高清视频点播-AI让你看片更丝滑

腾讯音视频实验室 今天

在线“看片”时，我们经常会遇到这些事情：视频画面突然卡住进入缓冲状态或者视频画面突然变得模糊而不忍直视。这些事情的背后很可能是网络环境突然变差了导致下载速度很慢，也可能是码率调整算法没有对当前环境做出合理的决策导致。

事实上，如何感知网络环境的变化并作出合理的码率调整并非易事。目前很多视频播放的客户端都提供了几种码率档位（标清、高清、超清、蓝光等）供用户自主选择，在网络环境好时用户可以自主切到高码率档位，网络环境差时切到低码率档位。当然，有些主流的视频播放客户端也提供了自适应（自动）这个选项，比如YouTube，当用户选择这个选项后，运行在背后的码率自适应算法会根据当前的网络情况和播放缓冲区等信息去自适应调整视频档位，旨在给用户提供更好的视频观看体验。事实上，码率自适应算法是学术界近年来的一个研究热点，音视频实验室和企鹅电竞团队也在点播码率自适应方法上进行了尝试和实践。

一、码率自适应技术简介

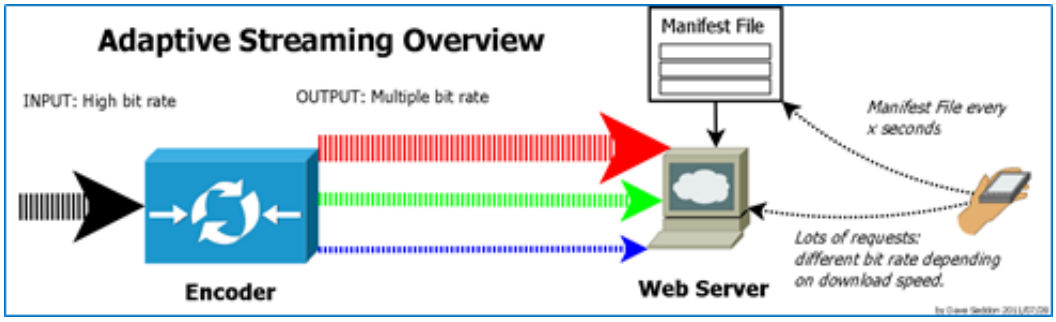

图1：码率自适应系统框架

码率自适应技术 (Adaptive Bitrate Streaming，ABR)是一种视频码率可以根据网络状况或客户端播放buffer情况自动调整的视频传输技术。如图1所示，一个视频源通过视频转码器转成不同的视频码率存储在web 服务器，同时每个码率的视频被切割成一个个小的视频分片，每个分片通常是可单独解码播放的视频内容，分片时长通常介于2秒到10秒之间。视频播放客户端首先获取不同码率的切片索引信息，然后根据当前的网络状况或者客户端的播放缓冲区情况自动选择档位最匹

配的视频片段下载。

目前，基于HTTP的码率自适应技术的实现方式从标准的类型来看主要有两大类：如图2所示，一类是企业方案，即提供了整体的技术解决方案，如Apple Live Streaming技术；另一类是一些国际标准组制定的技术标准，如MPEG的DASH(Dynamic Adaptive Streamingover HTTP)。

按标准的类型分类	具体技术
企业方案	Apple Live Streaming技术
	Adobe Dynamic Streaming技术
	Microsoft SmoothStreaming技术
国际标准组指定的技术标准	OIPF的 HTTP Adaptive Streaming
	MPEG的DASH(Dynamic Adaptive Streaming over HTTP)
	IETF的草案（由Apple公司提议的草案）

图2: 码率自适应技术分类

1.1 码率自适应算法的难点

视频码率自适应的目的是为了提高（或者最大化）用户在线观看视频的体验质量（qualityof experience, QoE），但是用户体验质量的定量表达本身也是一个难点。许多研究表明视频的质量（比如码率）、卡顿时间以及切换频繁度都将影响到用户的体验质量。最大化用户体验质量可以认为：尽可能最大化视频码率的同时尽量减少视频卡顿和码率档位切换。而实际上，组成用户体验质量的这些因素之间是互相影响甚至存在矛盾，比如用户如果能长期收到高码率的视频，视频质量自然会更好，但这又很可能增加视频卡顿的风险（网络如果变差，播放速度快于后续片段的下载速度）。另外，码率自适应调整具有累积效应，前面的码率决策会影响到未来的码率的决策。因此，一个好的码率自适应算法要兼顾各项指标，快速响应环境变化做出尽可能最优的决策。

1.2 码率自适应算法的分类

码率自适应算法（ABR）一般通过当前的网络状态或客户端播放缓冲区情况来动态地调整未来视频片段的码率档位以期最大化用户的QoE，近年来的ABR算法主要分为三类：

- Rate-based: 基于预测的吞吐量去决策下一片段的码率档位，例如FESTIVE，这类方法的主要思想是通过历史视频分片下载期间的网络状况来预测未来的网络带宽，进而驱动视频码率决策，例如预测带宽高时选择高码率视频，预测带宽低时选择低码率视频。

- Buffer-based: 基于客户端的播放缓冲区buffer情况决策下一片段的码率档位，例如BBA、BOLA。
- Hybrid: 混合模式，同时考虑预测吞吐量和buffer信息决策下一片段的码率档位，例如MPC

传统的码率自适应算法一般是基于人为设定的固定规则的来进行视频码率档位的动态调整。其中，单独基于buffer或者预测带宽的方法没有充分利用可用信息，而结合了两者信息的算法可以获得相对较好的效果，但其非常依赖于带宽预测信息，在带宽变化剧烈的场景中，准确预测带宽是非常困难的。基于混合模式的MPC（模型预测控制）算法虽然可以获得不错的结果，但其在求解最优解时计算耗时大，速度慢，在决策动作空间和优化区间步长稍大时，缺点显现出来。

反观码率自适应过程：首先，视频播放客户端会根据当前的网络情况、播放缓冲区大小等因素决定下一个片段的码率档位，然后客户端执行码率决策向CDN服务器请求对应质量的视频片段，进行下载，下载的过程中视频缓冲区也同时在播放消耗。下载完当前片段后，客户端进入了另外一种状态（播放缓冲区大小变化、是否卡顿等）且可以据此评价上一个动作的好坏，然后再重新进行新的决策，如此循环往复，直到结束。以上过程可以抽象成系统控制或者策略制定的问题。而相对于人为制定策略和控制规则而言，机器学习里的强化学习不需要人为地设定一些经验规则或策略逻辑，直接通过与环境的不断交互中去学习策略，在不断试错和经验总结中学到好的决策和控制模型。本文在研读业界近年来码率自适应算法的基础上，重点对基于强化学习的码率自适应算法进行了探索和研究。再进一步讨论之前，下面先简单介绍强化学习的基本概念。

二、强化学习简介

总所周知，近年来机器学习里的深度学习在计算机视觉、语音识别和自然语言处理等流域遍地开花，取得了很多突破性的进展。而机器学习从学习信号区分的话，可以分为：有监督学习、无监督学习和强化学习。强化学习是系统从环境学习以使得奖励期望最大的机器学习。强化学习和有监督学习的不同在于教师信号。强化学习的教师信号是动作的奖励，有监督学习的教师信号是正确的动作。举个例子，在猫狗分类时，对于一张图片是哪种动物，有监督学习给的信号是这张图片的真正label是猫或者狗。而基于强化学习的围棋AI中，在某种局面下，我们并没有确切的答案告诉你，一定要落子在哪里，而是可能会反馈给你落子在何处时的价值估计。

图3: 强化学习基本框架

强化学习（Reinforcement learning, RL）的基本思路是通过最大化智能体（Agent）从环境中获得的累计奖赏值，以学习到完成目标的最优策略。强化学习侧重于学习解决问题的策略，是制定策略或者系统控制的通用框架，其通过和环境的不断交互和动作尝试来调整和优化策略决策，一般由智能体Agent、环境Environment、动作action、执行动作后环境反馈的观察状态S和立即奖惩reward组成，其学习的过程可以描述为：（1）在每个时刻，agent与环境交互得到一个环境的状态观察St，并利用DL（深度学习）等方法来感知观察，以得到状态S的抽象特征表示；（2）基于某种策略将当前状态映射为相应的动作，并基于期望回报来评价各动作的价值（3）环境对此动作做出反应，得到新的观察状态St+1并反馈上一个动作at的立即奖赏reward(r_t)。通过不断循环以上过程，不断试错和学习，以期得到实现目标的最优策略。

三、基于Actor-Critic框架的码率自适应算法

近两年，随着深度学习的火热和应用领域的延伸，学术界也尝试利用机器学习的方法来解决码率自适应问题。其中，MIT的计算机科学和人工智能实验室在基于AI的码率自适应算法上做了创新性的尝试并取得了不错的结果，受到他们思路的启发，我们也在基于AI的码率自适应算法进行了实践探索。本文基于强化学习的码率自适应算法中采用的是Actor-Critic的策略梯度方法来进行策略的学习，下图4是基于Actor-Critic框架的码率自适应算法的模型框架。

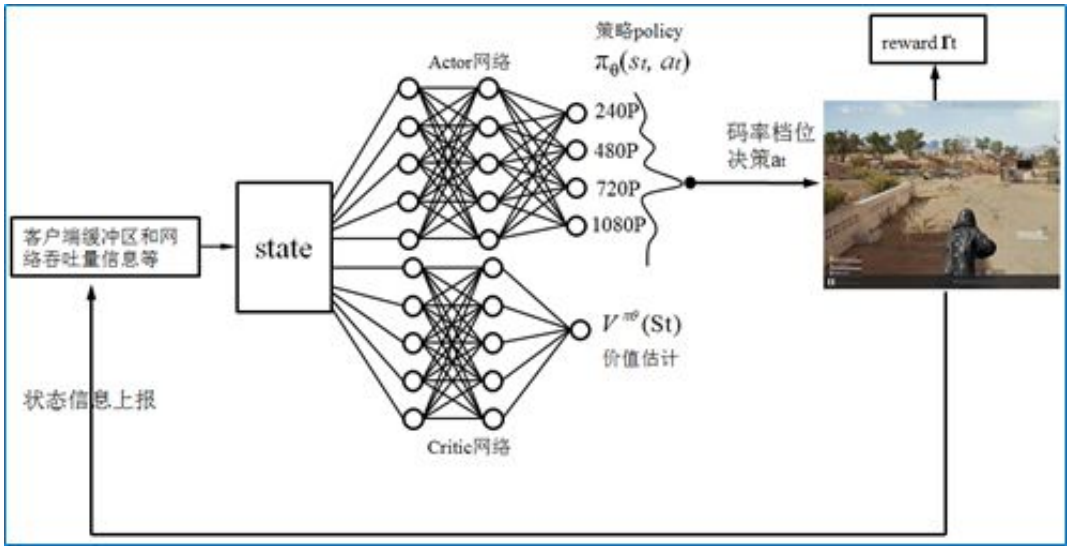

图4：基于Actor-Critic框架的ABR算法模型

在AC框架中，actor（演员）网络进行策略决策，学习状态state到动作的映射关系，而critic（评论家）负责价值估计，即估计价值函数。策略决策和价值估计需要在训练过程中迭代优化，一开始actor的策略可能随机，critic也没有好的打分规则。但是由于reward的存在，critic评分随着训练的进行会越来越准，actor的决策表现也会越来越好，最终学习到好的策略。

在模型训练过程中， reward信号的设计是非常重要的，在不同reward下模型学习到的策略也就会有差异。本文希望解决的问题是如何在点播系统中最大化视频观看用户的体验质量（QoE），而研究表明：视频的质量(码率高低)、rebuffering（重新缓冲）以及档位切换的平滑性都将影响到用户体验的质量，因此在设计reward时，可以由这些因素共同决定。

模型输入的状态信息也是非常重要的点，结合reward的设计，模型状态输入信息主要包括：吞吐量信息、播放缓冲区信息等。

如图4所示，输入状态经过actor网络和critic网络后输出的分别是策略动作（码率档位）和价值估计，而在实际部署只需要策略决策时，仅保留训练好的actor网络。在实际的模型训练时，采用的是A3C（Asynchronous AdvantageActor-Critic）架构进行快速有效的模型训练。

四、预研结果和分析

模型训练和测试的吞吐量数据由宽带网络数据和移动（3G\4G）网络数据组成，训练集和测试集均包含了100多种网络数据。图5和图6展示了在该测试集上

基于强化学习的模型1和模型2对比传统方法MPC(采用表现较好的robustMPC)的结果。总平均reward代表了在100多种网络情况下，视频播放过程中客户端的累计奖励值的平均。模型1和模型2相对于robustMPC分别有约6.4%和8.8%的提升。

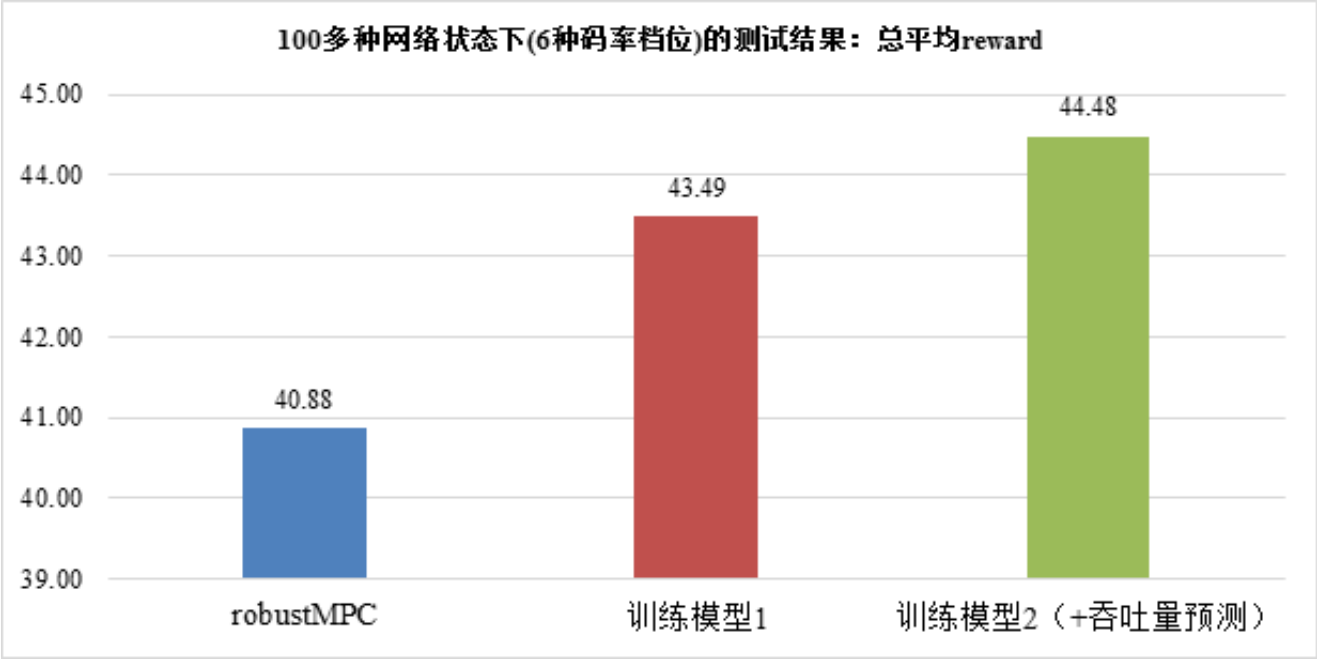

图5：100多种网络状态下(6种码率档位)的测试结果对比

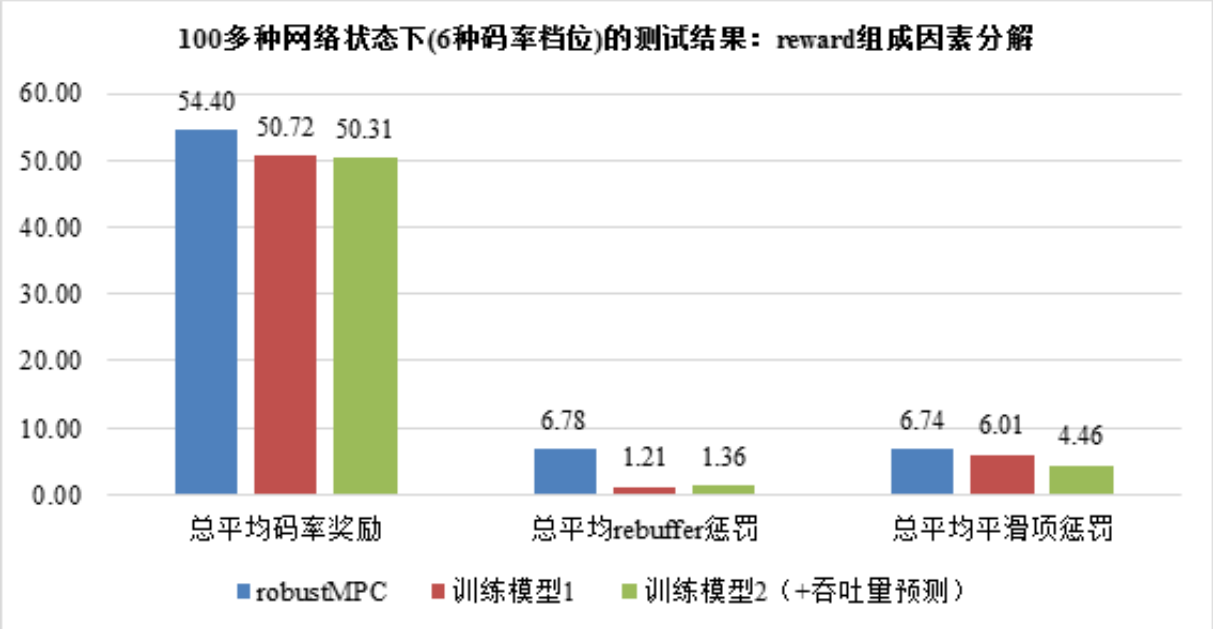

图6：100多种网络状态下(6种码率档位)的测试结果对比(reward组成因素分解)

图6中对组成总平均reward的各个因素进行分解，可以看到传统方法的robustMPC虽然在总平均码率奖励上比强化学习模型1和2要高，但与此同时带来的重新缓冲（rebuffering）的风险也更高，导致rebuffering的惩罚明显多于其他两种模型。而基于强化学习的ABR模型1和2能够更好地兼顾各项因素使得累计奖励到达更高。

图5和图6的结果都是基于数据库的带宽信息进行测试的，为了验证在实际场景中的应用效果，我们将强化学习的码率自适应算法应用于DASH点播系统中，在真实的网络损伤环境下验证各种算法模型的表现：在视频播放客户端和视频服务器之间的视频下载的网络链路上分别添加了限速、丢包、抖动和时延等损伤环境进行测试，结果如图7所示。相对而言，在真实的损伤场景中，模型1和模型2的整体表现比robustMPC稳定，在损伤场景的测试集上，模型1和模型2相对于robustMPC分别有约9.2%和8.9%的提升。结果和需要注意的是，我们的模型是离线使用数据库中有限的带宽数据进行模拟训练的，将其直接应用在实际环境中的表现说明了其具有不错的泛化性。

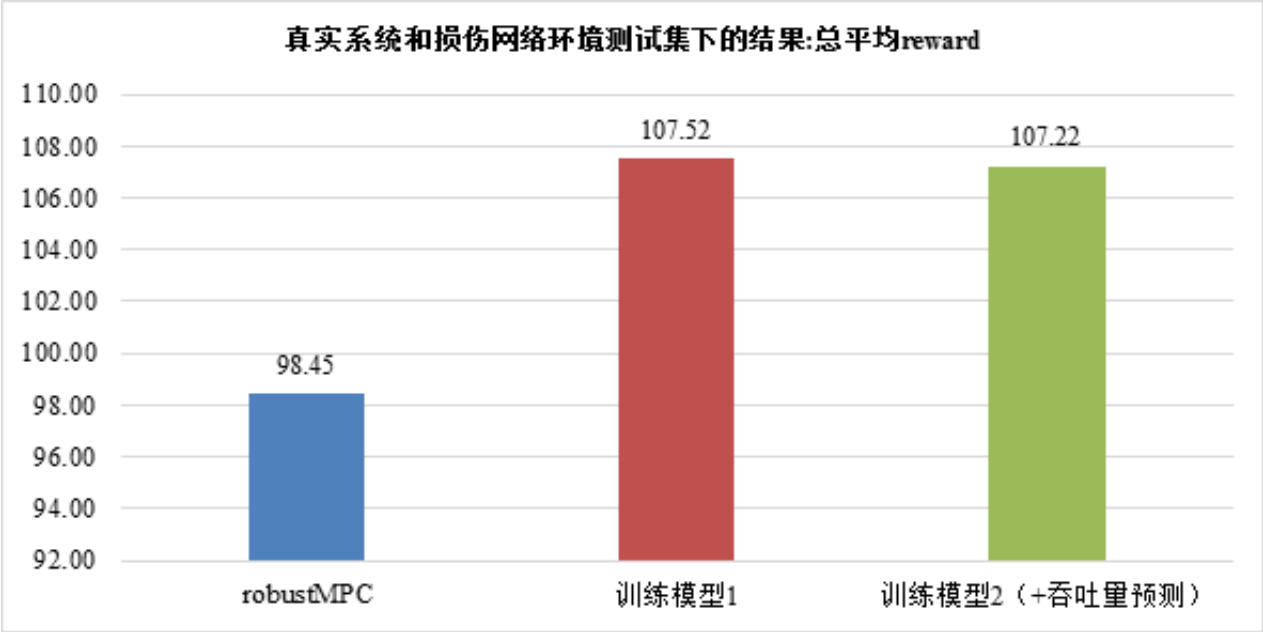

图7：真实系统和损伤网络环境测试集下模型的测试结果

五、线上实践和效果

在上述预研的基础上，我们将本文介绍的基于强化学习的码率自适应算法应用于企鹅电竞的点播业务（HTTP+HLS），用于决策客户端当前应该下载的视频片段的档位（例标清、高清、超清等）。为了提升用户体验，电竞团队联合腾讯云团队做了"帧对齐"的优化工作，实现点播过程的无缝切换效果。图8是针对电竞点播业务部署的点播AI流控系统，预测服务器负责码率决策和下发档位，并向训练服推送训练数据，训练服负责模型训练和模型同步。

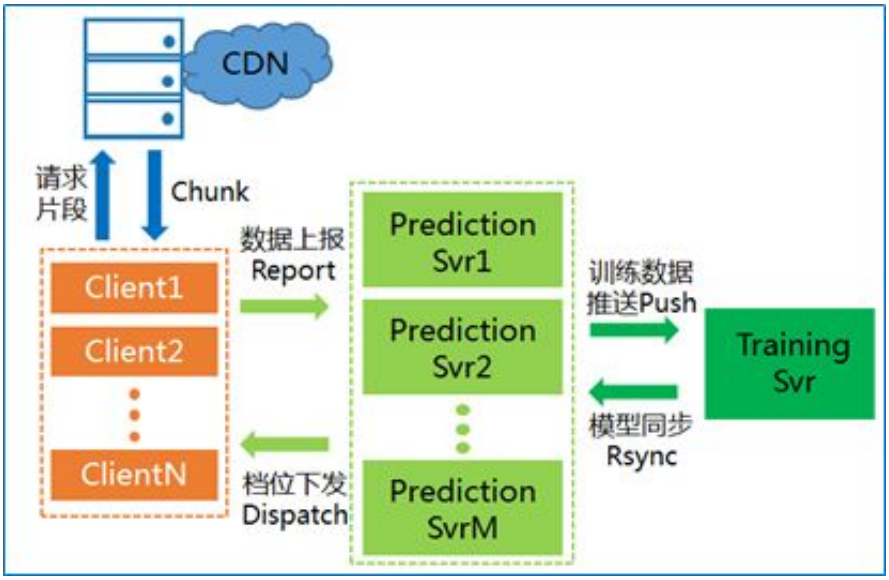

图8：点播AI流控系统

实际的线上数据统计对比如下：

音视频后台打分统计	点播流控方法	得分统计				视频档位统计		
		总得分	码率得分	卡顿惩罚分	切换惩罚分	标清档位占比	高清档位占比	超清档位占比
iPhone	强化学习AI模型	2.4499	2.642	0.1591	0.033	2%	4%	94%
	传统MPC算法	2.3647	2.588	0.1763	0.047	4%	5%	91%
Android	强化学习AI模型	2.486	2.654	0.129	0.039	2%	3%	95%
	传统MPC算法	2.3371	2.554	0.1419	0.075	4%	6%	90%
合计	强化学习AI模型	2.4817	2.653	0.1333	0.038	2%	3%	95%
	传统MPC算法	2.3368	2.556	0.1462	0.073	4%	6%	90%

表1：流控后台打分算法统计的效果对比（对比传统算法）

音视频后台打分统计	点播流控方法	得分统计				视频档位统计		
		总得分	码率得分	卡顿惩罚分	切换惩罚分	标清档位占比	高清档位占比	超清档位占比
iPhone	强化学习AI模型	2.44	2.638	0.164	0.034	2%	3%	95%
	非流控模式	1.733	2.387	0.589	0.065	9%	10%	81%
Android	强化学习AI模型	2.448	2.629	0.138	0.043	2%	4%	94%
	非流控模式	2.073	2.507	0.335	0.099	6%	6%	88%
合计	强化学习AI模型	2.447	2.63	0.141	0.042	2%	4%	94%
	非流控模式	2.068	2.506	0.339	0.099	6%	6%	88%

表2：流控后台打分算法统计的效果对比（对比非流控模式）

以上表格1和表格2结果显示:

- AI模型的总得分比传统MPC算法增加约6%, 超清档位占比增加约5%，且卡顿和切换惩罚更小
- AI模型的总得分比非流控模式增加约18%，超清档位占比增加约6%~14%左右，且卡顿和切换惩罚明显更小

注：表格2是关闭MPC传统算法后，AI模型算法对比非流控模式(用户自主选择档位)的统计数据。

点播客户端质量统计	算法分类	加权总分	首帧缓冲时间	二次缓冲时长	无缓冲率	二次缓冲每小时比率	错误数每小时比率	标清播放时间	高清播放时间	超清播放时间	清晰度得分
5.20	非AI流控	87.29	1.29 (95.89分)	1.09 (99.9分)	0.82 (64.46分)	3.70 (67.23分)	0.26 (96.47分)	7.80%	4.28%	87.92%	96.03
	AI流控	91.34	0.92 (98.13分)	0.48 (100分)	0.90 (75.92分)	3.08 (72.32分)	0.23 (98.32分)	3.69%	1.79%	94.52%	98.17
5.19	非AI流控	86.44	1.30 (95.78分)	1.13 (99.90分)	0.82 (64.81分)	4.34 (63.14分)	0.30 (94.53分)	8.37%	5.30%	86.33%	95.59
	AI流控	91.78	0.89 (98.30分)	0.42 (100.0分)	0.91 (77.58分)	3.02 (72.83分)	0.23 (98.22分)	3.37%	1.46%	95.18%	98.36
5.18	非AI流控	85.99	1.32 (95.67分)	1.16 (99.86分)	0.81 (63.88分)	4.42 (62.69分)	0.31 (94.24分)	10.00%	4.70%	85.30%	95.06
	AI流控	91.49	0.91 (98.21分)	0.46 (100.0分)	0.90 (76.78分)	3.07 (72.36分)	0.24 (97.95分)	3.45%	1.55%	95.00%	98.31
合计	非AI流控	86.63	1.30 (95.80分)	1.12 (99.90分)	0.82 (64.43分)	4.10 (64.50分)	0.29 (95.21分)	8.65%	4.79%	86.56%	95.58
	AI流控	91.55	0.90 (98.22分)	0.45 (100.0分)	0.90 (76.79分)	3.05 (72.52分)	0.23 (98.18分)	3.48%	1.58%	94.93%	98.29

表3：电竞客户端打分算法统计的效果对比

以上表格3结果显示:

- AI流控的总分比非流控模式增加约4.9分
- 清晰度：超清档位占比增加约8.4%
- 流畅度:

1）AI流控较非AI无缓冲率得分高约12分，无缓冲率高约8%

2）AI流控较非AI二次缓冲每小时比率的得分高约8分

上述结果表明，基于强化学习的AI算法在点播流控的应用中，能够更好地兼顾码率、卡顿和切换因素，在提供更高清晰度体验的同时，也能更好地避免卡顿的产生，从而提供给用户更好的视频观看体验。

六、小结和展望

本文简要介绍了基于强化学习的码率自适应算法，在实践预研验证和分析的基础上，将该AI算法模型应用于实际项目。在音视频实验室和企鹅电竞团队等的共同努力下，在基于AI的点播流控探索和实践上，取得了初步的成效。而如何在直播、实时通话系统中进行更好的码率自适应调整值得我们进一步研究和探索。