节主要对时间序列的概念、时序数据挖掘的主要任务、时序预测的常见处理角度等相关内容进行介绍。

2.2.1 时间序列的概念

时间序列是指按照时间顺序排列的一组数据，是一种重要的高维数据类型。原则上，同一条时间序列是对同一个客观对象的同一个物理量，按照时间顺序进行采样并排列组成的一组序列。因此，几乎所有按照顺序记录的数值都可以存储为时间序列，比如股票价格、天气条件、物理系统参数变化、产品质量监控等[57]，所以，时间序列几乎存在于所有的科学和实践领域中。时间序列可以是单变量的，即仅记录一个变量的时序变化情况，也可以是多变量的，即在同一时间点对来自不同来源的一组观察进行记录，组合作为该系统该时刻点下的一个样本值，将这样按照时间顺序采集到的数据集称为多元时间序列。

2.2.2 时序数据挖掘的主要任务

时间序列数据挖掘的任务可以定义为从时序数据点集合中提取有意义的信息。根据研究目的不同，可以分为：分类、聚类、预测、分割和相似度分析等。本节重点介绍时间序列分类、聚类和预测三种主要任务。

(1) 时序分类

时间序列分类是把整个时间序列当作输入，并赋予这个序列某个离散标记的过程。在实际应用时，时序数据往往是无标注的，此时可以使用时序聚类的方法，先对部分时间序列进行聚类分析，将属于同一个簇的时间序列进行特征提取的同时并对每个簇进行标签化，然后将标签簇中的时序数据作为训练集。在训练分类器的过程中，将未标注的时序数据作为测试集，与训练集中的时间序列基于某种距离度量指标进行相似度分析，从而完成类别判断。时间序列分类任务中常用三种分类

算法：K-最近邻、支持向量机（SVM）和随机森林。K-最近邻算法是将目标时间序列分配给数据集中与其“最相似”的 K 条时间序列多数属于的类别中。随机森林是一种用于分类的集成学习算法，在训练时构建多条决策树，通过综合考虑每棵决策树的分类结果对时序数据进行分类。除了这三种算法，人工神经网络（Artificial Neural Network，ANN）、贝叶斯网络、长短期记忆神经网络（Long Short-Term Memory，LSTM）等机器学习、深度学习算法，也可以用来对时间序列进行分类。

无论是哪一种分类算法，都需要对目标时序与已知时序基于某种度量指标进行相似度分析，方可进行距离“远近”的判断，所以在时间序列分类问题中，距离度量指标的选取尤为重要。这里的指标可以是基于时间序列的形状和总体变化趋势的，例如欧式距离，动态时间扭曲（DTW）；也可以是基于从序列中提取到的特征，例如频率、幅度、斜率等。此外，时间序列分类任务的一个最主要障碍就是该数据类型下时间维度过高的问题，大多数时间序列分类任务的首要步骤就是对时序进行降维，常用到的算法有奇异值分解（SVD）、主成分分析（PCA）。

(2) 时序聚类

时间序列聚类的目标是对数据集中的时间序列根据某些原则进行分簇或是找到数据集的自然分簇和结构。与分类任务类似，同一个簇中的时间序列必须根据某种距离度量指标进行相似性分析。传统的聚类算法有自组织映射（SOM）、隐马尔可夫模型（HMM）或支持向量机。目前应用最广泛的聚类算法是 K-means（K-均值），K-means 首先从数据集中随机地选择 K 个时间序列作为聚类中心，然后根据数据集中目标时序到中心时序的距离，将它们分配到最“近”的簇中，这种过程需要重复的进行多次直到 K 个簇均收敛，即簇与簇之间的时序方差达到最大阈值，同一个簇内的时序方差达到最小阈值。针对时间序列本身性质的不同，在进行时序聚类时会存在不同的操作方式，例如由于时间序列的高维性，在对时序数据进行聚类分析时往往先需要进行切片处理，将数据切分成一个个等长的子序列。如果时间序列存在周期性，就可以先将时序数据分割成长度等于周期的非重叠切片，然后再使用常规的聚类算法对其进行分析；当时间序列不具有周期性时，如果同样使用非重叠切片的方法，很可能会分裂时序数据的重要结构信息，此时就需要从该任务本身出发，结合所研究问题的先验知识对非周期时序数据进行重叠切片处理，例如在对心电图（ECG）序列进行切片分析时。

(3) 时序预测

时间序列预测任务是基于过去的观察值估计连续时间序列的未来值。比如根据过去几周的记录，预测明天股票的收盘指数或是网店商品价格。时间序列预测需要建立观测变量随着时间变化的行为模型。最简单直观的模型是回归算法，以一定的误差尽可能地拟合时间序列的数据变化曲线。如果某个变量的下一个数值紧密

地依赖于之前数值的变化，例如航班延误、以及气象、经济场景中的时间序列，就可以采用差分整合移动自回归模型（Auto Regressive Integrated Moving Average model，ARIMA）进行时序预测，但是，这种方法需要时序数据具有平稳性，即数据的均值和方差在时间维度上不随系统发生变化，然而现实中很多场景下的时间序列都不具备平稳性，比如网络流量变化、设备故障告警等，此时，可以考虑使用循环神经网络（Recurrent Neural Network，RNN）等深度学习方法，通过自动化地挖掘时序特征、隐式地表达序列的前后依赖关系进行时序预测。

由于本文中主要的时序挖掘任务属于预测问题，有关于时序预测的常见处理角度我们会在 2.2.3 节进行详细介绍。

2.2.3 时序预测的常见处理角度

时序预测问题的常见处理思路可以主要分为三种：从统计学的角度、从经典机器学习算法的角度、从卷积神经网络等深度学习模型的角度。

(1) 从统计学的角度

模型分析法是统计学中研究时序预测问题的主要方法，也是获取到一个时间序列后对其进行初步分析的最自然、最直接的方法。这类方法中最具有代表性的经典模型有：自回归差分移动平均模型（ARIMA）、广义自回归条件异方差模型（GARCH）、马尔可夫链模型（MC）等。这类模型的最大特点是直接从预测变量本身Y的角度建立回归模型进行预测，比如股价预测问题中，预估T时刻的股价$y(T)$，就需要对过去一段时间股价的历史信息$\{y(1),\dots,y(t),\dots,y(T-1)\}$，从诸如多阶差分、自相关函数等角度进行统计分析，用以计算回归模型的相关参数，从而通过直接拟合股价变化曲线预测$y(T)$。然而这个角度的研究存在三个显而易见的弊端：

1) 只能够使用预测变量本身的时序变化特征，无法综合考虑到其他与预测变量相关的影响因素，比如股价预测问题中的政策因素、企业舆情等。

2) 表面上是在时间维度上建立回归预测模型，但是中心思想就是根据以往的数据记录，分别预测$y(T), y(T+1)$ ...，并没有真正考虑到这些预测变量本身的时间相关性。

3) 由于模型本身的时间复杂度，只能够处理有限时长的序列，可以考虑的时间窗口有限。

(2) 从经典机器学习算法的角度

与统计模型相比，从机器学习算法的角度进行时间序列预测，可以同时考虑到与预测变量相关的其他影响因素。通过同时对预测变量和其他影响因素的历史记录序列进行特征提取，可以在每一个时刻点下获取到“多元的特征组合”，从而将简

单的一元时序预测转换为多元时序预测。与统计模型相比，机器学习算法可以通过丰富的特征工程手段，将时间序列的历史信息尽可能多的建模在每一个样本点的特征中，不需要进行窗口滑动机制或是自回归机制，规避了统计模型中因为时间序列的高维度性所带来的时间复杂度的问题。但是，这种方法实际上是从时间上把每一个时刻点的数据打散，将一个时序预测问题转换为非时序预测问题，在对每一个样本点引入更多特征信息的同时，带来两点弊端：

1) 受掣于特征工程的有效性，人为地割裂了序列之间的时间相关性。

2) 模型不具有记忆性，能够处理的序列时长有限，两个时间距离越远的样本点残留的时序信息越少。

(3) 从循环神经网络等深度学习模型的角度

相比于统计学方法和经典机器学习算法，近年来流行的循环神经网络（Recurrent Neural Network, RNN）等深度学习模型不仅可以在每个时刻点下同时考虑多维影响因素，还可以对每一个时刻点下特征组合的时间依赖关系进行显式建模，如建立数学映射模型：$f\big(X(1),X(2),\dots,X(T)\big)\rightarrow y(1),y(2),\dots,y(T)$，其中$X(T)$是在$T$时刻下的多维影响因素的特征组合，$y(T)$是$T$时刻预测变量的数值，如图 2-4 所示。

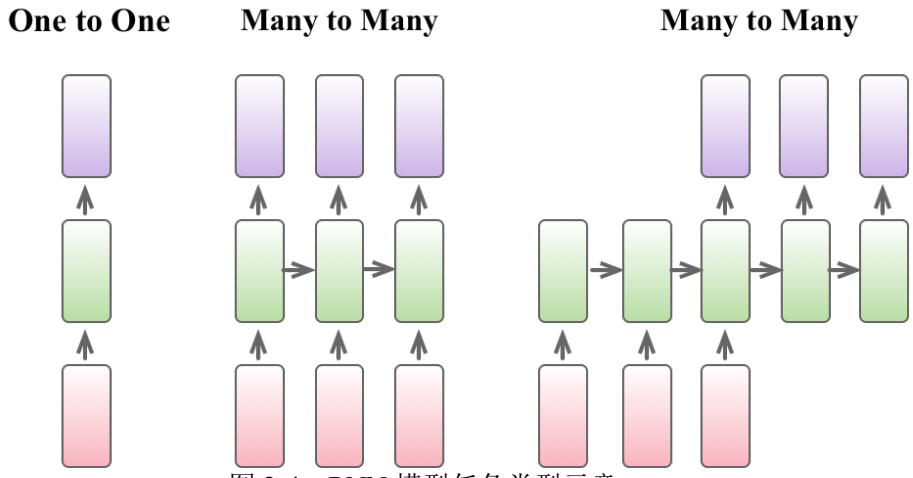


图 2-4 RNN 模型任务类型示意

Fig 2-4 Common task types of RNN

图 2-4 是 RNN 模型的常见任务类型，绿色框是 RNN 的神经元结构，红色框是输入X，紫色框是预测目标y。其中 One to One 任务类型可以代指经典机器学习算法的模型结构。使用机器学习算法进行时序预测时，其实是将时序预测问题转换为了非时序预测问题，每一个时刻样本点得到一个输出值，这种处理方式下模型看待时间序列的视野是单一的，即建立的数学映射模型是：$f\big(X(T)\big)\rightarrow y(T)$；而在 RNN 中，可以通过 Many to Many 的任务结构建立起$(sequenceX,sequenceY)$之间

的映射，这里的$sequenceX$和$sequenceY$即分别代指$X(1), X(2), \ldots, X(T)$和$y(1), y(2), \ldots, y(T)$。并且，相对于普通神经网络的各个计算结果之间相互独立的特点，RNN 的每一个隐藏层的计算结果都与当前输入以及上一次隐藏层的结果相关。通过这种方法，RNN 的在当前时刻下的运算便具备了“记忆”之前几次结果的能力。

2.3 网络视频 QoE

本文的主要落脚点是对网络视频用户 QoE 进行感知和预测，涉及到网络视频 QoE 评估技术的研究和讨论，因此，本节首先对用户体验质量的概念、网络视频质量评估的概念、网络视频质量评估的模型分析法等相关方面进行介绍，具体的研究方法和研究成果将在第四章进行描述。

2.3.1 用户体验质量的概念

用户体验质量（Quality of Experience, QoE）是用于衡量用户在体验某项服务（比如网页浏览、视频观看等）时喜悦或是烦恼程度的指标，提供了用户对特定商品、服务或应用的期望、认知和满意度的评估[45]。

QoE 来源于 QoS（Quality of Service，服务质量），但是又与 QoS 不同，虽然 QoE 的评价对象常常是服务本身和支撑服务的网络，但是评价主体始终都是终端用户，而 QoS 机制主要是从网络的角度进行业务评估，不涉及终端用户的主观感受测量。

QoE 是一种主观指标，用户的主观感受可能会受到整个服务系统中各部分参数的影响，这些影响因素根据支持服务的不同对象主体，可以分为四类：

(1) 系统：主要是构成服务的技术系统中的因素，比如网络中的丢包、延时，终端设备的性能（屏幕的大小、像素等），播放器的缓冲方式；
(2) 环境：用户所处的环境因素，比如地点、时间等；
(3) 内容：服务本身的特征，比如编码格式、帧率、内容流行度等；
(4) 用户：用户自身的心理因素，比如浏览喜好、视觉疲乏度等。

2.3.2 网络视频质量评估的概念

网络视频服务广泛地应用于视频会议、视频点播、远程医疗等领域。QoE 可以从用户的角度反映提供商的服务质量。所以，网络视频服务提供商可以通过感知用户 QoE 对自己的服务质量进行监控，并基于 QoE 感知结果在用户投诉或是放弃服

务之前对客户体验进行改进。目前基于网络视频 QoE 评估的相关研究已经成为了一个热点方向。

网络视频质量评估是针对网络视频的传输流，从用户质量体验出发评估和测量网络视频流的质量[9]。现有的网络视频流 QoE 质量评估主要分为两种：主观评价法、客观评价法。

(1) 主观评价法

主观 QoE 评价方法是指通过让用户对业务做出主观评价直接获取用户的质量感受。为了获得用户的感知，需要选择一定数量的测试用户，通过对测试用户实际使用服务时的体验进行纪录，并对记录进行 QoE 分数定量，得到用户对该服务的主观打分。这种方法直接从用户自身出发，模拟真实的服务场景，可以最大可能地反应用户的主观体验。

但是这种方法实施难度比较大，严格受控的实验环境，虽然可以有针对性地选择每次参与实验的影响因素类型，但是也同时带来了实施步骤复杂、代价大、可移植性差的问题[45]，难以进行大规模的测量和验证，并且会由于测试人员本身的差异性影响打分结果，不具有通用性、说服力较差。

(2) 客观评价法

与主观评价方法截然不同，客观评价法的应用过程中不需要用户的主观评价，仅通过业务输出内容的质量水平或服务的原始质量的失真水平来评价业务[9]，例如，在 HAS 视频服务中，内容服务提供商可以通过将视频演播过程中的平均码率、卡顿次数、启动延时等 QoS 指标与用户的 QoE 联系起来，建立两者之间的数学函数关系，通过测量和收集这些与 QoE 相关的客观 QoS 指标，间接客观地反映用户体验质量。

因此，客观评价方法具有简单易用、方便实施、通用性强的特点。但是，为了尽可能准确地通过客观参数感知主观体验，就需要同时保证 QoS 指标与 QoE 分数之间数学函数关系的准确性和可解释性，而这不仅需要足够的专业知识，也需要对服务的类型和用户群体倾向进行认真考虑。

现有的网络视频质量评估的研究大都采用伪主观方法，也叫模型分析法。本文第二部分的工作将主要基于模型法对 HAS 视频用户的 QoE 进行感知和预测，所以本章将网络视频质量评估的模型分析法单独放在 2.3.3 节进行详细介绍。

2.3.3 网络视频质量评估的模型分析法

模型法，又叫伪主观评价法，是一种在主观评价法和客观评价法基础上发展起来的集成评价方法，该方法不仅引入了客观评价方法中模型参数映射的思想，而且

会进行测试样本收集，将受测人员对服务的主观评分作为 QoE 度量标准，用于质量评估模型的训练和验证。所以这种方法在简单易用的同时，又会具有较高的用户主观体验感知精度。模型分析法应用在网络视频质量评估中的主要步骤，如图 2-5 所示。

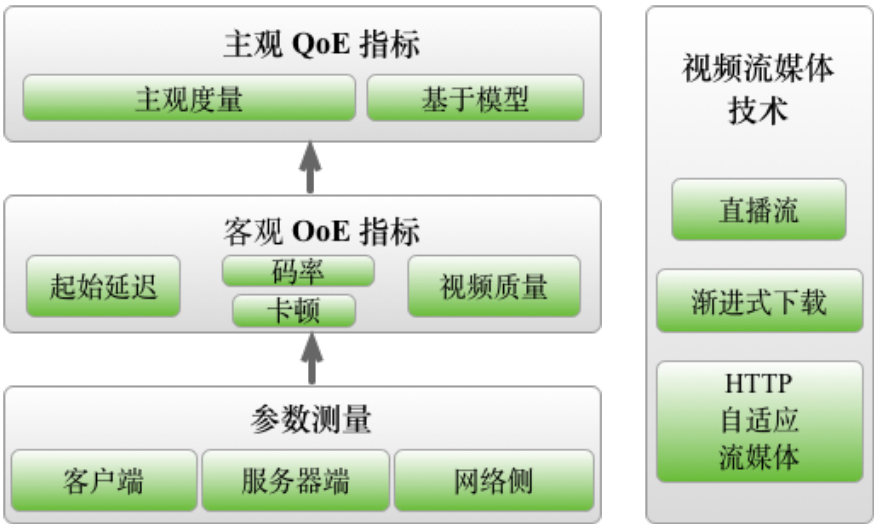


图 2-5 网络视频 QoE 模型评估法

Fig 2-5 Network Video QoE Model Evaluation

(1) 根据视频服务的特点（如直播或点播），确定合理的客观 QoE 度量指标（例如卡顿率、平均码率等）；

(2) 根据业务逻辑和应用架构，确定与客观 QoE 度量指标相关的全部或是关键 QoS 参数（例如编码机制、网络丢包、重传等）；

(3) 确定测试方案和部署测试环境，组织受测人员对服务进行体验评分；

(4) 整理并筛选带有 QoE 评分的样本数据；

(5) 通过机器学习等方法，计算 QoS 参数与客观 QoE 指标之间的映射关系，建立客观 QoE 模型；

(6) 通过数理统计等方法，结合经验，确定不同客观 QoE 指标的影响权重，设计客观 QoE 指标与用户主观 QoE 打分之间的数学函数。

由此可见，网络视频质量评估的模型分析法中存在两种模型的建立，分别是客观 QoE 模型和主观 QoE 函数。本文中，我们的研究侧重于客观 QoE 模型的建立，根据 HAS 视频的实际演播特征确定应用层客观 QoE 度量指标，以及与该指标相关的网络流量特征，使用机器学习、深度学习算法，对视频用户的 QoE 进行实时感知和预测。

2.4 本章小结

本章主要介绍了本文研究过程中涉及到的主要技术知识和相关研究，包括 HAS 视频流、时间序列数据挖掘、网络视频 QoE。

首先，介绍了 HAS 视频流的一般工作机制，阐述了目前 HAS 传输技术中三种主流的企业解决方案：MSS、HLS、DASH，并从分块逻辑等角度比较了 HLS 与 DASH 之间的异同，论述了三种最具有代表性的 ABR 算法：MPC、BOLA、HYB。

然后，介绍了时间序列数据挖掘技术，阐述了时间序列的概念，介绍了时序数据挖掘技术的三种主要任务：分类、预测、聚类，并且详细论述了时序预测任务的常见处理角度。

最后，介绍了网络视频质量评估的一般概念，重点阐述了网络视频 QoE 的模型分析法。

本文的研究内容、研究方法、研究成果会在后续章节中进行详细介绍。

3 基于加密视频流特征分析的块序列重构

HAS 视频传输机制中，码率自适应请求算法（ABR）的使用，给予了视频客户端抵御网络流量波动的能力，允许客户端在网络条件复杂的情况下，通过动态地调整视频块的请求质量（不同码率的视频块），实现对播放器演播状态的调节，可以说，该算法保证客户端能够自适应网络变化，有效保障了视频用户的 QoE。但是，对于运营商来说，客户端的 ABR 算法影响了它从网络流量中对用户 QoE 的感知，导致运营商无法准确区分用户 QoE 的调节效果是来源于客户端的 ABR 调控机制，还是自己对于网络资源的调度。换句话说，客户端 ABR 算法之于运营商，相当于在运营商本就狭窄有限的视野下，进一步“蒙蔽了她的双眼”！因此，运营商若想要更准确地感知用户 QoE，首先就需要能够从网络流量中挖掘出客户端的 ABR 调控信息，用于指导其推测播放器的演播状态，规避其因为“视野狭窄和蒙蔽”所带来的对 QoE 调节效果误判的问题。

我们发现，实际中，客户端根据 ABR 算法对视频内容按照“块”的粒度进行请求，相较于网络流量的“包”序列，视频流的块序列可以更直观地反映客户端的 ABR 调控信息。因此，本章主要研究如何从加密的 HAS 视频流量中重建视频块序列。为了重建视频块序列，本章首先基于网络测量的方法对加密视频流的传输特性进行了分析，总结了 HAS 技术在实际应用时的流量传输特征，并根据分析结果提出一种 HAS 视频流块序列重建算法。

3.1 问题描述

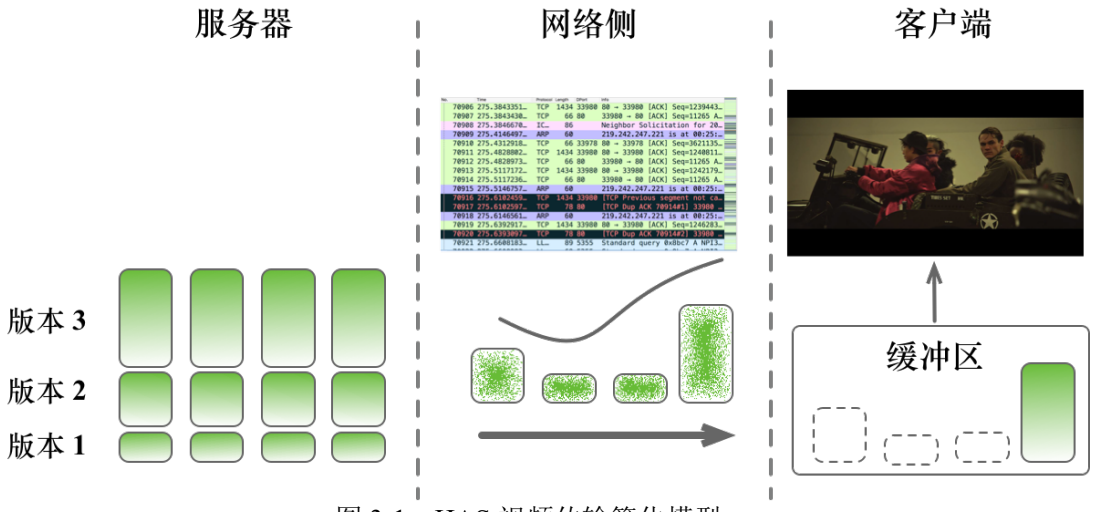


图 3-1 HAS 视频传输简化模型

Fig 3-1 Simplified model of HAS video transmission

图 3-1 是 HAS 视频传输的简化模型。如图所示，HAS 视频传输机制中，服务器端对视频内容按照“块”的粒度进行存储，客户端根据 ABR 算法对视频内容按照“块”的粒度进行请求。但是，当视频块数据在网络中进行传输的时候，由于网络最大数据传输单元的容量限制，视频块会被分解成一个个 TCP 数据包进行传输[59]，这种处理方式会在一定程度上模糊甚至破坏掉网络流量序列中本来包含的客户端 ABR 调控信息（例如视频块请求时刻、码率等级、码率切换水平等），影响运营商基于网络流量评估用户 QoE。

我们发现，相较于“包”粒度，“块”粒度的视频流量序列能够更大可能性地保留客户端的 ABR 调控信息，方便运营商了解客户端对视频块请求的调控情况，指导其更有效地感知播放器的演播状态和了解用户的观看体验。因此，我们认为，在基于网络流量感知用户 QoE 之前，首先需要对视频流量进行“块粒度”重建得到“视频块序列”。重建视频块序列可以在数据预处理的层面上，重构出网络视频流量（即以“分组”或“TCP 报文”为颗粒度的流量）中那些“隐藏”了的客户端演播信息，有助于后期提取流量特征、建立 QoS-QoE 的视频用户体验质量映射模型。

但是，在对视频流量进行“块”粒度重建时，需要解决三个难题：

(1) 确定分块传输边界。在重构视频块序列时，首先需要从报文序列中准确地找到每一个视频分块的传输边界（即每一个视频分块传输的起始报文和终止报文），然后才可以根据分块传输边界进行流量聚合（将原本属于同一个分块的所有封包的字节数累加到一起），得到每一个视频块。因此，分块传输边界的界定原则直接关系到最终视频块序列的重构质量。

(2) 区分音频块和视频块。HAS 机制中，相较于采用多码率版本进行存储和请求的视频块，同一个视频的音频块通常只有一种码率。对比视频块序列，音频块序列的变化情况基本不反映用户的体验质量，所以，我们需要从重建的块序列中找到并剔除音频块，得到纯粹的视频块序列，方便后期通过时序分析的方法对其进行研究。因此，如何从视频流量中区分音频块和视频块也是一个需要考虑的问题。

(3) 网络流量加密。HAS 视频流量加密传输的普及，在保护用户隐私的同时，会隐藏掉网络流量中包含的视频明文信息（比如报文请求内容的媒体类型、码率等），“剥夺”了原本可以使用封包明文信息指导修正块序列重建工作的可能性，为从网络流量中重构视频块序列进一步增大了难度。

现有的研究中，已经存在从视频流量中重建视频块序列的相关工作。文献[3]中同时定义了三条规则（HTTP request 标志、下载持续时间阈值标志、TCP RST/FIN 标志）用来确定分块的传输边界，并使用统计分析的方法，将每一个等级的视频码率与一定范围的分块大小一一对应，从而实现根据分块的大小从流量中区分音频块和视频块，但是这种方法中没有考虑到视频流“双流并传”和“分块并传”（3.2

节中对该特性进行了详细描述）对分块传输边界识别的影响，音频块和视频块的区分准则也只适用于同一条视频反复播放的情况下，不具有通用性。文献[14][15]和[59][60]中使用“同块同 ACK 号”规则（拆分自同一个视频块的网络报文的 ACK 号相同），从流量中查找分块的传输边界并进行流量聚合，但是，即便是使用该规则对 HAS 视频流量进行块粒度重建时，仍然会因为 HAS 视频流的实际传输特性而产生阻碍。

因此，为了更好地解决上面提到的三个主要难题，更有效地从加密视频流量中准确地重建视频块序列，我们首先基于网络测量对 HAS 加密流量特征进行分析，并根据分析结果提出了一种视频块序列重建算法。

3.2 HAS 视频加密流量特征分析

3.2.1 测量环境

首先对本章研究内容的测量环境进行简单介绍，如图 3-2 所示。

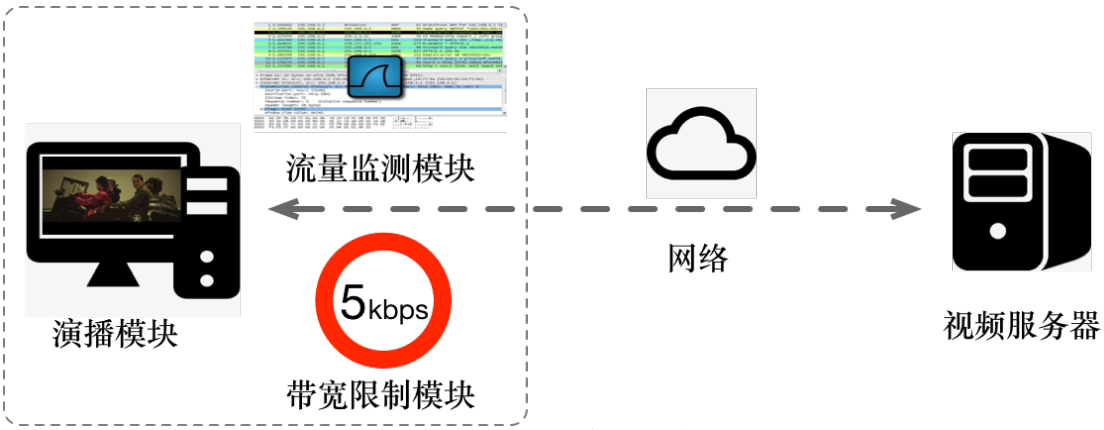


图 3-2 实验环境

Fig 3-2 Experiment environmet

为了基于网络测量对加密视频流的传输特性进行分析，我们搭建了受控的实验数据采集和测量环境，如图 3-2 所示。我们的实验环境主要包含三个模块：演播模块、流量监测模块、带宽限制模块。我们使用一台运行 Ubuntu Kylin 16.04 LTS 操作系统的 PC 作为客户端，通过加载 Firefox 浏览器实现视频的点播和显示；使用 Wireshark 网络封包分析软件作为流量监测模块，用于对视频流量数据进行采集和分析；通过调用 Linux 内核中的 TC（Traffic Control）组件作为带宽限制模块，用以适时地限制 PC 网卡的流量传输速率，控制播放器观看视频时的网络环境。我

们编写了自动化运行脚本，流程化地调用这三个模块的功能，实现视频源的选择与点播、播放日志的爬取与记录、网络流量的采集与保存本地等功能。

此处，需要额外强调的是，由于不同的 HAS 客户端存在不同的码率自适应请求逻辑，所以在实际测量时，使用不同客户端测量到的 HAS 视频流传输特性可能会存在一定的差异。本文中使用的 HAS 视频客户端是开源视频播放器 DASH Reference Client 2.9.3[55,56]。

常见的加密传输协议有 TLS/SSL（Transport Layer Security/Secure Sockets Layer）、PCT（Private Communications Transport）等。本文针对的是基于 TLS/SSL 和 HTTP 协议发展起来的 HTTPS（Hypertext Transfer Protocol Secure）加密传输协议。HTTPS 协议是由 HTTP+SSL 协议构建的可进行加密传输、身份认证的网络协议，通过在应用层（HTTP）和传输层（TCP/UDP）之间加入 SSL/TLS 层，隐藏掉网络中 HTTP 报文的明文信息，对用户发起的服务类型进行加密，从而实现用户的隐私保护。

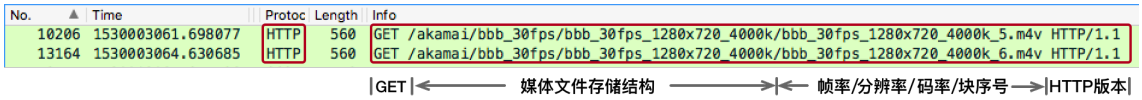


图 3-3 HAS 未加密视频流报文

Fig 3-3 HAS unencrypted video stream message

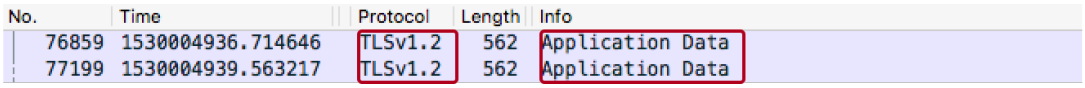


图 3-4 HAS 加密视频流报文

Fig 3-4 HAS encrypted video stream message

图 3-3 和图 3-4 分别是同一个视频在未加密传输和加密传输时的请求报文。图 3-3 中显示，在未加密传输的情况下，除了标准时间、协议版本、包长以外，请求数据包的报文中还包含了本次请求内容的明文信息（我们在此处隐藏了报文的源 IP 地址和目的 IP 地址），例如该视频块在服务器中的存储结构；该视频块的帧率、分辨率、码率、分块序号以及使用的 HTTP 协议版本，以图 3-3 中的 10206#数据包为例，本次请求的媒体分块是一个 m4v 格式的视频的第 5 个分块，该视频的帧率是 30fps、分辨率是 1280*720、码率是 4000kbps。然而，对于同一个媒体文件的加密请求包，如图 3-4 中的 76859#、77199#数据包，我们可以看到，报文中同样位置处的信息被“Application Data”替代隐藏。

本章的后续小节中，我们对 HAS 加密视频流的传输特性进行分析，介绍观察到的 HAS 视频的四种传输特征：双流并传、分块并传、同块同 ACK 号、分块重传。

3.2.2 双流并传特征

在同一段视频流中，将具有相同五元组（即源和目的 IP 地址、源和目的端口、传输协议）的流量称为一条 TCP 流。在一次视频演播过程中，我们对捕获到的视频流按照 TCP 流的数目进行统计，观察到 HAS 视频在整个传输过程中先后会创建多条 TCP 流，但是在同一段时间下，网络中最多只会存在两条 TCP 流并行传输，文献[60]中将这种现象叫做 HAS 视频流的“双流传输特征”。图 3-5 是我们对同一条 HAS 视频的单向网络流量基于“IP+端口号”的统计结果。

如图 3-5 所示，本次获取到的流量数据中，来自 IP 地址为 219.242.247.210 的视频流量占捕获总流量的 50.82%（图中红色加粗方框），其中有 99.87%的视频流量以 TCP 流的方式先后使用客户端的四个端口（33972#、33978#、33980#、33970#）进行传输。下面的描述中，我们使用客户端开启的端口号代表每个 TCP 流的序号，例如将通过客户端 33970#端口与服务器之间的 TCP 流量称为 33970#TCP 流。

Topic / Item	Count	Rate (ms)	Percent ▼	Burst rate	Burst start
▼ Destinations and Ports	83282	0.1375	100%	3.5800	52.995
▼ 219.242.247.210	42324	0.0699	50.82%	1.7900	52.995
▼ TCP	42270	0.0698	99.87%	1.7900	52.995
33972	27317	0.0451	64.63%	1.7900	52.995
33978	8795	0.0145	20.81%	0.4800	105.404
33980	3313	0.0055	7.84%	0.1300	123.137
33970	2595	0.0043	6.14%	0.8700	29.322
47054	61	0.0001	0.14%	0.1800	491.742
45586	31	0.0001	0.07%	0.1600	605.983
45588	24	0.0000	0.06%	0.2200	606.034
60696	15	0.0000	0.04%	0.0200	491.201
38534	13	0.0000	0.03%	0.0400	24.607
36084	13	0.0000	0.03%	0.0300	104.421
60612	11	0.0000	0.03%	0.0100	10.205

图 3-5 流量统计

Fig 3-5 Traffic Statistics

图 3-6 和图 3-7 分别是 33970#TCP 流和 33972#TCP 流的时间序列，描述了流量（TCP 封包的序列号）随着时间的变化情况。图 3-6 和图 3-7 中描绘的时间序列，来源于客户端在请求同一条 HAS 视频的过程中开启的两条 TCP 流。我们以 600 秒为一次捕获周期，对该视频文件的网络流量进行抓取，发现客户端先后开启了四条 TCP 流用于该视频文件的传输，其中每条 TCP 流的持续时间从 65s 到 500s 不等，通过包解析可以获得每条 TCP 流传输的起止时间：第 33970#TCP 流从 2 秒持续到 117 秒（如图 3-6 中的坐标横轴所示），第 33972#TCP 流从 2 秒持续到 67 秒（如图 3-7 中的坐标横轴所示），第 33978#TCP 流从 80 秒持续到 600 秒，第

33980#TCP 流从 122 秒持续到 600 秒。根据所有 TCP 流的重叠传输时间可知，虽然整个视频流的传输过程中相继开启了四条 TCP 流，但是每个时刻下参与传输的 TCP 流的数目最多只有两条。

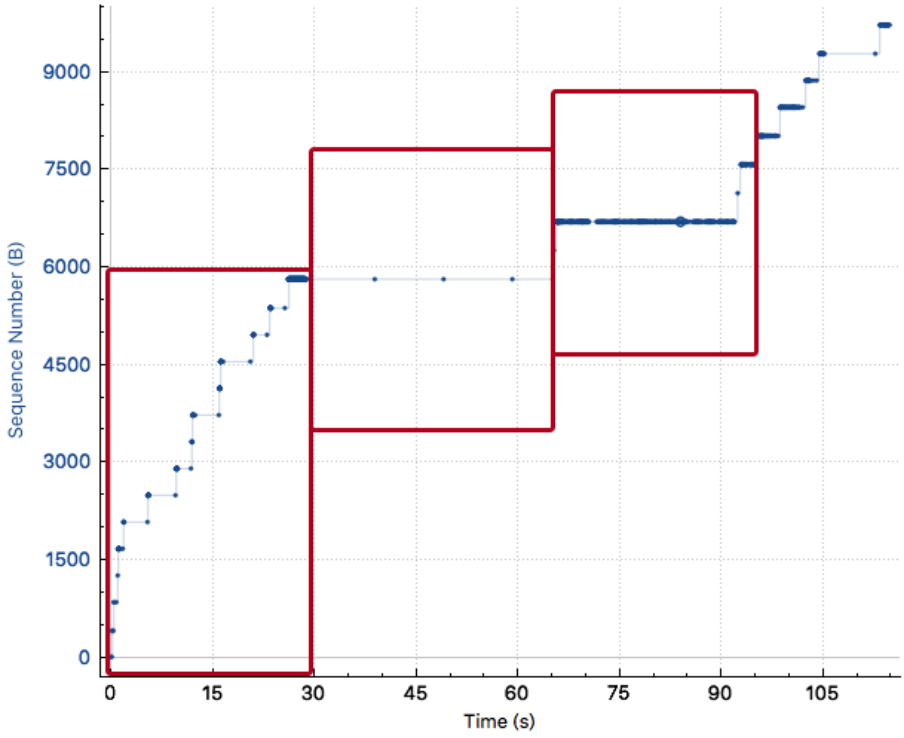


图 3-6　33970#TCP 流时间序列

Fig 3-6　33970#TCP stream time series

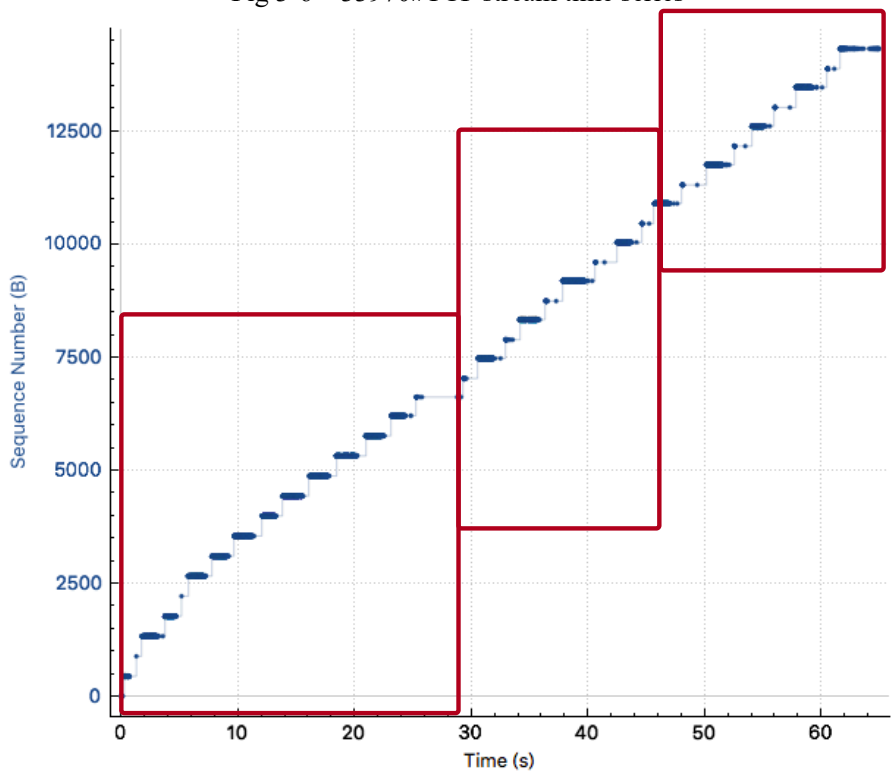
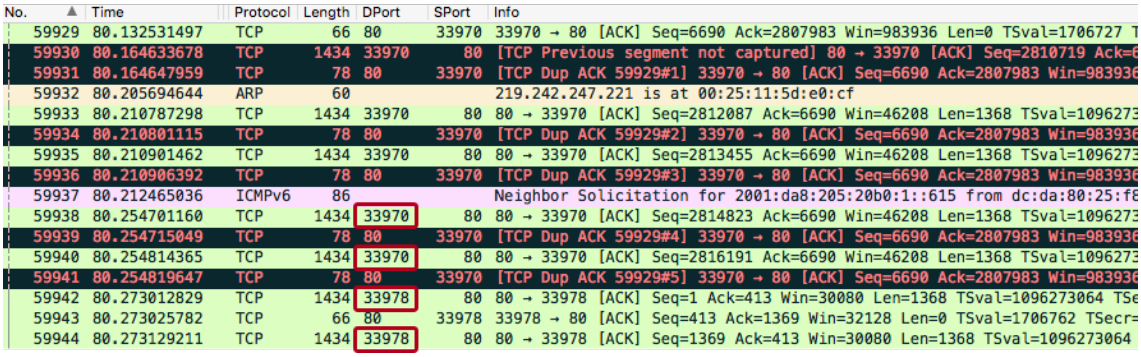


图 3-7　33972#TCP 流时间序列

Fig 3-7　33972#TCP stream time series

为了验证新 TCP 流开启的原因，我们从该 HAS 视频流的离线抓包文件中查看了 80s 处的数据报文，如图 3-8 所示。

图 3-8 中 59930#包中报文[TCP Previous segment not captured]显示网络中出现了丢包，表示此时没有收到 59929#之前的数据包，需要进行重新传输，之后的 59931#包到 59941#包都是客户端对 59929#包的重复应答，在第 5 次重复应答后，客户端最终选择了终止 33970#TCP 流的传输，并且在第 59942#包处开始建立 33978#TCP 流。由此可见，当网络条件变差出现持续丢包时，客户端会中断原有连接，通过转换端口新建一条 TCP 流来继续完成传输[60]。



No.	Time	Protocol	Length	DPort	SPort	Info
59929	80.132531497	TCP	66	80	33970	33970 → 80 [ACK] Seq=6690 Ack=2807983 Win=983936 Len=0 TSval=1706727 T
59930	80.164633678	TCP	1434	33970	80	[TCP Previous segment not captured] 80 → 33970 [ACK] Seq=2810719 Ack=
59931	80.164647959	TCP	78	80	33970	[TCP Dup ACK 59929#1] 33970 → 80 [ACK] Seq=6690 Ack=2807983 Win=98393
59932	80.205694644	ARP	60			219.242.247.221 is at 00:25:11:5d:e0:cf
59933	80.210787298	TCP	1434	33970	80	80 → 33970 [ACK] Seq=2812087 Ack=6690 Win=46208 Len=1368 TSval=109627
59934	80.210801115	TCP	78	80	33970	[TCP Dup ACK 59929#2] 33970 → 80 [ACK] Seq=6690 Ack=2807983 Win=98393
59935	80.210901462	TCP	1434	33970	80	80 → 33970 [ACK] Seq=2813455 Ack=6690 Win=46208 Len=1368 TSval=109627
59936	80.210906392	TCP	78	80	33970	[TCP Dup ACK 59929#3] 33970 → 80 [ACK] Seq=6690 Ack=2807983 Win=98393
59937	80.212465036	ICMPv6	86			Neighbor Solicitation for 2001:da8:205:20b0:1::615 from dc:da:80:25:f8
59938	80.254701160	TCP	1434	33970	80	80 → 33970 [ACK] Seq=2814823 Ack=6690 Win=46208 Len=1368 TSval=109627
59939	80.254715049	TCP	78	80	33970	[TCP Dup ACK 59929#4] 33970 → 80 [ACK] Seq=6690 Ack=2807983 Win=98393
59940	80.254814365	TCP	1434	33970	80	80 → 33970 [ACK] Seq=2816191 Ack=6690 Win=46208 Len=1368 TSval=109627
59941	80.254819647	TCP	78	80	33970	[TCP Dup ACK 59929#5] 33970 → 80 [ACK] Seq=6690 Ack=2807983 Win=98393
59942	80.273012829	TCP	1434	33978	80	80 → 33978 [ACK] Seq=1 Ack=413 Win=30080 Len=1368 TSval=1096273064 TSe
59943	80.273025782	TCP	66	80	33978	33978 → 80 [ACK] Seq=413 Ack=1369 Win=32128 Len=0 TSval=1706762 TSecr=
59944	80.273129211	TCP	1434	33978	80	80 → 33978 [ACK] Seq=1369 Ack=413 Win=30080 Len=1368 TSval=1096273064

图 3-8　HAS 视频流重传报文序列

Fig 3-8　Retransmission sequence of HAS video packets

综上所述，虽然 HAS 视频在传输过程中先后可能会创建多条 TCP 流，但是在同一段时间下，网络中最多只会存在两条 TCP 流并行传输。HAS 视频的“双流并传”特征，为我们后面对视频块序列进行分块类型校正提供了指导。

3.2.3　分块并传特征

HAS 视频流采用分块传输机制，每一段视频内容会被分成视频块和音频块交替传输。综合上一小节的“双流并传特征”，我们发现，同一段内容的视频块和音频块可以分别被放在两条 TCP 流中并行传输，并且任何一条 TCP 流均可以传输视频块或是音频块，并不存在现有研究中阐述的视频 TCP 流或是音频 TCP 流的区分，我们把这种现象叫做“分块并传特征”。图 3-9 和 3-10 是同一条 HAS 视频在不同时间段下请求方向的流量图。

综合每条请求报文的明文信息，我们发现，图 3-9 中，从 11s 到 30s 这段时间，33970#TCP 流（绿色区域）用于请求音频块（通过解析该条请求报文的明文信息获知），33972#TCP 流（橘色区域）用于请求视频块，两个 TCP 流在同一段时间下并行工作，完成媒体内容的请求；而图 3-10 中，在 30s-63s 这段时间，网络中只存

在 33972#TCP 流（橘色区域）同时对音频块和视频块进行串行请求，此时的 33970#TCP 流虽然没有参与数据传输但是也并未断开连接；在其后的 63s-122s 这段时间，33970#TCP 流又重新开始工作并转为请求视频块，与此同时，客户端建立了 33978#TCP 流与 33970#TCP 流配合工作，并行请求相应内容的音频块。

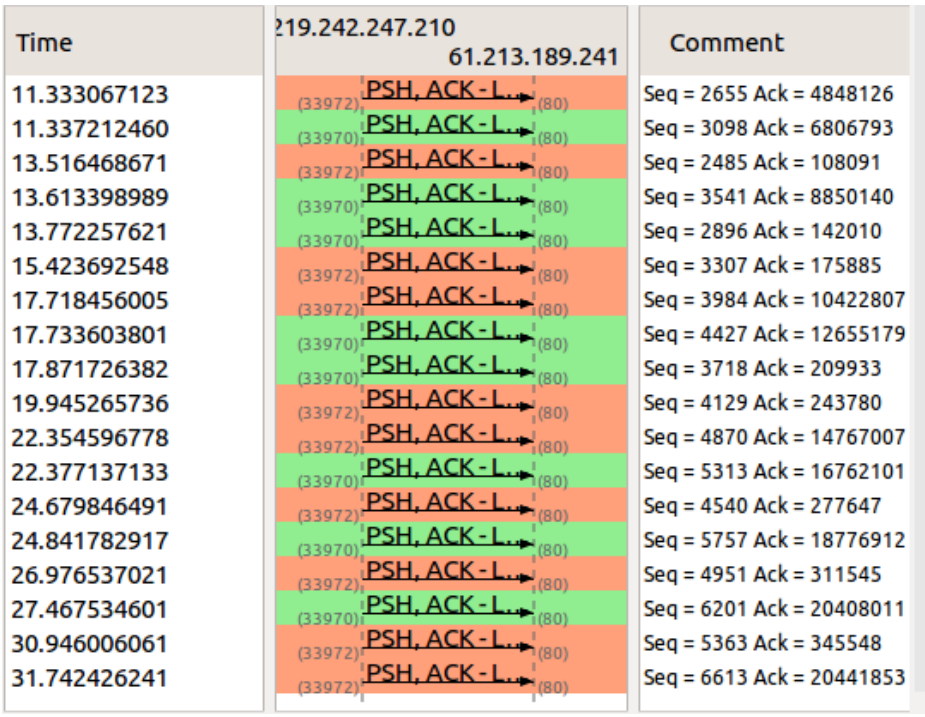


图 3-9　HAS 视频请求流量图一

Fig 3-9　HAS video request flow graph 1

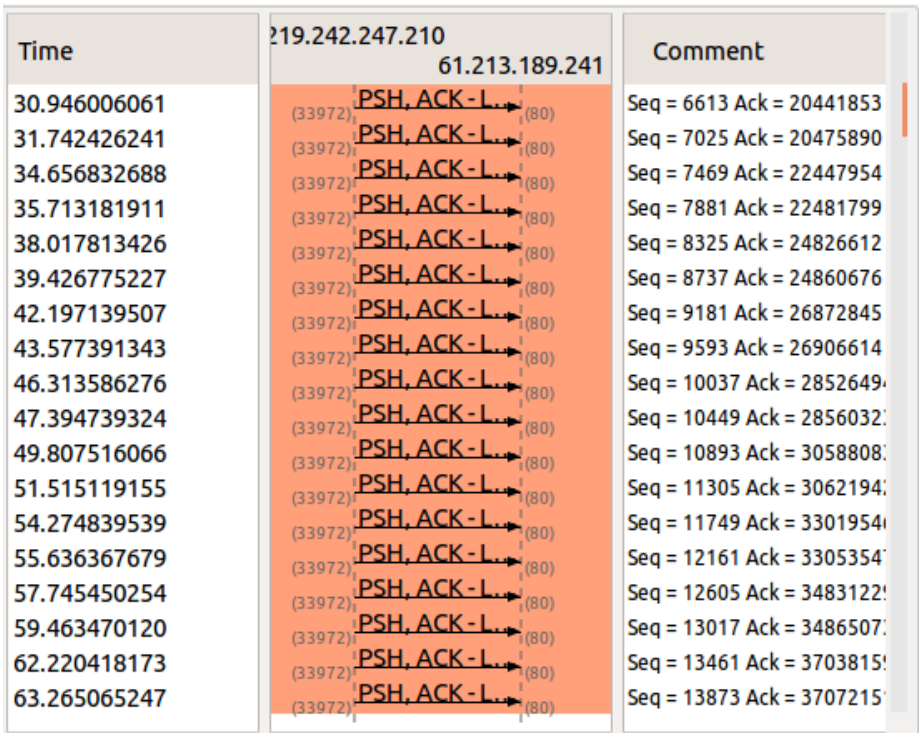


图 3-10　HAS 视频请求流量图二

Fig 3-10　HAS video request flow graph 2

同样地，我们还可以从 TCP 流的时序图中，发现视频流的“分块并传”特征。我们回看图 3-6 和 3-7。正常情况下，如果一条 TCP 流的传输速率稳定，并且不发生中断、丢包、大时延的情况，那么在时序图上看到的将是一条笔直上升的斜直线，斜线的斜率等于流量随时间变化的速率。33978#TCP 流和 33980#TCP 流在同一段时间内并行参与视频分块传输，用于交替请求视频块和音频块，由于两条流请求的视频块码率基本保持不变，所以在时序图上会呈现出总体变化趋势相同的两条斜直线。反观图 3-6 中的 33970 #TCP 流和图 3-7 中的 33972#TCP 流，结合这两条流的明文封包，前 30s，33970#TCP 流主要被用来请求音频块，33972#TCP 流被用来请求视频块，由于音频块和视频块的大小不同，所以用于传输两种数据块的时间也有显著差异，前 30s 中，33970#TCP 流和 33972#TCP 流分别呈现出的是一连串阶梯式上升的点和短横线。在 30s-60s 这段时间中，33970#TCP 流发生中断，33972#TCP 流被用来串行传输视频块和音频块，此时的时序图呈现出一连串短横线和点交错出现的阶梯序列。在 65s-90s 这段时间，33970#TCP 流转而开始请求高码率视频块，大体积的高码率视频块需要拆分成更多的 TCP 包进行传输，在网络中的传输时间也更长，所以这段时间的 33970#TCP 流时序图呈现的是一条水平的长横线。

由此可以看出，HAS 传输方式中，既会存在单条 TCP 流串行传输视频块和音频块，也会存在同时建立两条 TCP 流分别对视频块和音频块进行并行传输，并且每条 TCP 流对于分块的传输类别并不固定。多种多样的分块并传特性给予了 HAS 视频流灵活应对网络复杂条件的能力，但是却会给运营商从网络流量中精准识别每一个分块的传输边界和类型造成干扰，具体内容，我们会在后面章节中进行介绍。

3.2.4 同块同 ACK 特征

HAS 视频传输机制中，服务器端对视频内容按照“块”的粒度进行存储，客户端依照播放顺序每次向服务器请求一个视频块，但是因为 TCP 层最大传输单元的容量限制，服务器给客户端返回的视频块，在经过网络的时候会在 TCP 层被拆分成一个个大小相同的数据包进行传输。结合 HAS 视频流量的明文信息，我们观察到，来自于同一个 TCP 流下同一个视频块的网络封包的 ACK 号相同，这个特征在已有参考文献[14][15]和[59][60]中也有描述。我们把这种现象叫做 HAS 视频流的“同块同 ACK”特征。如图 3-11 所示。

图 3-11 是视频分块在网络中传输的行为特征。因为服务器端发送的响应报文是对客户端上一次请求报文的回应，回应的方法就是通过 ACK 号进行请求确认：ACK 号(响应报文) = Sequence 号(请求报文) + 包长(请求报文)。由于 TCP 层对数

据块进行拆分的原因，这些视频数据包实际上都是对同一个视频块请求信息的响应，所以理论上这些视频数据包都应该具有相同的 ACK 号。

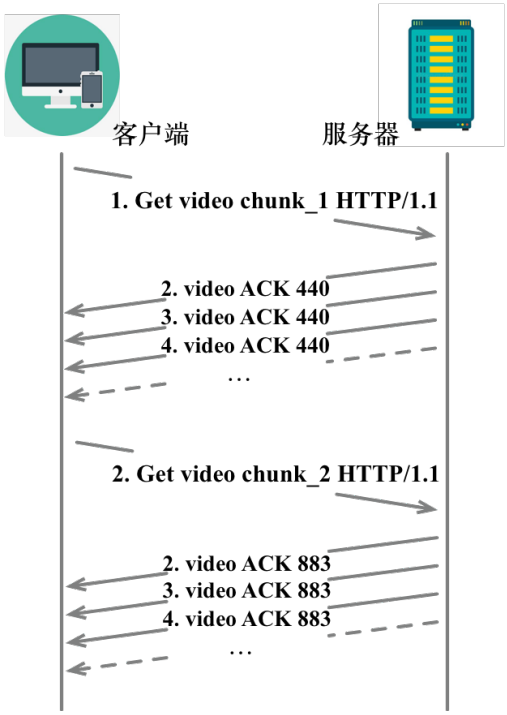


图 3-11 视频流网络传输行为[14,60]

Fig 3-11 Video stream network transmission behavior[14,60]

我们使用 HAS 视频在实际演播过程中捕获到的封包数据对这种特征进行了验证，如图 3-12 所示。

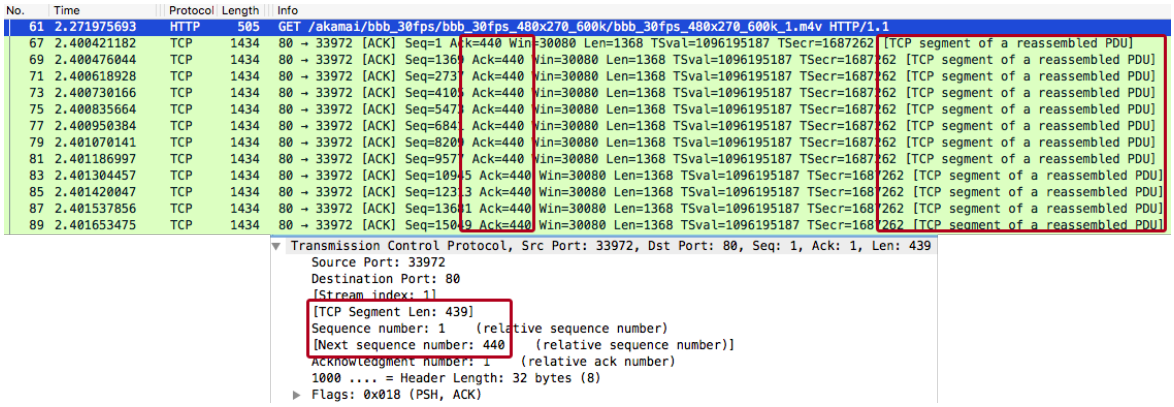


图 3-12 视频流报文序列

Fig 3-12 Video stream message sequence

图 3-12 的上图是 HAS 视频流量的明文信息，下图是上图中 61#包的解析信息。考虑到请求方向客户端的 TCP ACK 数据包不传输实质性的信息，我们通过“tcp.len !=0 and tcp.len!= 1”指令对请求方向的客户端 TCP ACK 包进行了过滤，所以，这部分的信息不在图 3-12 中显示。图 3-12 的上图中，61#包是客户端向服务

器发送的请求报文，从该包的 Info 列可知，该报文是 1#视频块的请求包。67#包-99#包是服务器对客户端的响应报文，可以看到这些响应报文的包信息末尾都注明了[TCP segment of a reassembled PDU]，表明这些响应报文是对同一个请求报文的响应包。下图的解析信息显示，61#包的 Sequence number = 1，TCP Segment Len = 439，而上图中 67#-99#包的 ACK 号均为 440，符合响应报文 ACK 号的计算方法。综上所述，67#-99#包是对 61#视频块请求封包的响应封包，而这些响应封包都具有相同的 ACK 号。同样的特征，还可以从 TCP 流的时序图中观察到，如图 3-13 和图 3-14 所示。

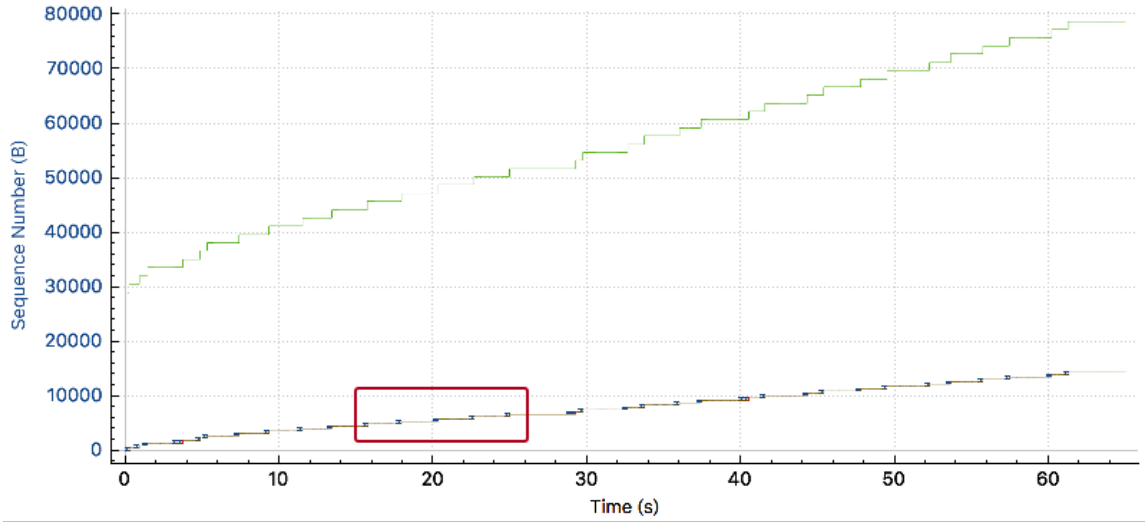


图 3-13　33972#TCP 流时序图一

Fig 3-13　33972#TCP stream time series 1

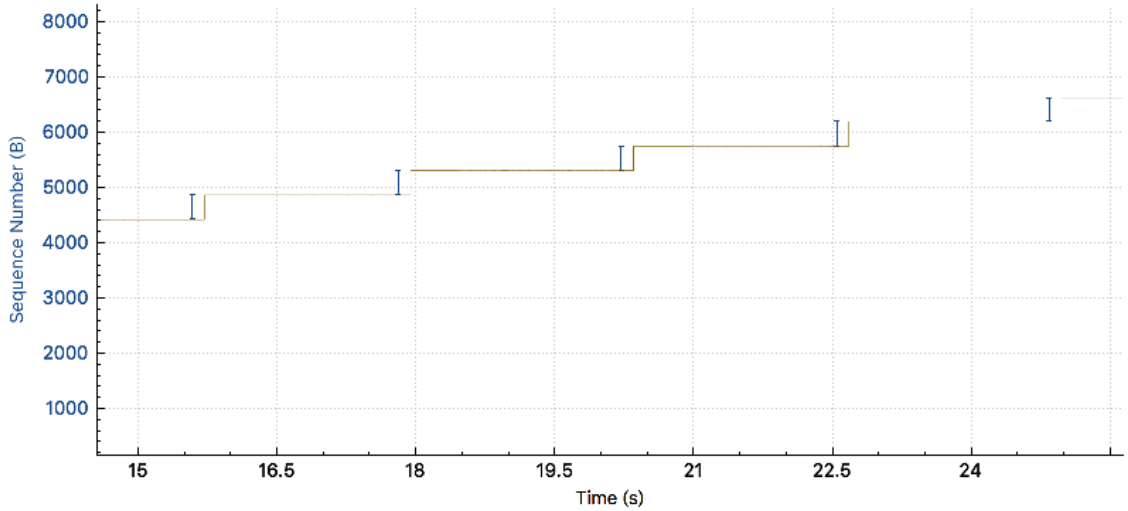


图 3-14　33972#TCP 流轨迹时序图二

Fig 3-14　33972#TCP stream time series 2

图 3-13 是 33972#TCP 流的轨迹时序图。图 3-13 中，上方浅色的线表征当前时刻的 TCP 窗口大小，下方浅色的线代表了当前包的 ACK 号，靠近下方浅色线的

一个个数据点代表当前时刻发送的数据包。图 3-14 是对图 3-13 中 15s-25s 这段时间的 ACK 号曲线进行放大的结果。可以看到，下图中的 ACK 曲线呈现阶梯式上升的趋势，客户端每发起一次视频块请求动作，都会在 TCP 流的轨迹序列中激发起一级“台阶”的产生，每级台阶上存在多个数据包，表明这些数据包的 ACK 号相同，由此可知，同一个视频块会被分成多个 ACK 号相同的数据包进行传输。

此外，我们还可以看到，图 3-13 中的 TCP 窗口线整体呈现出阶梯式上升的趋势，但是每级台阶的宽度不尽相同，这是因为同一个 TCP 流可以同时用来传输视频块和音频块，而视频块与音频块的码率不同，并且即便都是视频块，码率不同的视频块的大小也会表现出显著的差异，导致每种块的流量（包含的字节数）不同，因此用于传输该块的数据报的数量就会不同，于是就会在 TCP 流的轨迹序列中呈现出“阶梯式上升、台阶高度相同，台阶宽窄不一”的特点，这种特点也从侧面印证了 HAS 视频流码率的动态传输特征和上节中我们发现的“分块并传特征”。

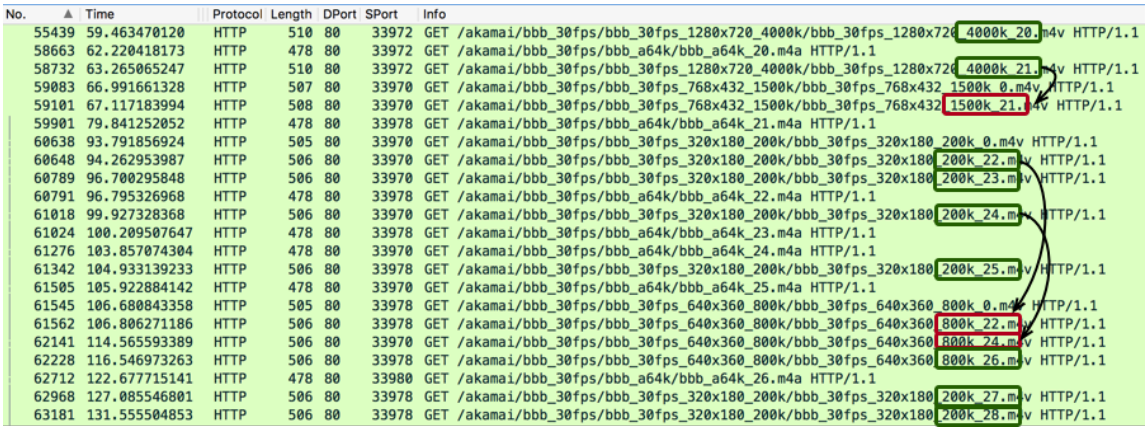
由此可以看出，HAS 传输方式中，网络对于同一个 TCP 流下同一个视频块的传输会拆分成连续多个 ACK 号相同的数据包进行。后面章节中，我们会根据这个重要的特征，对视频流进行分块边界识别和流量聚合，将属于同一个视频分块的数据报文识别出来，对视频流基于“块”粒度进行序列重建。

3.2.5 分块重传特征

HAS 视频服务过程中，客户端会根据当前的网络状况选择视频块的请求码率，并将这些视频块提前存储在缓冲区中，用于在未来时刻提取出来进行播放[58]。但是，由于网络的波动性，根据过去网络情况请求的最优视频码率，不一定在当前的网络条件下也是最优的。

我们观察到，HAS 传输机制在实际应用时存在“后悔药”的操作，即支持视频块的重新传输。比如，本来第三个视频块选择传输的是 200kbps 的视频块，由于当前网络性能的提升，客户端有条件对相同视频内容的视频块进行更高码率的储备，于是又重新请求传输了第三个视频块的 600kbps 码率版本，最终客户端实际播放的是 600kbps 的三号视频块，并将缓存中 200kbps 的三号视频块直接丢弃掉，在这个过程中，虽然用户最终在客户端中只看到了高码率的三号视频分块，但是网络侧的确是先后传输过相同内容的视频块的两个码率版本[60]。这种现象称为 HAS 视频流的“分块重传”特征。

图 3-15 是 HAS 视频流请求报文的明文信息。客户端按照自适应码率请求逻辑依据视频分块的先后顺序对视频块进行请求下载。



No.	Time	Protocol	Length	DPort	SPort	Info
55439	59.463470120	HTTP	510	80	33972	GET /akamai/bbb_30fps/bbb_30fps_1280x720_4000k/bbb_30fps_1280x720_4000k_20.m4v HTTP/1.1
58663	62.220418173	HTTP	478	80	33972	GET /akamai/bbb_30fps/bbb_a64k/bbb_a64k_20.m4a HTTP/1.1
58732	63.265065247	HTTP	510	80	33972	GET /akamai/bbb_30fps/bbb_30fps_1280x720_4000k/bbb_30fps_1280x720_4000k_21.m4v HTTP/1.1
59083	66.991661328	HTTP	507	80	33970	GET /akamai/bbb_30fps/bbb_30fps_768x432_1500k/bbb_30fps_768x432_1500k_0.m4v HTTP/1.1
59101	67.117183994	HTTP	508	80	33970	GET /akamai/bbb_30fps/bbb_30fps_768x432_1500k/bbb_30fps_768x432_1500k_21.m4v HTTP/1.1
59901	79.841252052	HTTP	478	80	33978	GET /akamai/bbb_30fps/bbb_a64k/bbb_a64k_21.m4a HTTP/1.1
60638	93.791856924	HTTP	505	80	33970	GET /akamai/bbb_30fps/bbb_30fps_320x180_200k/bbb_30fps_320x180_200k_0.m4v HTTP/1.1
60648	94.262953987	HTTP	506	80	33970	GET /akamai/bbb_30fps/bbb_30fps_320x180_200k/bbb_30fps_320x180_200k_22.m4v HTTP/1.1
60789	96.700295848	HTTP	506	80	33970	GET /akamai/bbb_30fps/bbb_30fps_320x180_200k/bbb_30fps_320x180_200k_23.m4v HTTP/1.1
60791	96.795326968	HTTP	478	80	33978	GET /akamai/bbb_30fps/bbb_a64k/bbb_a64k_22.m4a HTTP/1.1
61018	99.927328368	HTTP	506	80	33970	GET /akamai/bbb_30fps/bbb_30fps_320x180_200k/bbb_30fps_320x180_200k_24.m4v HTTP/1.1
61024	100.209507647	HTTP	478	80	33978	GET /akamai/bbb_30fps/bbb_a64k/bbb_a64k_23.m4a HTTP/1.1
61276	103.857074304	HTTP	478	80	33970	GET /akamai/bbb_30fps/bbb_a64k/bbb_a64k_24.m4a HTTP/1.1
61342	104.933139233	HTTP	506	80	33978	GET /akamai/bbb_30fps/bbb_30fps_320x180_200k/bbb_30fps_320x180_200k_25.m4v HTTP/1.1
61505	105.922884142	HTTP	478	80	33970	GET /akamai/bbb_30fps/bbb_a64k/bbb_a64k_25.m4a HTTP/1.1
61545	106.680843358	HTTP	505	80	33978	GET /akamai/bbb_30fps/bbb_30fps_640x360_800k/bbb_30fps_640x360_800k_0.m4v HTTP/1.1
61562	106.806271186	HTTP	506	80	33978	GET /akamai/bbb_30fps/bbb_30fps_640x360_800k/bbb_30fps_640x360_800k_22.m4v HTTP/1.1
62141	114.565593389	HTTP	506	80	33970	GET /akamai/bbb_30fps/bbb_30fps_640x360_800k/bbb_30fps_640x360_800k_24.m4v HTTP/1.1
62228	116.546973263	HTTP	506	80	33978	GET /akamai/bbb_30fps/bbb_30fps_640x360_800k/bbb_30fps_640x360_800k_26.m4v HTTP/1.1
62712	122.677715141	HTTP	478	80	33980	GET /akamai/bbb_30fps/bbb_a64k/bbb_a64k_26.m4a HTTP/1.1
62968	127.085546801	HTTP	506	80	33978	GET /akamai/bbb_30fps/bbb_30fps_320x180_200k/bbb_30fps_320x180_200k_27.m4v HTTP/1.1
63181	131.555504853	HTTP	506	80	33978	GET /akamai/bbb_30fps/bbb_30fps_320x180_200k/bbb_30fps_320x180_200k_28.m4v HTTP/1.1

图 3-15　HAS 视频流请求报文

Fig 3-15　HAS video stream request message

如图 3-15 所示，客户端先是在 58732#包处请求了 4000kbps 的 21 号视频块，随后由于网络条件恶化，下载高码率的 21 号视频块太过费时会带来卡顿隐患，于是在 59101#包处重新请求了一次 1500kbps 的 21 号视频块。同理，客户端也分别在 60648#包和 61010#包处启动了 22 号视频块和 24 号视频块的 200kbps 码率版本请求，之后由于网络条件提升，又分别在 61562#包和 62141#包处重新请求了码率为 800kbps 的 22 号视频块和 24 号视频块，而在这些重新请求动作结束后，缓冲区中已经下载了的高码率的 21 号视频块和低码率的 22 号、24 号视频块均会被客户端丢弃掉。

由此可以看出，由于网络的波动，HAS 视频流的动态码率调节机制会适时的在视频流量中表现出“分块重传”特征。

3.3　视频块序列重建算法

本节，我们基于发现的特征，对 HAS 视频的流量基于“块”粒度进行时序重建，主要包含分块边界识别、流量聚合、区分视频块和音频块。

3.3.1　算法实现

基于上节中介绍的“同块同 ACK”特征，我们可以根据 ACK 号对流量进行分块聚合，但是在实际的应用的过程中仍然会遇到一些干扰：

(1) 视频块动辄会被拆解为成百上千个 TCP 包传输，由于网络的不稳定性，传输时经常会出现大量的丢包、重传，对基于 ACK 号进行流量聚合造成干扰[60]。

(2) 根据“双流并传特征”，HAS 视频流在传输过程中会相继开启多条 TCP 流，用于在网络条件变差的情况下，通过切换 TCP 流接替先前数据的传输工作。但是

由于每条 TCP 流的 ACK 号是单独计算的，因此，每次 TCP 流的切换，都会在时序上打乱掉 ACK 的特征。

(3) 根据“分块并传特征”，同一时间下，音频块和视频块会采用不同的 TCP 流分开传输，导致同一段时间内，两种或是多种不同类型分块不同 ACK 号的 TCP 包夹杂在一起，影响到流量聚合后分块类型的判断。

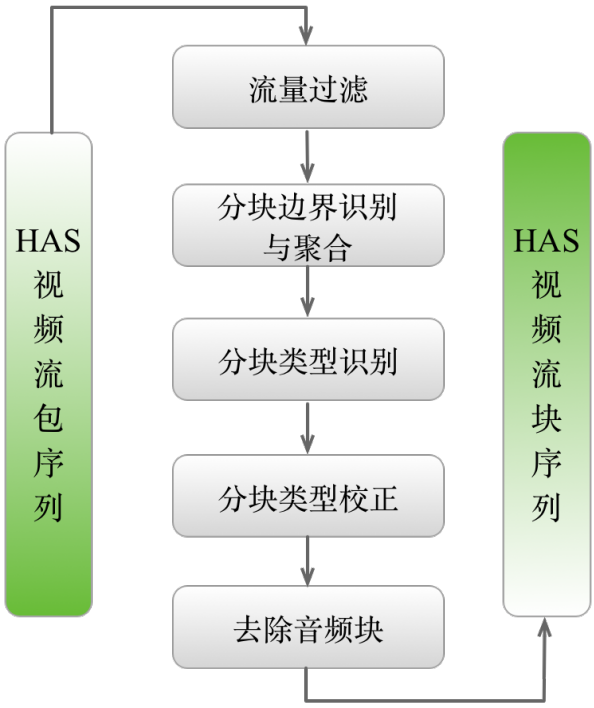


图 3-16 视频流块序列重建流程
Fig 3-16 Video stream chunk series reconstruction process

考虑到视频流在实际传输过程中可能会遇到的封包乱序、TCP 流切换等问题，我们以“ACK 号作为分块 ID 进行流量聚合”主线规则的同时，引入了多条支线规则，用于在对某个分块的识别出现分歧时，辅助判断该分块的传输边界和类别。

具体地，我们提出“视频块序列重建算法”，其基本思想是，首先对 HAS 视频流的包序列进行流量过滤去除噪声数据，然后基于主线规则“同块同 ACK 号”识别分块的边界并进行流量聚合，同时计算出音频块的数据量范围，对块序列中的分块类型进行区分，并基于支线规则校正分块的类型，最后从块序列中去除音频块，得到从加密流中重建的 HAS 视频块序列。算法逻辑流程图如图 3-16 所示：

(1) 视频流量过滤。首先，根据各个 IP 对之间流量在封包总流量中的占比和流量方向，识别视频流的客户端 IP 和服务器 IP，滤除背景流数据。然后基于“IP+端口”对视频流量进行统计，识别客户端开启的端口号，筛选出主要 TCP 流。并且根据 IP 对之间视频流的传输方向，将视频流区分为请求流和下载流。

(2) 分块边界识别与聚合。将 ACK 号作为块 ID 对流量进行聚合，但是考虑到 HAS 视频传输过程中会开启多条 TCP 流传输视频分块，每条 TCP 流的 ACK 号单独计算可能会扰乱掉 ACK 号的总体时序特征。比如，客户端先后开启了 1#TCP 流和 2#TCP 流传输 100#视频块和 101#视频块，由于每条 TCP 流的 ACK 号是单独计算的，流量中 101#视频块的 ACK 号有可能就会小于 100#视频块，此时如果仅仅根据 ACK 号对流量进行聚合，重建的块序列中 101#视频块就会出现在 100#的前面，打乱视频块序列的先后顺序。所以，我们首先是对所有的封包按照到达时间进行排列，然后再基于 ACK 号进行流量聚合，并且将每一个块的第一个封包的到达时间作为这个块的开始请求时间。这样做，可以有效的避免多 TCP 流传输时可能会造成的块序列乱序的问题。

(3) 分块类型识别。从流量中重建的块序列，还需要对其进行类型判断，区分视频块和音频块。HAS 视频的动态码率机制使得每一个视频分块都会存在多种码率等级，不同码率等级视频块的数据大小会有显著不同。但是，同一条 HAS 视频的所有音频块只会存在一种码率，并且由于编码内容的不同，同一条视频中，即便是最低码率的视频块也要比音频块大得多。因此，我们通过对视频流的块序列按照块流量大小进行数量排序，根据块大小的众数确定视频块和音频块的分类阈值γ，用于对分块类型进行判断。关于分类阈值γ，将在下一小节讨论。

(4) 分块类型校正。在网络条件差的情况下，客户端会通过切换 TCP 流接替当前视频块的传输。在这种情况下，仅仅通过 ACK 号、分块大小对流量进行聚合和类型判断，那些断开续传的分块很有可能就会落入到音频分块的长度识别范围中[60]，造成分块类型判断错误。

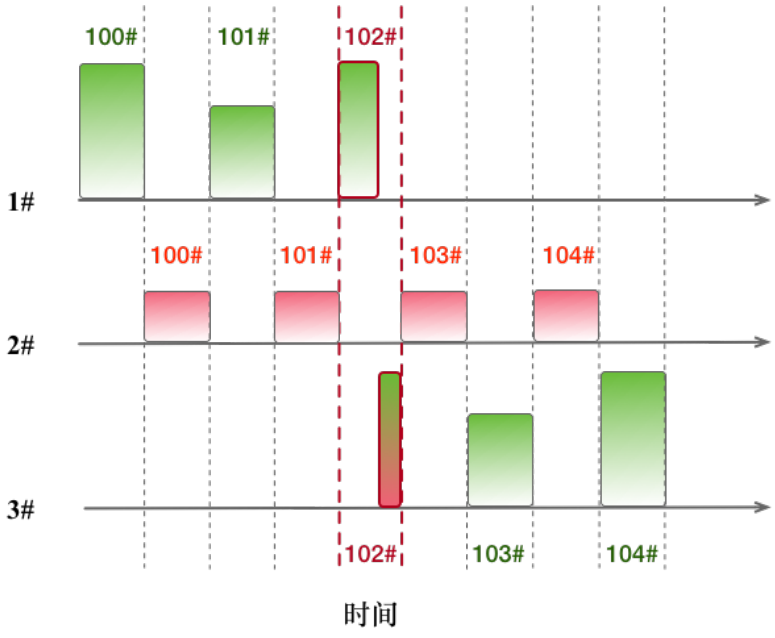


图 3-17 分块续传示意图

Fig 3-17 Chunk retransmisson diagram

图 3-17 是分块续传示意图，其中横向坐标轴是在时间上严格对齐的 3 条 TCP 流（1#、2#、3#）的时间轴，分别称之为 1#、2#、3#坐标轴；等间隔切分时间轴的虚线表现每个分块中包含的内容持续时长（HAS 视频中每个分块的内容持续时长相同）；1#和 3#坐标轴上大小不一的绿色方块代表视频分块（例如 100#、101#视频块），表示当前时刻该条 TCP 流被用于传输视频块，2#坐标轴上大小相同的粉色方块代表音频分块（例如 103#、104#音频块），表示当前时刻该条 TCP 流被用于传输音频块；图中颜色交叠渐变的 102#分块表示该分块的媒体类型不确定。

如图 3-17 所示，客户端先同时开启了 1#和 2#TCP 流分别进行视频块和音频块的传输，当 1#TCP 流在传输 102#视频块的时候，网络条件开始恶化，客户端切换到 3#TCP 流接替传输 102#视频块并同时更新了 ACK 号，此时 3#TCP 流续传的 102#视频分块根据块大小被误判为 102#音频块，并导致 2#TCP 流中正常传输的音频块按照顺序被误判为 103#音频块，将 3#TCP 流中继续传输的分块识别为 103#视频块，产生了错误的分块类型判断。

3.2 节中，我们观察到，由于 HAS 视频流的“双流并传”特征和“分块并传”特征，同一段时间下，网络中最多只会存在两条 TCP 流分别请求视频块和音频块，并且不会连续进行两次音频块的请求。因此，当网络中出现多条 TCP 流时，我们可以根据同一条流前后的分块判断类型和开始传输时间进行类型校正。比如，在重建后的块序列中，连续出现了 102#和 103#两个音频块，说明此处存在误判块，而在对分块类型进行初判后的结果中，2#TCP 流在这段时间传输的都是音频块，3#TCP 流在之后的一段时间传输的都是视频块，并且 103#音频块的开始传输时间在 103#视频块之前，说明在 102#音频块发生了误判，于是将这部分分块向前归并给 102#视频块，并对整条块序列重新进行序号排序，依此类推，对整条块序列的分块类型完成校正。

(5) 去除音频块。对分块类型校正的块序列进行音频块剔除，获得只包含视频分块的视频块序列。

3.3.2 实验和结果分析

为了从加密流量中区分出音频块和视频块，当一条视频流通过“同块同 ACK 号”的特征进行初步的块粒度重建后，我们需要对每个分块的流量（包含的字节数）进行统计分析，获得视频块和音频块的分类阈值γ。

HAS 视频流中视频块采用多码率进行编码，编码范围从几百 kbps 到几万 kbps 不等；而音频块采用单一码率进行编码，编码范围通常在 200kbps 以下。并且由于编码内容复杂度的不同，同一条视频中，即便是最低码率的视频块的流量也要比音

频块大得多。同时，虽然同一条视频中音频块和视频块总数量相同，但是因为编码策略的不同（客户端在视频演播过程中只能请求一种码率的音频块，却可以通过自适应码率调节，请求多种不同码率的视频块），使得不同码率下音频块和视频块的数目存在显著的差异，例如一次视频演播过程中 65Kbps 的音频块出现了上百次，但是 1.5Mbps 的视频块却可能只出现过 1 次，反映在块序列的统计结果中，音频块会在某个块流量范围下集中分布，而视频块则会零散地分布在不同的块流量下，并且流量跨度的范围也要大得多，如图 3-18 所示。

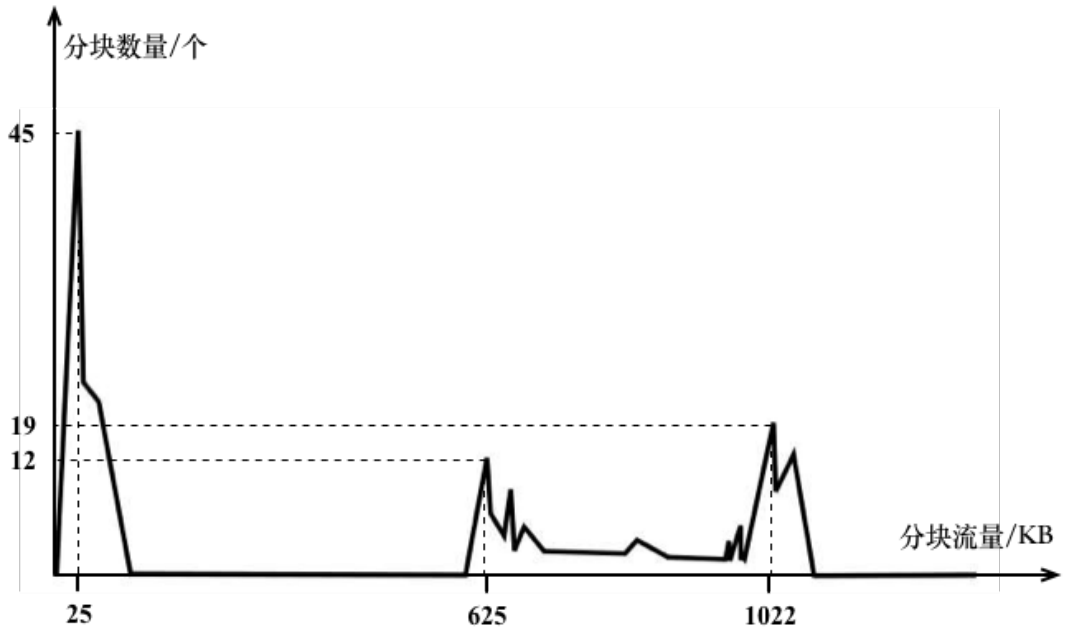


图 3-18 分块数量分布

Fig 3-18 Distribution of the number of chunks

图 3-18 是一条视频流在块序列重建后，对每个流量（字节数大小）下的分块数目进行统计的结果。图 3-18 中显示的这条视频流的源视频文件包含两种视频码率：2500kbps、4000kbps；一种音频码率：96kbps，由图可见，音频分块集中地分布在 30KB 左右的流量范围内，并且在 25KB 处的音频分块最多，图中表现为一个高耸的尖峰；而视频分块零散的分布两个尖峰附近：625KB 和 1022KB，此处可对应该视频文件中存在的两种视频码率。从图 3-18 中，我们还可以观察到，由于每个分块内容的不同，同一码率分块的实际块流量会存在一定的差异（例如，即便是相同码率的音频块也会存在一定的流量波动，并不都是同一种块流量），但是，即便是最低码率的视频分块也要比音频块的流量大得多。

综上所述，为了在区分音频块和视频块的同时，规避同一码率分块存在的流量波动问题，我们在每条视频块序列的统计结果中，在数量最多的分块流量数值的基础上扩大一倍，作为视频块与音频块的分类阈值γ，以图 3-18 为例，这条视频流的

分块类型分类阈值$\gamma = 50\text{KB}$。表 3-1 中罗列了我们观测的视频文件的类型和相关参数。

表 3-1 视频文件信息
Table 3-1 Video file information

文件类型	视频码率范围	音频码率	持续时长
科幻片 1	4 种码率(150kbps-750kbps)	72kbps	10min
科幻片 2	5 种码率(386kbps-2773kbps)	131kbps	12min
动画片	10 种码率(254kbps-14931kbps)	65kbps	10min
纪录片	2 种码率(3000kbps-4000kbps)	128kbps	5min
运动集锦	2 种码率(2500kbps-4000kbps)	96kbps	6min
动态写真	6 种码率(2859kbps-19683kbps)	194kbps	3min
电视屏保	1 种码率(1144kbps)	191kbps	5min

我们对 36 条类型不同、内容不同、码率范围不同的视频文件的流量序列，根据我们提出的算法进行了块序列重建，并将重建的块序列与从客户端抓取到的视频块请求序列进行比较。比较结果如图 3-19 所示。

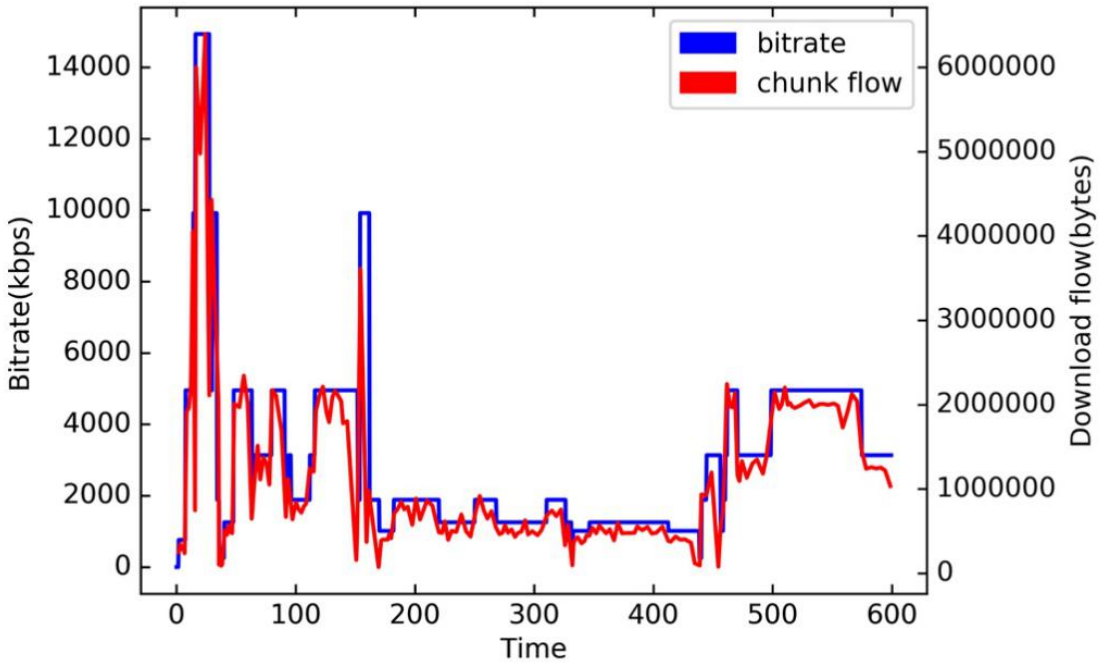


图 3-19 视频块序列与码率请求序列
Fig 3-19 Video chunk series and bit rate request sequence

图 3-19 是同一次视频演播过程中，从网络流量中重建的视频块序列与客户端

的视频块请求序列。红色实线是使用我们算法从加密视频流中重建的视频块序列，蓝色实线是这条视频流演播时从客户端抓取到的视频块请求序列。我们使用度量指标 RMSE（Root Mean Squared Error，均方根误差），比较每条视频块序列与客户端视频块请求序列的相似度，两条序列中所有匹配样本点的平均距离 RMSE 不高于 0.132，这表明通过我们的算法从加密流量中重建的视频块序列，很好地拟合了客户端视频块请求序列的变化趋势。

3.4 本章小结

为了从加密视频流量中重建视频块序列，本章基于网络测量的方法对加密视频流的传输特性进行分析，并根据分析发现提出了一种基于 HAS 视频流的块序列重建算法。

首先，从本章研究问题的意义、挑战和研究思路的角度，介绍了从网络流量中重建“视频块序列”对于运营商感知用户 QoE 的必要性，以及在重建过程中可能会遇到的三个主要问题：确定分块传输边界、区分音频块和视频块、网络流量加密。并指出通过挖掘 HAS 技术在实际应用时的视频传输特性，可以为从网络流量中重建视频块序列提供指导。

然后，介绍了我们基于网络测量对加密视频流传输特性进行分析的发现。详述了 HAS 技术在实际应用中的四种流量传输特性：双流并传、分块并传、同块同 ACK 号、分块重传。

紧接着，介绍了本文提出的视频块序列重建算法。从可能会遇到的干扰出发：封包乱序、TCP 流切换等，在制定了以“ACK 号为分块 ID 进行流量聚合”作为主线规则的同时，引入了多条支线规则，用于在对某个分块的识别出现分歧时，辅助判断该分块的传输边界和类别。算法主要包含 5 个步骤：视频流量过滤、分块边界识别与聚合、分块类型识别、分块类型矫正、去除音频块。

最后，我们搭建了实验数据采集和测试环境，评估了“视频块序列重建算法”的效果。实验结果表明，使用我们算法从网络流量中重建的视频块序列与客户端的块请求序列之间的样本均方根误差不高于 0.132，这表明重建的视频块序列很好的拟合了客户端的块请求序列的变化趋势，证明了算法的有效性。

4　基于加密流量的 HAS 视频用户 QoE 感知

在上一章中，我们重构出以“块”为颗粒度的视频流量序列，本章将对此视频流的“块”（时间）序列进行挖掘，目标是推断视频用户的观看体验。如前所述，用户的观看体验可以采用（应用层）客观的 QoE 指标，例如播放码率、卡顿次数、卡顿时间长度、帧率等来度量。考虑到用户在不同的播放状态下会有不同的视频观看体验[4]。在本章中，我们定义了一种新的客观 QoE 指标，称为“缓冲区综合状态”，用来度量用户 QoE，研究如何根据在网络侧观察到的视频块序列来推断缓冲区综合状态，进而评估用户 QoE。

4.1　问题描述

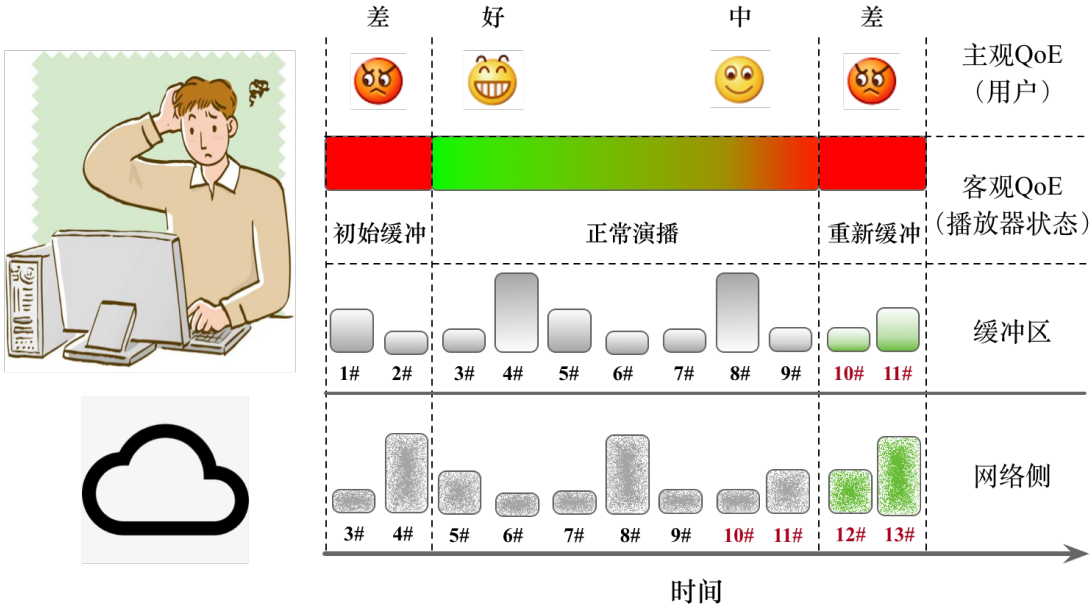


图 4-1　网络视频 QoE 评估

Fig 4-1　Network video QoE assessment

图 4-1 是从运营商的视角评估网络视频用户 QoE 的简要模型。QoE 是主观感受，虽然它可以从用户的感受中反映视频流的服务质量（如图 4-1），但是，它难以捕捉，故而，现有的研究中往往采用客观的 QoE 参数表征用户的主观 QoE，其中又以应用层播放器的演播指标最为常用，例如延迟、卡顿次数、码率切换幅度、帧率等。这些应用层客观 QoE 指标的变化情况，反映出播放器处于不同的演播状态：初始缓冲、正常演播、重新缓冲（卡顿）等（如图 4-1“播放器状态”）。为了抵抗网络抖动、避免被动丢帧导致画面花屏或卡顿，HAS 视频客户端中设置有

缓冲区，支持客户端根据当前的网络状况动态地选择下一个视频块的请求码率，并将这些视频块提前存储在缓冲区中，用于在未来时刻提取出来进行播放[58]（缓冲区中已经被播放的视频块会被丢弃掉，如图 4-1 中 1#至 9#灰色视频块；未播放的视频块需要在缓冲区中充盈到一定数量，播放器方可重新启动播放，如图 4-1 中第 10#和 11#绿色视频块），所以说，缓冲区的状态，会直接影响播放器的演播状态。而运营商，可以在网络中捕获视频流量，并通过对已经拆分成数据报文的视频流量基于“块粒度”重建得到视频块序列（如图 4-1 中描绘：网络中的每个视频块是由许多个点状的数据报文组合成的），用以估计客户端缓冲区的视频块请求和填充状态，即缓冲区的状态，进而感知播放器的演播状态。此外，HAS 客户端的“提前请求、预先缓存”的视频块请求机制，又会导致客户端当前请求的视频块与用户观看的视频块存在时差，如图 4-1 中最下层，运营商当前在网络中已经先后“看到”了第 12#和 13#视频块正在请求和下载，但是这两个块却需要等到完全下载到缓冲区中之后，客户端才可以在未来时刻提取出来播放给用户观看。换句话说，用户当前时刻的 QoE 受到网络中过去下载的视频块质量的影响，所以，运营商若想要感知用户在当前时刻的 QoE，需要“回头看”其对于视频流量的历史捕获情况。

已有的研究表明，用户处于不同的播放状态下会有不同的视频体验[4]。而在上一章中，我们提出了基于 HAS 视频流的块序列重建算法，可以从加密视频流中合理有效地重建视频块序列。本章，我们对此视频流的“块”时间序列进行挖掘，目的是通过网络流量的块序列推断播放器的演播状态，进而感知视频用户的 QoE。在研究过程中，我们不仅根据 HAS 客户端的演播特征，提出了一种更合理的应用层客观 QoE 指标，并且使用了适用于时间序列的模型和训练技巧，同时，在算法实现时，考虑到了运营商看到的视频数据与当前用户看到的视频数据存在“时间差”的问题。

4.2 新的客观 QoE 指标

我们的目标是只使用视频传输过程中网络侧流量数据的统计信息，实时预测客户端的视频播放状态，从而实现对视频用户 QoE 的跨层感知。我们将网络侧的流量信息称为视频流 QoS 参数X，将客户端视频播放状态定义为标度Y。因为视频流数据具有先后顺序排列的时间特征，是典型的时间序列，可以使用时序数据挖掘的相关方法进行分析。因此，我们的研究目标可以概括为：使用时间序列数据挖掘的方法，以X为特征参数，Y为目标，寻找X与Y之间的映射模型M: $X \rightarrow Y$，使得当知道网络中当前时刻的X时，可以通过M实时预测下一时刻的Y，即实时预测客户端的视频处于何种播放状态。

下面，我们主要介绍本章研究内容中使用的客观 QoE 指标。

在众多影响用户观看体验的因素中，“卡顿”是最重要的客观 QoE 指标之一，现有的工作大都是采用“卡顿”来评估用户 QoE。“卡顿”的通常定义为：播放器缓冲区占有量低于某个门限（一个极端的门限值= 0）时，视频演播中断，因此，现有的相关研究中大都通过观察缓冲区占有量来评估视频用户 QoE，缓冲区占有量也就自然而然成为视频用户 QoE 评估中经常选择的客观指标。但是，我们发现，在视频演播过程中，缓冲区占有量会经历上升-下降的周期性振荡，由于缓冲区占有量升降变化的“惯性”，即便是具有相同的占有量，当缓冲区处于不同的变化过程中，即处于上升或是下降过程，带来的视频观看体验是不同的。我们综合缓冲区的占有量和它的变化趋势，提出了一种全新的应用层客观 QoE 度量指标：“缓冲区综合状态“，用以表示不同的缓冲区状态。图 4-2 描述了在正常演播过程中缓冲区占有量变化的一般模型。

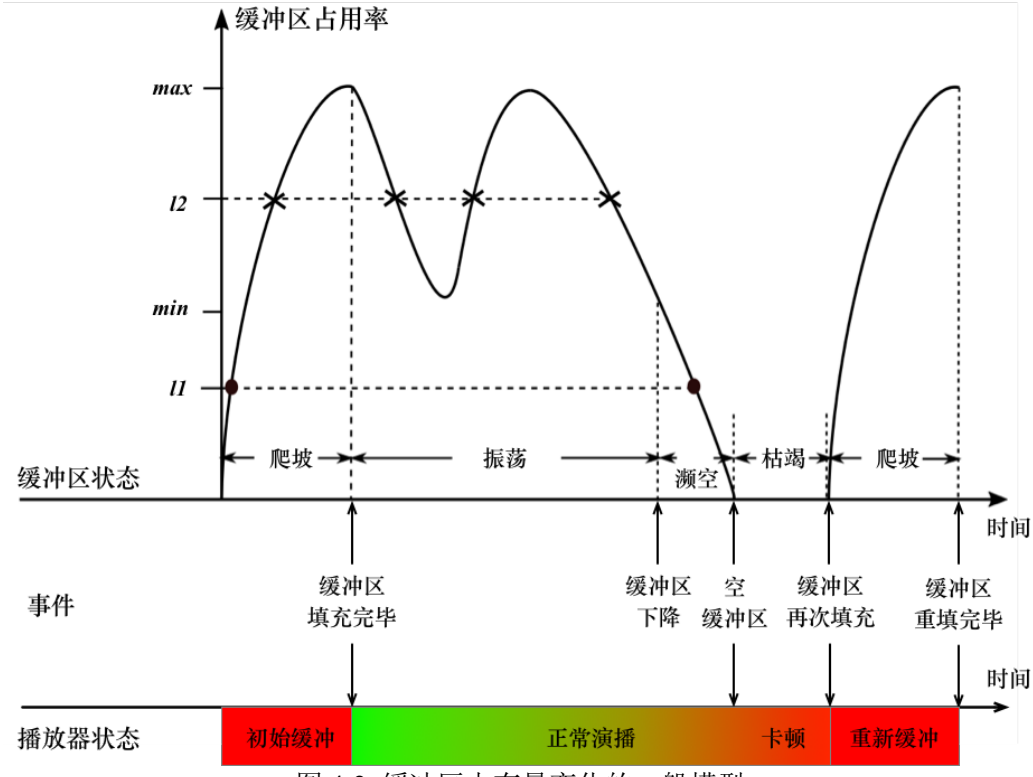


图 4-2 缓冲区占有量变化的一般模型

Fig 4-2 General model of buffer occupancy

图 4-2 的靠上部分刻画了一次完整的视频演播过程中，缓冲区可能会经历的四种状态：爬坡、振荡、濒空、枯竭。在爬坡状态下，缓冲区占有量从零开始增长一直到最大阈值max；在振荡状态下，缓冲区占有量在最大阈值max和最小阈值min之间来回振荡；在濒空状态下，缓冲区占有量虽然仍然高于零，但是已经低于最小

阈值min并且仍然保持减少的趋势；在枯竭状态下，缓冲区占有量保持为零。图 4-2 的靠下部分显示了与缓冲区状态一一对应的播放器状态，图 4-2 的中部罗列了导致缓冲区和播放器各自状态之间切换的激活“事件”，例如，在“空缓冲区”这个事件发生前后，缓冲区占有量逐渐减小到零进入“枯竭”状态，对应的，此时播放器也进入了“卡顿”状态。

我们之所以使用“缓冲区综合状态”对应播放器的播放状态，是因为我们发现播放器的播放状态和用户的观看体验不仅仅与缓冲区的占有量有关，还与缓冲区占有量的变化方向有关。

如图 4-2 所示，缓冲区占有量相同时，对应的缓冲区状态和播放器状态却有可能是不同的，此时用户的视频观看体验也是截然不同的。例如，当占有量为$l1$时，缓冲区可以处于“爬坡”状态或是“濒空”状态。当缓冲区处于“爬坡”状态时，视频播放器可能正处在起始缓冲或是重新缓冲阶段，在这两个阶段下，由于播放器正在进行着视频初始化，所以此时的视频画面是停滞不动的。相反，如果此时缓冲区处于的是“濒空”状态，视频画面却是流畅播放的，因为，对于“濒空”状态来说，虽然此时缓冲区的占有量已经低于最小阈值min，但是只要不为零，视频画面总还是流动的。与之类似，当缓冲区的占有量为$l2$时，缓冲区可以处于“爬坡”状态或是“振荡”状态，此时的播放器又会分别处于起始缓冲和正常演播的状态，这两个状态对应的用户体验质量也是完全不同的。由此可见，只是简单地观察缓冲区占有量来推测用户体验远远不够，应该进一步区分出缓冲区占有量变化趋势。因此，我们提出了一种全新的应用层客观 QoE 度量指标：缓冲区综合状态，用于更合理细致的刻画缓冲区状态和播放器状态。

4.3 视频用户 QoE 的实时预测算法

本节将具体介绍我们的视频用户 QoE 实时预测算法，首先会介绍算法的基本模型和流程，然后会描述特征提取的步骤和具体方法，这些特征用于时序数据挖掘过程中预测器的训练和评估，以实现对于播放器状态的实时预测。

4.3.1 算法简介

图 4-3 是视频用户 QoE 实时预测算法的大致流程。

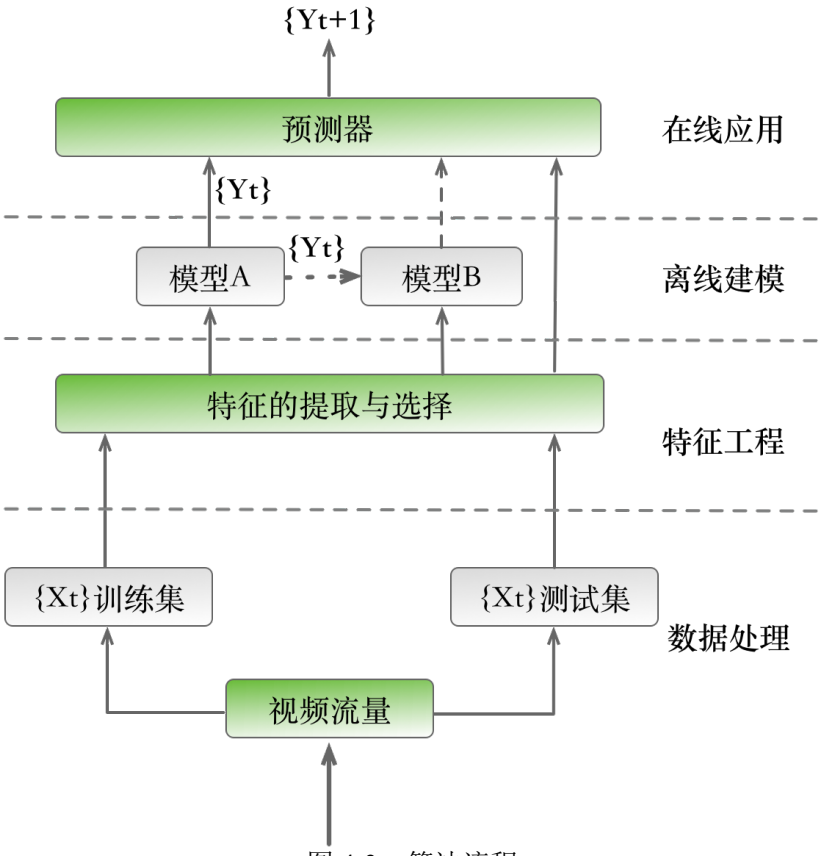


图 4-3 算法流程
Fig 4-3 Algorithm flow

如图 4-3 所示，算法的输入是 HAS 视频的网络流量数据，输出为播放器的缓冲区状态，算法的流程经历四个阶段，包含四个主要部分：

（1）数据处理。源数据X是 HAS 视频播放过程中客户端与服务器之间的网络封包数据流，根据传输方向，可以将数据流分为 HTTP 请求流X_r和 TCP 下载流X_d，即$X = \{X_r, X_d\}$。类别数据Y是根据“缓冲区综合状态”，对播放器缓冲区状态划分的四类状态值。由于视频播放的时间特性，网络封包数据X和播放器缓冲区状态Y都是按照时间演变的，因此，可以将X和Y表征为时间序列的形式。此外，我们还需要对 TCP 下载流的包序列按照“块”粒度重建，得到视频块序列。经过背景流去除、时间对齐、视频块序列重建之后，我们在数据预处理阶段得到两个时间序列：$\{X_t\}$和$\{Y_t\}$。

（2）特征工程。参考 HAS 视频流的一般传输机制，对$\{X_t\}$进行特征提取，获取到用于进行模型训练和测试的数据集。我们在 4.3.2 节中对特征的提取过程和细节进行了详细描述。

（3）离线建模。训练模型 A 和模型 B，模型 A 用于对播放器缓冲区状态进行实时判断，模型 B 用于对下一个时刻的播放器缓冲区状态进行预测。对于模型 A 和模型 B，我们采用“离线异步训练”和“在线同步应用”的方式进行构建。先使用训

练集构建模型 A，当模型 A 构建完成后，再综合训练集与模型 A 的分类结果构建模型 B。构建过程中，使用测试集进行模型评估，用以指导模型参数修正和效果提升。参数稳定后得到的最终模型即预测器。

（4）在线应用。利用训练好的预测器，在线预测播放器缓冲区的实时状态。当有一组新的视频流量数据到达时，首先对数据进行背景流去除、块序列重建等预处理操作，然后通过特征提取构建出时间序列X_t，再按照先后顺序将每一个时刻的X_t依次送入模型 A，模型 A 对t时刻送入的样本值X_t进行判断，得到当前时刻的状态判断结果Y_t，并将Y_t实时传送给预测器，预测器综合过去 n 时刻的时序特征$\{X_{t-n}, X_{t-n+1}, \dots, X_t\}$和模型 A 的判断结果$\{Y_{t-n}, Y_{t-n+1}, \dots, Y_t\}$，预测下一个时刻的状态值$Y_{t+1}$。

4.3.2 特征提取

我们的想法是建立起视频流量模式与缓冲区状态之间的映射关系，从而可以基于网络侧视频流量的 QoS 参数，实现对客户端视频用户 QoE 的跨层感知。在客户端，我们可以通过 4.2 节中提出的新的客观 QoE 指标：缓冲区综合状态，区分缓冲区的状态；然而在网络侧，我们还需要对网络流量进行特征挖掘，选取合适的 QoS 参数作为视频流特征，用于映射模型的训练和测试。本小节中我们将介绍如何从视频块序列中提取特征。

我们注意到，由于 ABR 算法的使用，HAS 视频客户端会根据缓冲区占有量和当前网络条件，动态地选择下一个视频块的请求码率。因此，客户端对视频块码率的请求情况（例如请求间隔、码率等级、码率切换程度）是一种可以很好反映缓冲区状态的信息。而对于运营商来说，虽然不能直接获取到客户端对视频块的码率请求情况，但是根据上一章中提出的“基于 HAS 视频流的块序列重建算法”，我们可以从加密视频流中重建视频块序列，并且实验表明，使用我们算法重建的视频块序列，可以很好地拟合客户端视频块请求序列的变化趋势。因此，运营商可以通过对网络流量的视频块序列进行特征挖掘，用以预测播放器的缓冲区状态。

下面，我们从一次 HAS 视频实际演播时，客户端参数（请求码率、缓冲区占有量）与网络流量块序列的参数（HTTP 请求时刻、块流量差值）之间的变化关系，说明从网络流量的视频块序列中挖掘缓冲区状态信息的可能性。如图 4-4 和图 4-5 所示。

图 4-4 是一次视频演播过程中，客户端请求码率、缓冲区占有量和网络中 HTTP 请求报文的时间序列，其中绿色实线是客户端的缓冲区占有量曲线，蓝色虚线是客户端请求码率的时间序列，红色数据点是网络中 HTTP 请求报文的离散序列，如

图 4-4 所示，在 10s-35s 这段时间内，当缓冲区占有量充裕的时候，客户端为了保证视频清晰度，提升用户的观看体验，更倾向于请求高码率的视频块；在 35s-45s 这段时间，当缓冲区占有量匮乏的时候，客户端为了保证视频流畅度，维持用户的观看体验，更倾向于请求低码率的视频块；客户端请求不同码率的视频块，会使得缓冲区的填充速度不同，而与此同时，客户端的播放速度是不变的，这就导致了缓冲区占有量一直经历着上升-下降的周期性振荡。此外，在 10s-35s 这段时间，因为开始时缓冲区占有量充裕，客户端开始稀疏地请求高码率的视频块，因此，这段时间网络中 HTTP 报文的请求间隔也较大；而在 35s-45s 这段时间，因为缓冲区占有量匮乏，客户端为了积攒数据块，开始以低码率频繁的请求新视频块，此时的网络中出现了密集的 HTTP 请求报文。

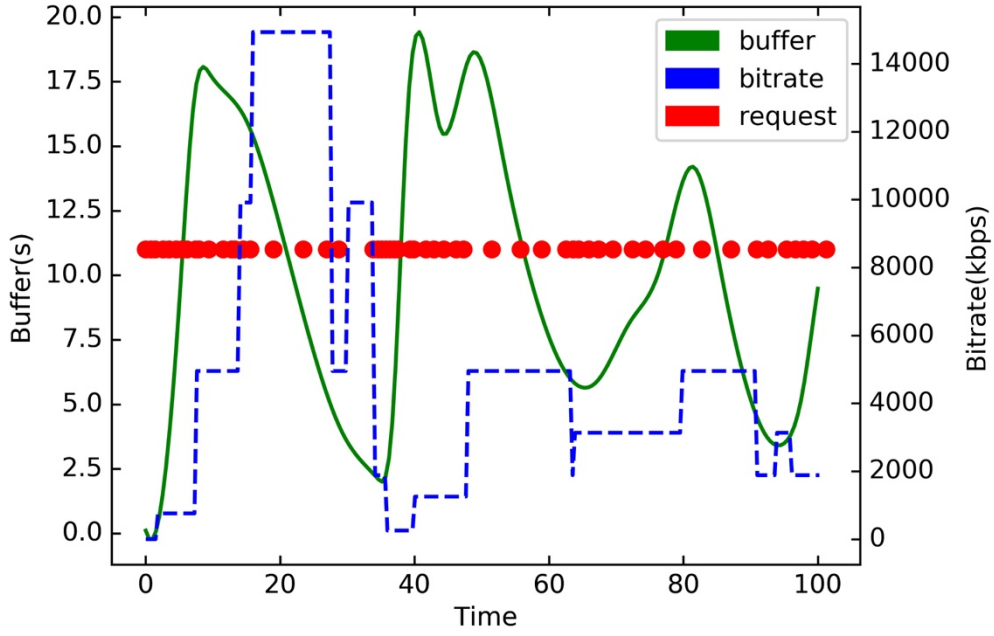


图 4-4 码率、缓冲区占有量与 HTTP 请求报文序列

Fig 4-4 The relationship between bitrate, buffer length and HTTP request message sequence

图 4-5 是一次视频演播过程中，客户端请求码率与网络中视频块差分流量的时间序列，其中红色虚线是客户端请求码率的时间序列，蓝色实线是网络中相邻两个视频块的流量差值序列。在图 4-5 中，0-40s、140s-180s 这段时间，由于客户端的请求码率大幅度的升高或下降，网络中视频块流量的差值呈现出比较明显的高尖峰；而与之相对，在 200s-400s 这段时间，因为客户端持续在低码率下进行码率请求和切换，此时网络中视频块流量差值也相对较小并且比较平缓。由此可见，客户端播放器视频块码率请求的动作信息（例如视频块密集请求、请求码率等级变化），都会在网络流量的块序列的特征中表现出来。

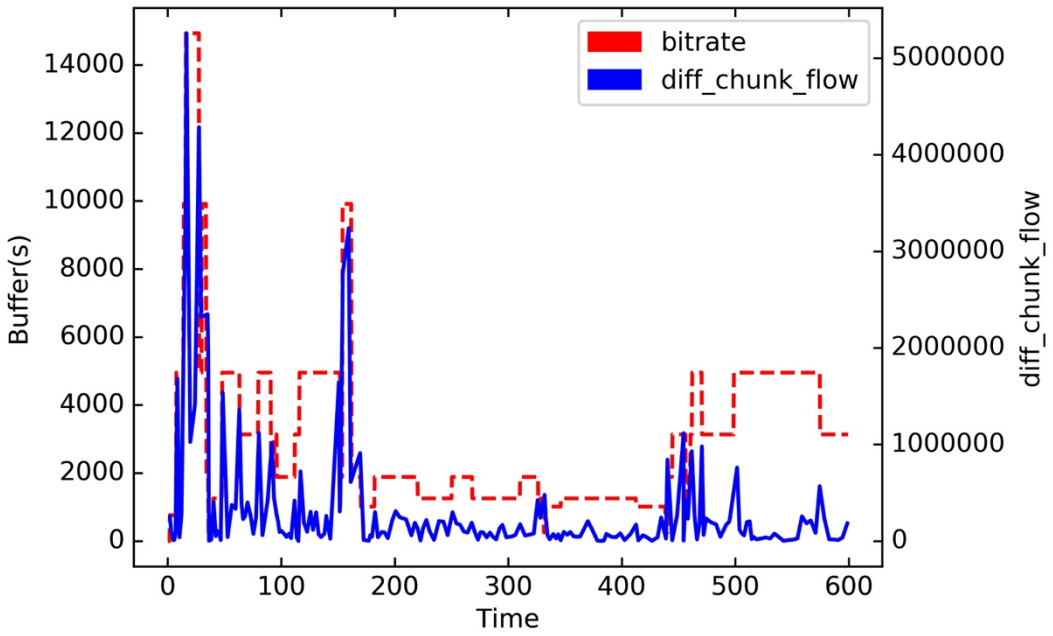


图 4-5 码率序列与块序列

Fig 4-5 The relationship between bitrate and chunk flow sequence

基于此，我们分别从 HTTP 请求流和 TCP 下载流中提取了四类特征：请求间隔、块流量、块残差、绝对块残差，每个特征的描述信息如表 4-1 所示。

表 4-1 块序列特征

Tabel 4-1 Characteristics of video chunk traffic sequences

特征名称	特征描述	来源
请求间隔	当前视频块的请求时刻与前一个视频块的差值	HTTP 请求流
块流量	不同序号的视频块中包含的字节数	TCP 下载流
绝对块残差	当前视频块的流量与前一个视频块的绝对差值	TCP 下载流
块残差	当前视频块的流量与前一个视频块的差值，差值可以是正值、负值或是 0，差值的正负代表了相邻两个视频块请求码率的升降变化	TCP 下载流

由于不同码率视频块的大小差异比较大，导致模型在训练时会更倾向于选择数值差别较大的特征（例如块流量），影响到模型的判断。因此，还需要对所有的特征进行“归一化”处理，将特征的数值统一到一个大致相同的特征区间中。本文中，我们使用的是归一化方法是“线性函数归一化”，将同一特征下的所有数值映射到[0,1]的范围，实现对于原始特征数值的等比缩放。归一化公式 4-1 如下：

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (4\text{-}1)$$

4.4 算法实现及实验结果分析

4.4.1 算法实现

我们算法的基本框架已经在 4.3.1 节进行介绍，具体实现时，框架中模型 A 和模型 B 可以选用不同的基本模型进行训练。本文中，我们使用随机森林作为模型 A。使用长短期记忆神经网络（Long Short-Term Memory，LSTM）作为模型 B。

我们在 4.2 节中提出了全新的 QoE 度量指标：缓冲区综合状态，并指出播放器缓冲区的状态不仅仅与缓冲区的占有量有关，还与缓冲区占有量的变化方向有关，也就是过去一段时间缓冲区占有量的变化趋势，因此，我们希望存在这种神经网络模型，当神经元对播放器下一时刻的缓冲区状态进行预测时，输入的是过去一段时间的视频流量信息，而不单单是当前时刻下的，换句话说，对于样本序列，该模型在时间轴上要具有“回头看”的能力。同时，该模型在对当前时刻神经元的预测结果进行输出的时候，还要能同时输入给下一时刻的神经元，用以指导下一个神经元的状态预测，即该模型对于状态预测结果要具有保存和“记忆”的能力。而对于这两种需求，长短期记忆神经网络都可以很好地满足，所以我们选择训练 LSTM 网络用做我们算法中的预测器模型。

(1) 构建随机森林分类器

随机森林是一个包含多个决策树的集成学习算法，其输出的类别是由所有树输出类别的众数决定的。“兼听则明、少数服从多数”的判别准则，使得随机森林相较于其他分类器往往具有更高的判断准确度。我们使用随机森林构建模型 A。

在训练阶段， 我们将每一时刻的视频块流量特征作为一个样本点，将每个样本点下的多个特征（请求间隔、块流量、块残差、绝对块残差）组合成一个一维数组，然后按照时间顺序依次将样本点的特征数组送入随机森林，随机森林对特征样本点进行判断后，将结果与真实值进行比较用以指导算法参数的调整和优化。

与一般的机器学习分类问题不同，视频流量序列在时间上具有先后到达的特性，相邻样本点之间并不是完全独立的，如果简单的照搬 K 折交叉检验的方式进行模型训练，会导致训练数据集和测试数据集之间出现不合理的“时间相关性”，影响模型的判断结果，比如通过 K 折切分后的数据集，在实际训练和验证的过程中，由于并不区分每组样本集在时间上的先后顺序，就有可能出现在本轮模型训练时，测试用的数据集在时间上竟然要比训练用的数据集还要靠前，此时对于模型来说，就出现了训练数据集提前“泄漏”信息的问题，因为它相当于让模型在训练时就提前看到了“未来”的数据信息，而在测试时用“过去”的信息进行测试，此时得到的模型就会产生过拟合。并且，在传统的 K 折交叉检验算法中，K 折切分后的每组数据

集会在 K 轮训练中依次作为训练集或测试集使用，这种训练方法原本是想在样本数量不多的情况下充分利用数据集的信息，使每个样本都能够参与训练和测试，但是在时间序列数据中，这种操作方式显然是不合适的，因为它会导致在本轮训练时，在时序上那些本不应该提前让模型看到的数据点，却已经在之前几轮的训练中被用到，产生了信息泄漏，于是进一步加剧了模型的过拟合。

合适的做法是，每一轮用于模型测试的数据集应该是在以往的测试环节中从来没有用到过的新数据。因此，我们使用 TimeSeriesSplit 的方法对视频流量序列进行分割。如图 4-6 所示，标准的 K 折交叉检验算法是把数据集划分为 K 组，每次训练时将不同的数据组循环作为训练集（绿色方块）或测试集（红色条纹方块）；而 TimeSeriesSplit 的方法中，每一次训练的数据集（绿色方块）永远是前一次训练中数据集的超集（即后一轮的训练集永远包含前一轮训练集），测试集则是在以往的训练中都未使用过的、时间上靠后的数据（红色条纹方块）。

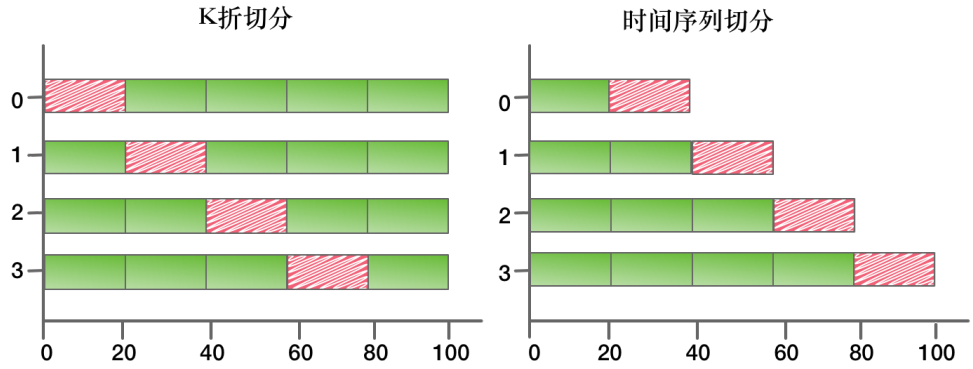


图 4-6 K 折切分与时序切分示意图

Fig 4-6 K-folding and timing segmentation

离线训练阶段，当随机森林分类器接收到一组视频流量序列，在对该序列下的所有样本点都依次完成状态分类之后，然后再将所有样本点的特征以及随机森林对每个时刻样本点的状态分类结果，组合成新的特征数组，按照时序统一送给 LSTM 用于训练；在线应用阶段，随机森林分类器每完成一次样本点的状态分类，就将结果送给 LSTM 预测器，作为该时刻下 LSTM 神经单元的补充输入信息之一，用于对下一时刻的播放器缓冲区状态进行预测。我们把这种训练和应用的过程叫做“离线异步训练、在线同步应用”。

(2) 构建 LSTM 预测器

长短期记忆神经网络（Long Short-Term Memory，LSTM）是一种时间递归网络，它最显著的魅力是可以将以前的信息保留到当前时刻的任务中，即对过往时刻的任务有一定的“记忆性”，例如在视频图像理解中，保留先前视频帧的信息可以有助于对视频当前帧内容的理解。

与常规的 LSTM 训练过程不同，本文中，为了能够同时用到缓冲区的状态分类信息，我们对 LSTM 预测器训练和应用过程中序列输入的形式有针对性地进行了更改。我们猜想，如果 LSTM 的输入序列中不仅包含视频块的网络流量序列，还包含播放器的缓冲区状态序列，结合其本身就具有的“回头看”和“记忆”的能力，LSTM 可以更好地对下一时刻的缓冲区状态进行预测。在现在的研究中，使用这种数据输入思想的方式主要有两种：第一种，能够实时获取到客户端播放器的缓冲区状态，这种方式对于“视野狭窄”的网络运营商来说并不现实，尤其是在线实时应用的情况下；第二种，是递归地将最近的预测结果用作后续预测的输入，这种方式下，后一步的预测会过度依赖前一步的预测结果，一旦在某一步发生预测错误，就会导致这种错误随着时间累积，效果波动性较大。我们的方法与这两种实现方式均不同，如图 4-7 所示。

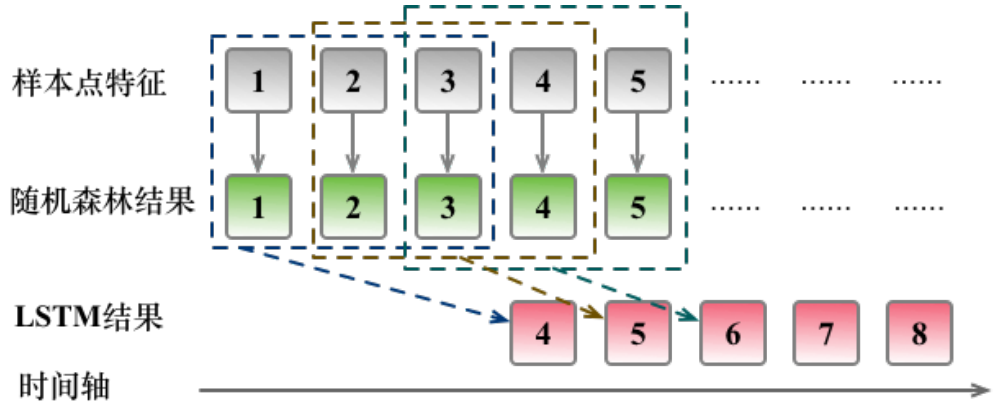


图 4-7 模型训练逻辑
Fig 4-7 Model training strategy

如图 4-7 所示，我们的方法是，将随机森林分类器的判断结果实时传送给 LSTM 预测器，支持 LSTM 综合多个时刻下的网络流量特征和缓冲区状态分类结果，对下一个时刻的缓冲区状态进行预测。

具体地说，我们将视频流量特征序列按照每一个时刻点为一个样本（如图 4-7 中第一层的一个个灰色方框）依次送入随机森林分类器，得到当前时刻下的缓冲区状态分类结果（如图 4-7 中第二层的绿色方框）。随机森林在将判断结果进行输出的时候，同时会输入给 LSTM 预测器，预测器以 n 个时刻为一个窗口（如图 4-7 中 n 为 3，代表每次“回看”3 步），综合最近 n 个时刻下的流量特征和缓冲区状态分类结果，预测下一个时刻的缓冲区状态（如图 4-7 中第三层的红色方框）。当有一个新的分类结果到达时，LSTM 预测器会将该窗口向后滑动一步，移除当前时间窗口下最前时刻的样本数据，将最新的分类结果和流量特征补充到样本数据的最后

时刻，然后将整个窗口作为一个新的样本数据集送入 LSTM，用于对下一个时刻的缓冲区状态进行预测。如此往复，直至窗口滑动到整个视频流序列的最后时刻。

4.4.2 实验环境和数据描述

我们在本章使用与第三章中相同的实验环境，实验台的具体参数已经在 3.2.1 节中有所介绍，此处不再赘述。我们观看的视频文件，包含动作片、科幻片、新闻报道、记录片、动态写真等多个类型，每种视频文件的信息如表格 4-2 所示。采集到的数据集中总共包含 32 条视频的网络流量轨迹和客户端播放日志。视频的流量轨迹是由捕获到的一个个视频封包组成的，每个封包的参数有：采集时刻、源 IP 地址、目标 IP 地址、协议、包长、基本信息；客户端播放日志是视频播放的实时信息，包括请求码率、缓冲区占有量等参数。

表 4-2 视频文件信息
Table 4-2 Video file information

文件类型	视频码率范围	音频码率	持续时长
科幻片 1	4 种码率(150kbps-750kbps)	72kbps	10min
科幻片 2	5 种码率(386kbps-2773kbps)	131kbps	12min
动画片	10 种码率(254kbps-14931kbps)	65kbps	10min
纪录片	2 种码率(3000kbps-4000kbps)	128kbps	5min
运动集锦	2 种码率(2500kbps-4000kbps)	96kbps	6min
动态写真	6 种码率(2859kbps-19683kbps)	194kbps	3min
电视屏保	1 种码率(1144kbps)	191kbps	5min

对于每一条视频的网络流量序列和客户端日志序列，我们以它的 MPD 文件请求时刻为两个序列的共同初始时刻，以 0.25s 的时间间隔对两个序列进行重新采样，同时使用客户端的日志序列对流量序列进行标签化，将两个序列整合到同一个时间轴下并进行监督化处理。

4.4.3 结果分析

我们使用准确率、精确率、召回率、F1 分数作为指标，评估模型的性能。精确率又叫查准率、召回率又叫查全率。具体定义如下：

准确率 = 正确分类的样本个数 / 总样本个数

精确率 = 分类正确的正样本个数 / 分类器判定为正样本的个数

召回率 = 分类正确的正样本个数 / 真正的正样本个数

F1 分数 = 精确率 * 召回率 * 2 / （精确率+召回率）

(1) 随机森林分类器的状态分类结果分析

我们使用 Sklearn 中的随机森林模型训练缓冲区状态分类器，基于每一时刻的网络流量特征对播放器缓冲区状态进行实时判断。具体地说，首先需要将每条视频的 TCP 下载流按照“块”粒度重建，得到各自的块序列，并根据每个视频块的大小（包含的字节数）和请求时间得到视频流量的四组特征序列：请求间隔序列、块序列、块残差序列和绝对块残差序列，最后将每个视频流中 MPD 文件的请求时间作为标准初始时刻，并以 0.25s 为采样间隔，将同一条视频流的客户端日志序列（包含缓冲区占有量和码率等级）与视频流量的四组特征序列进行时间对齐，组合成一组多元时间序列，并根据缓冲区占有量和变化趋势，标注每个时刻样本点的缓冲区综合状态（此处，我们使用四种缓冲区综合状态：0、1、2、3 为每个样本点打标，这四种缓冲区综合状态分别对应缓冲区的四种状态：爬坡、振荡、濒空、枯竭），从而将每个时刻点下的网络流量特征与播放器缓冲区状态一一对应起来，完成数据的标签化处理。

为了增加模型训练数据的多样性，提高评估结果的说服力，我们在对随机森林模型的训练过程中，有意识的引入了不同类型不同内容的多条视频数据。我们将表 4-2 中的 7 种类型的视频数据在不改变时序的情况下首尾拼接，整合后的带有标签的序列一共包含 61042 个时刻点，其中爬坡状态 3833 个、振荡状态 51454 个、濒空状态 3761 个、枯竭状态 1994 个。将样本序列按照图 4-6 中介绍的时间序列切分方式进行数据分割后开始模型的训练和评估。表 4-3 是随机森林模型的评估结果。

从表中数据可见，“爬坡”状态和“振荡”状态的识别精确率（查准率）高于 80%，但是，相较于这两种状态，“濒空”状态和“枯竭”状态的识别精确度只有 64%和 71%。

表 4-3 随机森林模型的评估结果

Table 4-3 Evaluation results of random forest models

缓冲区状态	类别	精确率	召回率	F1 分数	测试样本数
爬坡	0	0.81	0.95	0.87	1166
振荡	1	0.99	0.94	0.97	15413
濒空	2	0.64	0.85	0.73	1133
枯竭	3	0.71	0.94	0.81	601

我们通过绘制随机森林评估结果的混淆矩阵分析其原因，如表 4-4 所示。

表 4-4 随机森林模型评估结果的混淆矩阵
Table 4-4 Confusion matrix of random forest model evalution results

真实标签	预测标签			
	爬坡	振荡	濒空	枯竭
爬坡	94.59%(1103)	2.83%(33)	1.03%(12)	1.55%(18)
振荡	1.52%(234)	94.37%(14545)	**3.3%(509)**	**0.81%(125)**
濒空	0.794%(9)	6.71%(76)	**84.73%(960)**	**7.76%(88)**
枯竭	2.33%(14)	0.49%(3)	3.16%(19)	**94%(565)**

表 4-4 中是随机森林评估结果的混淆矩阵，其中每个表格中数据的格式是：真实标签为 A 的样本中被预测为 B 的概率，即识别准确率（真实标签为 A 的样本中被预测为 B 的样本个数）。

综合表 4-3 和表 4-4，可以看到，虽然濒空状态的识别精确率只有 64%，但是该状态下的识别准确率却高达 84.73%（即参与测试的 1133 个濒空状态样本中有 960 个均被正确识别），之所以会产生较低的识别精确率，是因为参与测试的 15413 个振荡状态测试样本中有 509 个被错误识别为濒空状态，虽然这部分被错判的数据只占振荡状态测试样本自身的 3.3%，但是从实际数目上看，却基本相当于濒空状态测试样本总数的一半以上，极大的影响了濒空状态下识别精确率的评估。同理，枯竭状态的评估结果中也存在状态识别精确率仅 71%，但是该状态下识别准确率高达 94%的情况。综上所述，产生这种在所有状态下的识别精确率和在单个状态下的识别准确率存在较大偏差的原因，是因为参与训练和测试的样本自身存在极大的不平衡，以本次模型训练为例，总样本数为 61042，四种状态的样本占比分别为：6.28%（爬坡）、84.29%（振荡）、6.16%（濒空）、3.27%（枯竭），由此可见，样本集中振荡状态样本数要远远高于其他状态样本数，所以振荡状态的评估波动会极大地影响到其他状态的识别精确率，尤其是“濒空”状态，因为 “振荡”状态与“濒空”状态在根据缓冲区综合状态进行状态划分时，距离相对其他状态更近，在根据“缓冲区占有量和变化方向”进行状态打标时，处于缓冲区占有量分类临界处的这两种状态也相对更容易产生混淆。

但是，从表 4-3 中同样可以看到，随机森林分类器对四种缓冲区状态的识别召回率（查全率）至少为 85%，平均高于 90%，并且综合指标 F1 分数（精确率和召回率的调和平均值）至少为 73%，平均高于 85%。

(2) LSTM 预测器的状态预测结果分析

我们使用 Keras 中的 LSTM 模型训练缓冲区状态预测器，基于过去一段时间的网络流量特征和随机森林状态评估结果对下一时刻的缓冲区状态进行预测。4.4.1 小节中图 4-7 已经介绍了本文中 LSTM 预测器的构建细节。

我们将同一条视频流的块序列和随机森林分类器的状态评估序列称为一条视频序列，使用 26 条不同类型（视频文件信息如表 4-2 所示）的视频序列构建 LSTM 模型，从中随机抽取 18 条视频序列（包含 29760 个样本时刻点）用于训练，剩下的数据用于测试（包含 13680 个样本时刻点）。

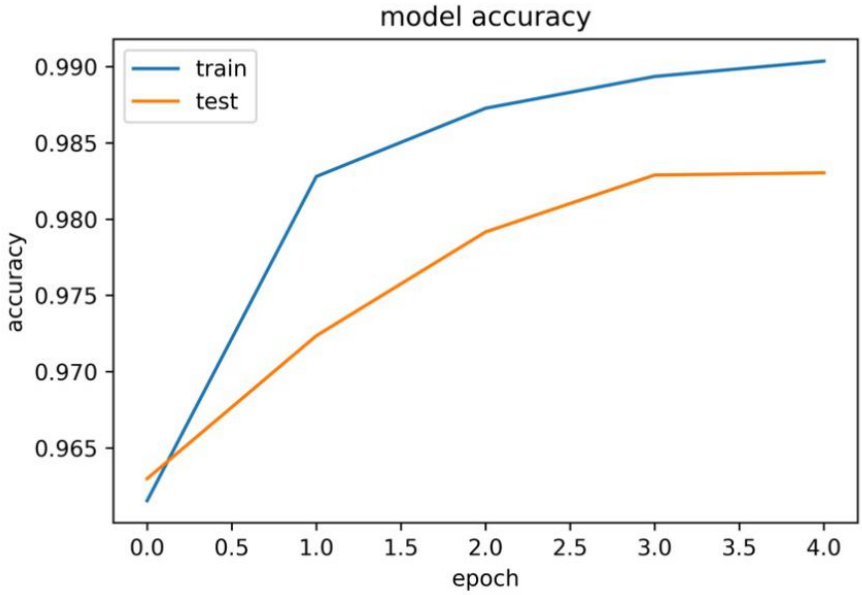


图 4-8 accuracy 变化曲线

Fig 4-8 Accuracy curve

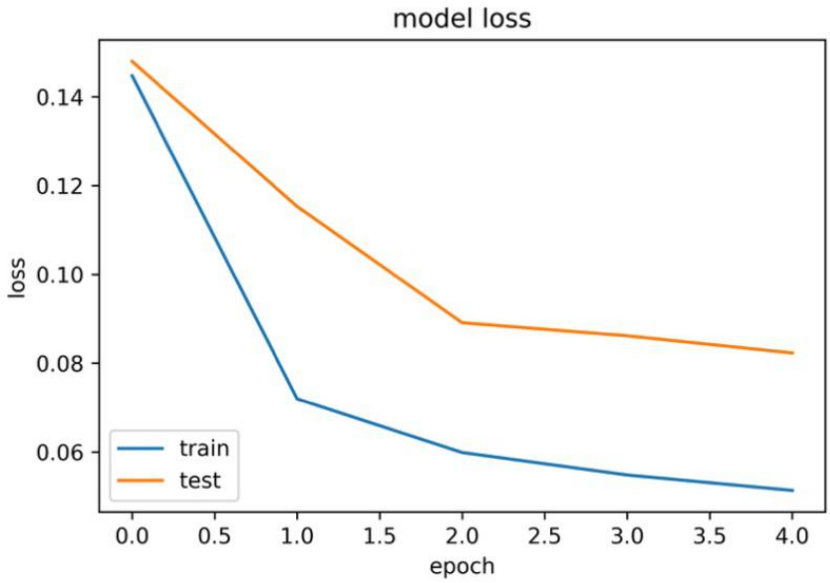


图 4-9 loss 变化曲线

Fig 4-9 Loss curve

图 4-8 和 4-9 分别是 LSTM 预测器训练时的模型精确率变化曲线和模型损失变化曲线。如图所示，模型在迭代 5 次后参数不再提升，停止训练，此时模型的测试集精确率高达 98%，损失值低于 0.1。

我们将训练集中每个时刻样本的评估结果导出，观察每个类别的识别精确率和召回率，如表 4-5 所示：

表 4-5 LSTM 模型的评估结果
Table 4-5 Evaluation results of the LSTM model

缓冲区状态	类别	精确率	召回率	F1 分数	测试样本数
爬坡	0	0.97	0.96	0.96	1197
振荡	1	0.98	0.98	0.98	10398
濒空	2	0.87	0.88	0.88	1387
枯竭	3	0.96	0.93	0.95	692

表 4-5 是 LSTM 模型的状态评估结果。虽然测试样本仍然存在明显的数据不平衡，但是四种类别的状态预测精确率（查准率）高于 87%，召回率（查全率）高于 88%，F1 分数高于 88%，虽然濒空状态的各项评估指标仍然显著低于其他状态类别，但是相较于随机森林分类器的评估结果存在明显提升。模型具有理想的预测效果。

4.5 本章小结

为了从运营商的角度对 HAS 视频用户的 QoE 进行感知，本章基于 HAS 视频的加密流量，提出了一种视频用户 QoE 预测算法。

首先，不同于现有研究中直接估计用户的主观 QoE，我们通过评估应用层的客观 QoE 指标，表征用户的主观 QoE，不需要组织用户进行主观 QoE 测试，模型客观、具有通用性。

此外，与现有研究中直接使用“缓冲区占有量”作为播放器演播状态的衡量指标不同，我们发现，客户端的演播状态不仅与缓冲区实时占有量有关，还与缓冲区占有量的变化趋势有关。因此，本文还提出了一种全新的应用层客观 QoE 度量指标：缓冲区综合状态，该指标同时综合了缓冲区占有量和变化趋势的信息，可以更细致、合理地表征播放器状态。

其次，为了只使用视频传输过程中网络侧流量数据的统计信息，实时预测客户

端的视频播放状态，从而实现对视频用户 QoE 的跨层感知。我们对 HAS 客户端与服务器之间的双向流量数据进行“块”粒度重建，使用时间序列数据挖掘的方法，基于重建的视频块序列，建立网络流量 QoS 特征参数与新的客观 QoE 指标的映射模型。我们采用“离线异步训练、在线同步应用”的方法使用随机森林模型和 LSTM 模型构建缓冲区状态预测器。其间，在随机森林模型的训练过程中，我们使用 TimeSeriesSplit 数据切分方法，规避了传统 K 折切分过程中训练数据集和测试数据集之间可能会出现的不合理“时间相关性”问题，防止模型过拟合；在 LSTM 模型的训练过程中，为了能够充分利用缓冲区状态的时序信息，我们对 LSTM 预测器中序列输入的形式有针对性地进行了更改。综合来说，我们的方法中使用了更适用于时间序列的模型和训练技巧，充分的利用了 HAS 视频流的时序特征，处理方式更贴近该数据类型的特点。

最后，我们搭建了实验数据采集环境，并对模型的预测效果进行了评估，实验结果表明，采用我们的方法对视频演播状态进行实时预测的精确度高于 87%，漏判率低于 12%。

5 结论

近年来，视频流媒体已经成为目前因特网的主要应用。为了给视频用户提供更好的网络服务，运营商有必要了解并保障视频流的服务质量和视频用户的观看体验。在众多的评价指标中，体验质量（QoE）综合考虑了端到端的服务质量（QoS）和用户主观因素，代表了用户对视频服务的主观满意程度。因此，运营商可以通过感知用户 QoE，了解用户感受，指导网络资源的调度，从而更好的保证视频用户无缝高质量的流媒体体验。在众多视频传输技术中，HTTP 自适应流媒体传输技术（HAS）不仅继承了传统流式传输技术“边下边播、随意拖动”的特点，并且引入智能化的视频播控逻辑，目前已广泛应用在视频传输中。

HAS 视频传输技术的引入，一定程度上改善了网络视频用户的体验。但是，也同时给运营商感知用户 QoE 带来了几点技术困难。首先，端到端的 HAS 视频传输机制中，每个端（服务器端、网络侧、客户端）“个体独立、任务分管”的服务原则，使得运营商的“视野”狭窄有限，无法直接获取到应用层客户端的视频演播信息，只能通过挖掘网络流量感知用户 QoE；其次，HAS 机制中 ABR 算法的应用，会进一步模糊掉网络流量波动所带来的用户 QoE 信息，使得运营商无法对用户 QoE 得以改善的原因进行准确追踪，给运营商感知用户 QoE 造成干扰；此外，目前越来越多的 HAS 视频流开始采用加密传输，视频流的加密传输也给运营商挖掘网络流量信息增大了难度。

本论文的第二部分工作，提出了一种新颖地从加密流量中实时预测 HAS 视频用户 QoE 的方法。在提出该方法之前，为给流量特征提取、建立网络流量与 QoE 指标映射模型，提供合理有效的数据挖掘方向。本论文的第一部分工作提出了一种基于 HAS 加密视频流的块序列重建算法。并基于网络测量对 HAS 视频的流量传输特性进行了分析。

5.1 本文主要工作

本论文的主要工作分为以下两个部分：基于加密视频流特征分析的块序列重构，以及基于加密流量的 HAS 视频用户 QoE 感知。

5.1.1 基于加密视频流特征分析的块序列重构

本部分主要总结基于加密视频流特征分析的块序列重构的主要研究成果，这

部分研究的具体贡献可以罗列如下：

(1) 介绍了对视频流量基于“块”粒度进行分析的好处，分析了从视频流量中重建“视频块序列”对于运营商感知视频用户 QoE 的必要性，以及在重建过程中遇到的三个主要问题。

(2) 基于网络测量对HAS加密视频流的传输特性进行了分析。根据分析发现，详述了 HAS 技术在实际应用中的四种流量传输特性：双流并传、分块并传、同块同 ACK 号、分块重传。

(3) 提出一种从加密视频流量中重建视频块序列的方法。根据视频流量传输特性，分析了视频块序列在重建过程中可能会遇到干扰问题，并提出了一种集合了视频流量过滤、分块边界识别与聚合、分块类型识别、分块类型矫正的视频块序列重建算法。

(4) 搭建实验数据采集环境，评估“视频块序列重建算法”的效果。实验结果表明，使用我们的算法从网络流量中重建的视频块序列与从客户端抓取到的视频块请求序列之间的样本均方根误差不高于 0.132，具有很好的拟合度。

5.1.2 基于加密流量的 HAS 视频用户 QoE 感知

本部分主要总结基于加密流量的 HAS 视频用户 QoE 感知的主要成果，这部分研究的具体贡献可以罗列如下：

(1) 我们的方法是通过评估应用层的客观 QoE 指标，表征用户的主观 QoE。不同于现有研究中直接估计用户的主观 QoE，不需要组织用户进行主观 QoE 测试，模型客观、具有通用性。

(2) 提出一种全新的客观 QoE 度量指标。与现有研究中，直接把缓冲区占有量作为客户端播放状态的度量指标不同，我们发现播放器的播放状态和用户的观看体验不仅仅与缓冲区的占有量有关，还与缓冲区占有量的变化方向有关。因此，我们提出了一种全新的应用层客观 QoE 度量指标：缓冲区综合状态，可以更合理细致的刻画缓冲区状态和播放器状态。

(3) 基于从流量中重建的视频块序列，使用时间序列数据挖掘方法，建立网络流量特征与客观 QoE 指标的映射模型，实现从加密视频流量中实时预测 HAS 视频用户的 QoE。预测算法中使用了适用于时间序列的模型和训练技巧，充分利用了 HAS 视频流的时序特征，处理方式更贴近该数据类型的特点。

(4) 搭建了实验数据采集环境，并对模型的预测效果进行了评估。实验结果表明，采用我们的方法对视频演播状态进行实时预测的准确度高于 87%，漏判率低于 12%。

5.2 未来工作展望

本论文的研究工作还可以从如下方面进行提升：

(1) 在重建视频块序列的研究中，还可以尝试从流量中挖掘出视频块的内容持续时间，进而实现能够直接从加密流量中算出每一个视频块的码率。

(2) 在构建网络流量特征与应用层客观 QoE 指标之间映射模型的研究中，还需要通过特征提取和平衡样本集等方法，解决因为样本不平衡导致的模型评估误差的问题。

参考文献

[1] 2018.Technical Report - SANDVINE: The 2018 Global Internet Phenomena report. https://www.sandvine.com/phenomena.(2018).

[2] Mangla T , Halepovic E , Ammar M , et al. MIMIC: Using passive network measurements to estimate HTTP-based adaptive video QoE metrics[C]// 2017 Network Traffic Measurement and Analysis Conference (TMA). IEEE, 2017.

[3] Mangla T, Halepovic E, Ammar M, et al. eMIMIC: Estimating HTTP-Based Video QoE Metrics from Encrypted Network Traffic[C]// 2018 Network Traffic Measurement and Analysis Conference (TMA). 2018.

[4] Xu S, Sen S, Mao Z M, et al. Dissecting VOD services for cellular: performance, root causes and best practices[C]//Proceedings of the 2017 Internet Measurement Conference. ACM, 2017: 220-234.

[5] Tang S, Qin X W, Wei G. Analysis on the state of mobile HTTP video streaming at the client-side[C]//2015 International Conference on Wireless Communications & Signal Processing (WCSP). IEEE, 2015: 1-6.

[6] Farshad A, Georgopoulos P, Broadbent M, et al. Leveraging SDN to provide an in-network QoE measurement framework[C]//2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE, 2015: 239-244.

[7] Huysegems R, De Vleeschauwer B, De Schepper K, et al. Session reconstruction for HTTP adaptive streaming: Laying the foundation for network-based QoE monitoring[C]//Proceedings of the 2012 IEEE 20th International Workshop on Quality of Service. IEEE Press, 2012: 15.

[8] Wu T, Huysegems R, Bostoen T. Scalable network-based video-freeze detection for HTTP adaptive streaming[C]//2015 IEEE 23rd International Symposium on Quality of Service (IWQoS). IEEE, 2015: 95-104.

[9] 鲁西. 基于 SDN 的保证 QoS 的网络资源分配和管理[D]. 2017.

[10] Hongyun Zheng, Yongxiang Zhao, Xi Lu, Rongzhen Cao, "A Mobile Fog Computing-Assisted DASH QoE Prediction Scheme", Wireless Communications and Mobile Computing, vol. 2018, https://doi.org/10.1155/2018/6283957.

[11] Martín V, Cabrera J, García N. Design, optimization and evaluation of a Q-Learning HTTP Adaptive Streaming Client[J]. IEEE Transactions on Consumer Electronics, 2016, 62(4): 380-388.

[12] De Grazia M D F, Zucchetto D, Testolin A, et al. QoE multi-stage machine learning for dynamic video streaming[J]. IEEE Transactions on Cognitive Communications and Networking, 2018, 4(1): 146-161.

[13] Nguyen T T T, Armitage G J. A survey of techniques for internet traffic classification using machine learning[J]. IEEE Communications Surveys and Tutorials, 2008, 10(1-4): 56-76.

[14] Pan W, Cheng G, Wu H, et al. Towards QoE assessment of encrypted YouTube adaptive video streaming in mobile networks[C]//2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS). IEEE, 2016: 1-6

[15] Pan W, Cheng G. QoE assessment of encrypted YouTube adaptive streaming for energy saving in Smart Cities[J]. IEEE Access, 2018, 6: 25142-25156.

[16] Jialin Zhang, Hongyun Zheng, Yongxiang Zhao and Yuchun Guo. "Bystander: QoE Perception for Dynamic Video Streaming from Encrypted Traffic." IEEE International Conference on Consumer Electronics-taiwan 2019.

[17] Mushtaq M S, Augustin B, Mellouk A. Empirical study based on machine learning approach to assess the QoS/QoE correlation[C]//2012 17th European Conference on Networks and Optical Communications. IEEE, 2012: 1-7.

[18] Alreshoodi M, Woods J. Survey on QoE\QoS correlation models for multimedia services[J]. arXiv preprint arXiv:1306.0221, 2013.

[19] Vassilev T I, Anthony P J. International Journal of Mobile Computing and Multimedia Communications[J]. Communications, 2016, 7(3).

[20] Gastaldo P, Zunino R, Redi J. Supporting visual quality assessment with machine learning[J]. EURASIP Journal on Image and Video Processing, 2013, 2013(1): 54.

[21] Balachandran A, Sekar V, Akella A, et al. Developing a predictive model of quality of experience for internet video[C]//ACM SIGCOMM Computer Communication Review. ACM, 2013, 43(4): 339-350.

[22] Aroussi S, Mellouk A. Survey on machine learning-based QoE-QoS correlation models[C]//2014 International Conference on Computing, Management and Telecommunications (ComManTel). IEEE, 2014: 200-204.

[23] Chen Y, Wu K, Zhang Q. From QoS to QoE: A tutorial on video quality assessment[J]. IEEE Communications Surveys & Tutorials, 2015, 17(2): 1126-1165.

[24] Casas P, Wassermann S. Improving QoE prediction in mobile video through machine learning[C]//2017 8th International Conference on the Network of the Future (NOF). IEEE, 2017: 1-7.

[25] Ghadiyaram D, Pan J, Bovik A C. Learning a continuous-time streaming video QoE model[J]. IEEE Transactions on Image Processing, 2018, 27(5): 2257-2271.

[26] Bampis C G, Li Z, Katsavounidis I, et al. Recurrent and dynamic models for predicting streaming video quality of experience[J]. IEEE Transactions on Image Processing, 2018, 27(7): 3316-3331.

[27] Zhang C, Patras P. Long-term mobile traffic forecasting using deep spatio-temporal neural networks[C]//Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing. ACM, 2018: 231-240.

[28] Martín V, Cabrera J, García N. Design, optimization and evaluation of a Q-Learning HTTP Adaptive Streaming Client[J]. IEEE Transactions on Consumer Electronics, 2016, 62(4): 380-388.

[29] Triki I, El-Azouzi R, Haddad M, et al. Learning from experience: A dynamic closed-loop QoE optimization for video adaptation and delivery[C]//2017 IEEE 28th Annual International

Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC). IEEE, 2017: 1-5.

[30] Mao H, Netravali R, Alizadeh M. Neural adaptive video streaming with pensieve[C]//Proceedings of the Conference of the ACM Special Interest Group on Data Communication. ACM, 2017: 197-210.

[31] Krishnamoorthi V, Carlsson N, Halepovic E, et al. BUFFEST: Predicting buffer conditions and real-time requirements of HTTP (S) adaptive streaming clients[C]//Proceedings of the 8th ACM on Multimedia Systems Conference. ACM, 2017: 76-87.

[32] Akhshabi S, Anantakrishnan L, Begen A C, et al. What happens when HTTP adaptive streaming players compete for bandwidth?[C]//Proceedings of the 22nd international workshop on Network and Operating System Support for Digital Audio and Video. ACM, 2012: 9-14.

[33] Ameigeiras P, Ramos-Munoz J J, Navarro-Ortiz J, et al. Analysis and modelling of YouTube traffic[J]. Transactions on Emerging Telecommunications Technologies, 2012, 23(4): 360-377.

[34] Tsilimantos D, Karagkioules T, Nogales-Gómez A, et al. Traffic profiling for mobile video streaming[C]//2017 IEEE International Conference on Communications (ICC). IEEE, 2017: 1-7.

[35] Rao A, Legout A, Lim Y, et al. Network characteristics of video streaming traffic[C]//Proceedings of the Seventh Conference on emerging Networking Experiments and Technologies. ACM, 2011: 25.

[36] Sieber C, Blenk A, Hinteregger M, et al. The cost of aggressive HTTP adaptive streaming: Quantifying YouTube's redundant traffic[C]//2015 IFIP/IEEE International Symposium on Integrated Network Management (IM). IEEE, 2015: 1261-1267.

[37] Gill P, Arlitt M, Li Z, et al. Youtube traffic characterization: a view from the edge[C]//Proceedings of the 7th ACM SIGCOMM conference on Internet measurement. ACM, 2007: 15-28.

[38] Ali-Eldin A, Kihl M, Tordsson J, et al. Analysis and characterization of a video-on-demand service workload[C]//Proceedings of the 6th ACM Multimedia Systems Conference. ACM, 2015: 189-200.

[39] Biernacki A. Identification of adaptive video streams based on traffic correlation[J]. Multimedia Tools and Applications, 2019: 1-21.

[40] Biernacki A. Analysis and modelling of traffic produced by adaptive HTTP-based video[J]. Multimedia Tools and Applications, 2017, 76(10): 12347-12368.

[41] Biernacki A. Analysis of aggregated HTTP-based video traffic[J]. Journal of Communications and Networks, 2016, 18(5): 826-836.

[42] Abbessi W, Nabli H. General approach for video traffic: from modeling to optimization[J]. Multimedia Systems, 2018: 1-17.

[43] Lin Y T, Bonald T, Elayoubi S E. Flow-level traffic model for adaptive streaming services in mobile networks[J]. Computer Networks, 2018, 137: 1-16.

[44] Ramos-Muñoz J J, Prados-Garzon J, Ameigeiras P, et al. Characteristics of mobile youtube traffic[J]. IEEE Wireless Communications, 2014, 21(1): 18-25.

[45] 林闯, 胡杰, 孔祥震. 用户体验质量(QoE)的模型与评价方法综述[J]. 计算机学报, 2012, 35(1):1-15.

[46] Schatz R, Hoßfeld T, Casas P. Passive youtube QoE monitoring for ISPs[C]//2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing. IEEE, 2012: 358-364.

[47] Huang T Y, Johari R, McKeown N, et al. A buffer-based approach to rate adaptation: Evidence from a large video streaming service[J]. ACM SIGCOMM Computer Communication Review, 2015, 44(4): 187-198.

[48] Spiteri K, Urgaonkar R, Sitaraman R K. BOLA: Near-optimal bitrate adaptation for online videos[C]//IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications. IEEE, 2016: 1-9.

[49] Akhtar Z, Nam Y S, Govindan R, et al. Oboe: auto-tuning video ABR algorithms to network conditions[C]//Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication. ACM, 2018: 44-58.

[50] Jiang J, Sekar V, Zhang H. Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive[J]. IEEE/ACM Transactions on Networking (ToN), 2014, 22(1): 326-340.

[51] Yi S, Yin X, Jiang J, et al. CS2P:Improving Video Bitrate Selection and Adaptation with Data-Driven Throughput Prediction[C]// Conference on Acm Sigcomm Conference. 2016.

[52] Tian G, Liu Y. Towards agile and smooth video adaptation in dynamic HTTP streaming[C]//Proceedings of the 8th international conference on Emerging networking experiments and technologies. ACM, 2012: 109-120.

[53] Williams R J, Peng J. Function optimization using connectionist reinforcement learning algorithms[J]. Connection Science, 1991, 3(3): 241-268.

[54] Yin X, Jindal A, Sekar V, et al. A control-theoretic approach for dynamic adaptive video streaming over HTTP[C]//ACM SIGCOMM Computer Communication Review. ACM, 2015, 45(4): 325-338.

[55] DASH Industry Forum. https://github.com/Dash-Industry-Forum/dash.js

[56] Spiteri K, Sitaraman R, Sparacio D. From theory to practice: Improving bitrate adaptation in the DASH reference player[C]//Proceedings of the 9th ACM Multimedia Systems Conference. ACM, 2018: 123-137.

[57] 贾澎涛, 何华灿, 刘丽, et al. 时间序列数据挖掘综述[J]. 计算机应用研究, 2007, 24(11):15-18.

[58] 于凡格. 缓冲区综合状态特性对流媒体 QoE 的影响及其在资源分配方面的应用[D]. 2015.

[59] 林高全, 潘吴斌, 程光, et al. 移动网络 YouTube 视频码率及分辨率识别方法[J]. 计算机工程与应用, 2017(15).

[60] 徐健. 面向移动网络 YouTube 加密流量码率及分辨率识别方案研究[D]. 2017.

作者简历及攻读硕士学位期间取得的研究成果

一、作者简历

张加林，男，1992 年 11 月生；

2011 年 9 月至 2015 年 7 月，就读于南通大学电子信息学院电子信息工程专业，取得工学学士学位；

2016 年 9 月至 2019 年 7 月，就读于北京交通大学电子信息工程学院通信与信息系统专业，取得工学硕士学位。

二、发表论文

[1] Jialin Zhang, Hongyun Zheng, Yongxiang Zhao and Yuchun Guo. "Bystander: QoE Perception for Dynamic Video Streaming from Encrypted Traffic." IEEE International Conference on Consumer Electronics-taiwan 2019.

三、参与科研项目

[1] 北京市交通拥堵收费政策舆情分析

[2] 基于大数据的提案系统及其实施效果相关舆情演进分析

独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作和取得的研究成果，除了文中特别加以标注和致谢之处外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京交通大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

学位论文作者签名：张加林 签字日期：2019 年 6 月 3 日

学位论文数据集

关键词*	密级*	中图分类号	UDC	论文资助
QoE；HAS 视频流；ABR；运营商；加密；时序数据挖掘；块序列	公开			
学位授予单位名称*		学位授予单位代码*	学位类别*	学位级别*
北京交通大学		10004	工学	硕士
论文题名*		并列题名		论文语种*
视频用户 QoE 的感知和预测				中文
作者姓名*	张加林		学号*	16120164
培养单位名称*		培养单位代码*	培养单位地址	邮编
北京交通大学		10004	北京市海淀区西直门外上园村 3 号	100044
学科专业*		研究方向*	学制*	学位授予年*
通信与信息系统		信息网络	3	2019
论文提交日期*	2019.06.03			
导师姓名*	郑宏云		职称*	副教授
评阅人	答辩委员会主席*		答辩委员会成员	
	郭宇春		赵永祥；李纯喜；陈一帅；张立军	
电子版论文提交格式 文本（ ） 图像（ ） 视频（ ） 音频（ ） 多媒体（ ） 其他（ ） 推荐格式：application/msword；application/pdf				
电子版论文出版（发布）者		电子版论文出版（发布）地		权限声明
论文总页数*	76			
共 33 项，其中带*为必填数据，为 21 项。				