基于深度强化学习的码率自适应算法研究

易 令,李泽平
(贵州大学计算机科学与技术学院,贵州贵阳550025)

摘 要: 码率自适应(Adaptive BitRate,ABR)算法是视频客户端提高用户体验质量(Quality of Experience,QoE)的一种有效途径.针对现有ABR算法存在频繁缓冲、视频卡顿、画质较低和网络吞吐量预测不准确等问题,本文提出一种基于深度强化学习的码率自适应(Deep Reinforcement Learning based ABR,DRLA)算法.DRLA用实际网络带宽数据训练神经网络,通过收集客户端缓冲区占用率和网络吞吐量向视频服务器请求最佳码率的视频.首先,DRLA用基线函数方法优化损失函数L,用熵随机探索方法防止损失函数局部收敛;其次利用约束条件限制新旧策略的散度更新幅度提高算法的鲁棒性;最后通过置信域(trust region)优化找到最优策略,使得QoE达到最优.与现有ABR算法对比的实验结果表明:DRLA减少了训练时间,能进一步提高算法的鲁棒性和用户的QoE,并在实际环境下验证了算法的有效性.

关键词: 码率自适应算法;体验质量;深度强化学习;基线函数;熵;置信域
中图分类号: TP393 **文献标识码:** A **文章编号:** 0372-2112(2022)05-1192-09
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20210487

Research of Adaptive Bitrate Algorithm Based on Deep Reinforcement Learning

YI Ling, LI Ze-ping
(*School of Computer Science and Technology, Guizhou University, Guiyang, Guizhou* 550025, *China*)

Abstract: Modern video players employ adaptive bitrate(ABR) algorithms to improve user quality of experience (QoE). Aiming at the problems of the existing ABR algorithms, for example, these algorithms usually lead to frequent re-buffering, video freezes, low video quality, or inaccurate network throughput prediction. In this paper, we propose a deep reinforcement learning algorithm based on ABR(DRLA). DRLA trains the neural network with the actual network bandwidth data, and requests the video with the best bit rate from the video server by collecting the client buffer occupancy rate and network throughput. DRLA optimizes the loss function with the baseline function method. To encourage exploration, we add an entropy regularization term to the update rule of the policy network. Then, DRLA uses constraints to limit the divergence of the new and old policies. Besides, DRLA optimizes the policy to use trust region to improve QoE. Compared with the existing ABR algorithms on the QoE metrics, DRLA reduces training time, is more robust, and can further improve QoE, and the experimental results verify the effectiveness of this algorithm.

Key words: adaptive bitrate algorithm; quality of experience; deep reinforcement learning; baseline function; entropy; trust region

1 引言

随着视频观众不断增加,视频流量呈上升趋势,预计未来几年内视频流量将占据互联网流量的80%以上[1].研究表明[2]:观众会因视频启动缓慢、播放码率较低或经常卡顿等问题而缩短观看时长,甚至放弃观看视频,从而减少视频供应商创收的机会.用户希望观看到高质量的视频,视频质量可以通过视频编码的码率来量化.当视频以较高的码率呈现时,用户观看更加投入,观看时间更长[3].然而,由于网络带宽限制,用户并不能够持续观看高码率的视频.如果选择高于可用网络带宽的码率将导致视频在播放过程中重新缓冲,因为视频的播放速率超过了视频的下载速率.

收稿日期:2021-04-16;修回日期:2022-01-27;责任编辑:覃怀银
基金项目:国家自然科学基金(No.61462014)

基于HTTP(HyperText Transfer Protocol)的动态自适应流(Dynamic Adaptive Streaming over HTTP, DASH)[4]是目前视频在线传输的主要形式. 视频文件被切分为不同码率的视频块存储在服务器上,视频客户端的ABR(Adaptive BitRate)算法根据缓冲区占用率和网络吞吐量向服务器请求最优码率的视频,服务器根据请求的码率找到对应的视频块发送给视频客户端.

针对ABR算法Pensieve[5]存在训练时间长、不稳定、收敛困难和不能请求最优码率等问题,本文提出DRLA,一种基于深度强化学习的码率自适应算法. DRLA利用基线(baseline)函数减少因不同奖励值造成的策略梯度方差,加速收敛;同时,通过限制新旧策略的更新幅度,避免了因新旧策略差异过大而造成收敛困难,提升了算法的鲁棒性;最后采用置信域方法优化策略,进一步提升ABR算法的性能.

2　相关工作

图1是ABR算法工作原理:ABR算法根据客户端当前缓冲区占用率和网络吞吐量通过HTTP协议向视频服务器请求最优码率的视频,服务器根据请求找到对应码率的视频发送给播放器. ABR算法需优化QoE(Quality of Experience):(1)最小化重新缓冲事件,防止缓冲区为空导致没有视频播放;(2)尽可能请求高码率的视频;(3)维持缓冲区稳定,尽量避免频繁缓冲或频繁码率切换.

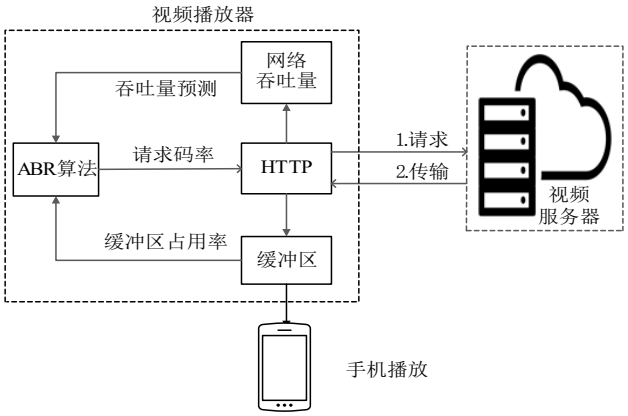


图1　ABR算法工作原理

ABR算法目前主要分为3类[5]:基于传统启发式、基于强化学习和基于深度强化方法. 基于传统启发式的ABR算法研究进展:Spiteri等[6]使用李雅普诺夫算法优化缓冲区占用率(Buffer Occupancy based Lyapunov Algorithm, BOLA),目的是稳定缓冲区容量以减少重新缓冲时间,提高视频质量. Sun等[7]根据视频块的下载情况(Cross Session Stateful Predictor, CS2P)采用马尔科夫模型来预测网络吞吐量,在网络带宽增加的情况下,CS2P将会请求更高码率的视频. Festive算法[8]根据过去几个视频块的网络吞吐量下载情况,使用谐波均值方法预测码率,将预测的码率限制在一定范围内. Yin等[9]应用控制论方法(Model Predictive Control, MPC)结合缓冲区占用率和网络吞吐量预测码率,然而该算法严重依赖于网络吞吐量的准确预测. 由于网络吞吐量是实时变化的,因此MPC很难准确预测网络吞吐量.

基于强化学习的ABR算法研究进展:Claeys等[10]采用强化学习Q-learning预测码率,该算法以表格的形式存储和学习状态价值函数,而不是采用神经网络近似方法;但在真实的网络条件下存在大量的状态与动作空间,由于表格的存储有限,因此不适应真实的网络带宽条件. Mao等[11]提出了基于强化学习的ABR算法(Reinforcement Learning based ABR, ABRL),利用贝叶斯[12]优化QoE的各项指标,通过训练线性策略来减少视频客户端与后台模拟环境之间的时延;但线性函数会导致ABR算法性能下降.

基于深度强化学习的ABR算法研究进展:Pensieve[5]使用异步优势方法[13]训练神经网络生成ABR算法,比简单固定的启发式ABR算法提升了QoE. 但Pensieve在训练时存在收敛慢和不稳定等不足,造成训练难以获得最优的ABR算法模型. Lekharu等[14]采用长短记忆(Long Short-Term Memory)网络替换卷积网络进行时序特征提取,从而提高QoE;但该方法不能有效地学习基线函数,导致训练效率低. Gadaleta等[15]提出了一种Deep Q-learning的ABR算法,与Actor-Critic方法相比,具有更快的收敛速度;但Q学习的最大化策略函数会造成高估问题,导致预测码率不准确. 针对现有方法没有考虑优化多用户偏好QoE问题,因此Huo等[16]提出了基于元学习的ABR算法,将元学习与多任务深度强化学习相结合,能够优化不同用户的多样化QoE.

3　问题建模

强化学习(Reinforce Learning, RL)是Agent与环境交互以获得最大累计奖励的学习方法. Agent从环境中观察到状态s_t(视频码率R_n、缓冲区占用率B_t和网络吞吐量C_t)并采取动作a_t(视频码率R_n)给环境,环境反馈给Agent奖励r_t,且状态转移到s_{t+1}. 整个过程遵循马尔科夫状态转移(Markov Decision Process, MDP),MDP用元组$<s_t, a_t, r_t>$来描述. 其中$s_t = \{R_n, B_t, C_t\}$,有关RL的各项参数定义如下:

输入(状态s_t)　神经网络的输入由视频码率R_n、缓冲区占用率B_t和网络吞吐量C_t三部分组成. 详细描述

如下：

（1）视频码率（R_n）：视频V由N个连续的视频块组成，即$V=\{1,2,\cdots,N\}$，每一个视频块包含M秒的视频并以不同的码率进行编码. 令Z为所有可用视频码率的集合，视频播放器能够选择下载视频块n在码率$R_n\in Z$，令$d_n(R_n)$为视频码率R_n的大小. 在恒定码率的情况下：$d_n(R_n)=M\times R_n$，在不同码率的情况下：$d_n\sim R_n$为不同码率的视频块. 选择的码率R_n越高，用户感知视频的质量$q(R_n)$就越高.

（2）缓冲区占用率（B_t）：视频块从播放器的缓冲区下载时，包含已经下载和剩余的视频块. 令$B_t\in[0,B_{\max}]$为时间t的缓冲区占用率，即缓冲区内剩余视频块的播放时间$B_{\max}$为缓冲区的最大容量，取决于具体的播放器.

（3）网络吞吐量（C_t）：视频块n在时间t的网络吞吐量为C_t. 视频播放器从时间t_n开始下载视频块n，该块的下载时间为$d_n(R_n)/C_n$，取决于所选块的码率R_n和下载过程中的平均下载速率C_n. 一旦块n被完全下载，视频播放器将等待Δt_n并开始下载在时间t_{n+1}的视频块$n+1$.

输出（动作a_t） 在状态空间s_t下采取对应动作a_t可能的概率，输出是一个n维向量. 视频码率R_n对应下载速率C_n，即视频流传输到缓冲区的过程中，播放器根据下载速率C_n选择对应码率R_n，输出为离散的动作空间.

奖励值（r_t） RL的目标是使累积奖励值r_t最大，即使用户感知的视频质量$q(R_n)$、不同块之间的切换频率$|q(R_{n+1})-q(R_n)|$和重新缓冲时间T_n三个指标的加权值QoE最大，QoE线性公式已经被广泛用于评估ABR算法[3,5~11]. 用户观看一段视频由N个连续的视频块组成，因此本文的QoE公式总结为

$$r_t=\mathrm{QoE}^N=\alpha\sum_{n=1}^{N}q\left(R_n\right)-\lambda\sum_{n=1}^{N-1}\left|q\left(R_{n+1}\right)-q\left(R_n\right)\right|-\mu\sum_{n=1}^{N}T_n \quad (1)$$

其中α,λ,μ分别是视频质量，视频质量变化和重新缓冲时间变化对应的非负权重系数. QoE的各项指标之间的变量关系满足文献[9].

4 DRLA算法设计

DRLA算法的目标是使式（1）的QoE值最大，在问题建模的基础上，通过优化策略得到最优策略函数，且端到端训练神经网络生成ABR算法模型. 下面将详细介绍DRLA算法具体优化、模拟环境、神经网络结构设计和训练过程.

4.1 算法优化

对于DRLA算法，存在最优策略π，优于或至少等于其余所有策略. 训练采用策略梯度[17]方法，该方法是使累计期望奖励值最大，优化策略π的参数θ. 通过搜索最佳策略π来训练Agent沿轨迹（动作、状态和奖励）方向移动，该优化过程目的是最大化目标函数$J(\pi)$并求梯度：

$$\nabla J(\pi_\theta)=\mathrm{E}_{s,a\sim\pi_\theta}\left[\nabla\log\pi_\theta(s_t,a_t)Q^{\pi_\theta}(s_t,a_t)\right] \quad (2)$$

其中$Q^{\pi_\theta}(s_t,a_t)$是从状态s_t开始选择动作a_t的期望累计回报，$\nabla\log\pi_\theta(s_t,a_t)$是策略梯度的更新方向. 通过以下四个方法对动作价值函数$Q^{\pi_\theta}(s_t,a_t)$进行优化，使累计奖励值最大：

（1）基线函数方法优化损失函数L

关于损失函数L定义如下：

$$L=(Q^{\pi_\theta}(s_t,a_t)-b_t)^2/2 \quad (3)$$

为减少策略梯度方差，b_t采用输入依赖基线（input-dependent baseline）[18]函数，能够消除外部因素造成的方差，加速收敛并对动作值函数$Q^{\pi_\theta}(s_t,a_t)$使用蒙特卡洛方法近似[19]：$v^{\pi_\theta}(s_t)\approx Q^{\pi_\theta}(s_t,a_t)$，使预测值接近真实值. 利用Reinforce算法[20]估计状态值函数$v^{\pi_\theta}(s_t)$，在时间t，通过环境反馈奖励r_t，$v^{\pi_\theta}(s_t)$更新规则如下：

$$v^{\pi_\theta}(s_t)=\sum_{i=t}^{n}\gamma^{i-t}r_i \quad (4)$$

（2）熵损失函数

为防止L局部收敛到次优确定性策略π，将π的熵H添加到目标函数，鼓励Agent随机探索达到最优策略. 关于熵正则项[13]以及包含策略函数的梯度表示为

$$\nabla J(\theta)=\sum_{t=1}\nabla\log\pi_\theta A(s_t,a_t)+\beta\nabla_\theta H(\pi_\theta) \quad (5)$$

其中超参数β控制熵正则项更新幅度，$\nabla_\theta\log\pi_\theta(s_t,a_t)$指导优势函数$A$的策略梯度更新方向，优势函数$A$是对状态价值函数$Q^{\pi_\theta}(s_t,a_t)$的蒙特卡罗近似. 优势函数定义为

$$A(s_t,a_t)=v^{\pi_\theta}(s_t)-b_t \quad (6)$$

$A(s_t,a_t)$为动作a_t带来的优势，即Agent从环境中观察到奖励r_t与基线b_t之差.

（3）新旧策略的散度

由于策略π在更新过程中具有随机性，如果新策略与旧策略差距过大，损失函数L难以收敛，Agent在探索过程中则会浪费很多时间，导致DRLA算法不能收敛到最优策略. 因此约束新旧策略的KL散度[21]有利于收敛. 表示如下：

$$\mathrm{KL}(\theta,\theta_{\mathrm{old}})=\mathrm{E}_{s\sim\pi_\theta}[\mathrm{KL}(\pi_{\mathrm{old}}\|\pi_\theta)] \quad (7)$$

（4）置信域优化

为了使式（2）策略最优，损失函数L最小[21,22]，由$\underset{\theta}{\mathrm{maximize}}\,\mathrm{E}_{s,a\sim\pi_{\theta_{\mathrm{old}}}}\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\mathrm{old}}}(a_t|s_t)}Q^{\pi_\theta}(s_t,a_t)\right]$，使新旧策略限制在小于等于$\delta$的区域内：

$$\mathrm{E}_{s\sim\pi}[\mathrm{KL}(\pi_{\mathrm{old}}\|\pi_\theta)]\le\delta \tag{8}$$

通过 π_{old} 进行抽样,得

$$\nabla J(\pi_\theta)=\mathrm{E}_{s,a\sim\pi_{\mathrm{old}}}\left[\frac{\pi_\theta}{\pi_{\mathrm{old}}}A(s_t,a_t)\nabla\log\pi_\theta\right] \tag{9}$$

又因为 $\pi_\theta\nabla\log\pi_\theta=\nabla\pi_\theta$,得

$$\nabla J(\pi_\theta)=\mathrm{E}_{s,a\sim\pi_{\mathrm{old}}}\left[\frac{\nabla\pi_\theta}{\pi_{\mathrm{old}}}A(s_t,a_t)\right] \tag{10}$$

该策略梯度对应的目标函数为

$$J(\pi_{\mathrm{old}})=\mathrm{E}_{s,a\sim\pi_{\mathrm{old}}}\left[\frac{\pi_\theta}{\pi_{\mathrm{old}}}A(s_t,a_t)\right] \tag{11}$$

为了约束新旧策略的更新幅度,加入KL散度作为正则项:

$$J(\pi_\theta)=\mathrm{E}_{s,a\sim\pi_{\mathrm{old}}}\left[\frac{\pi_\theta}{\pi_{\mathrm{old}}}A(s_t,a_t)-\beta_1\mathrm{KL}(\pi_{\mathrm{old}}\|\pi_\theta)\right] \tag{12}$$

其中,超参数 β_1 是在实验迭代过程中动态调整,约束KL散度.

Actor网络的参数更新是根据Critic网络进行的,以最大化 $J(\pi_\theta)$. 策略网络参数 θ 更新为

$$\theta\leftarrow\theta_{\mathrm{old}}+\alpha\nabla J(\pi_\theta)+\beta\nabla_\theta H(\pi_\theta) \tag{13}$$

因此得到目标函数 $J(\pi)$ 之后,Agent在训练过程中每次都能请求最优码率,能够让式(1)的QoE最大,DRLA算法最优.

4.2　详细设计

模拟环境　模拟环境表示视频客户端的内置缓冲区,包括缓冲区容量 $B_{\max}$ 和当前缓冲区大小 B_t. 当每个视频块 n 被下载后,模拟环境调用ABR算法,由ABR算法决策下一个视频块的码率 R_n. 对于每一个视频块的下载,模拟环境根据视频块的码率 R_n 和跟踪得到的平均下载速率 C_n 来确定下载时间 $d_n(R_n)/C_n$. 如果下载完当前所有的视频块,将会清空缓冲区置 B_t=0. 如果当前缓冲区的容量 B_t 小于下载时间 $d_n(R_n)/C_n$,将会清空缓冲区并停止下载. 然后,模拟环境将下载视频块的持续时间添加到缓冲区. 当缓冲区容量 B_t 大于设置值 $B_{\max}$ 时,模拟环境则停止下载视频块.

视频流通过缓冲区进行播放,每一个视频块从缓冲区下载后,模拟环境需记录:(1)预测的网络吞吐量 C_{t+1};(2)下载前一个视频块的网络吞吐量 C_t;(3)下载前一个视频块消耗的时间 $d_n(R_n)/C_n$;(4)缓冲区内剩余视频块的数量 N. 根据以上信息预测的码率 R_n 作为输出,输出是一个离散的动作空间.

神经网络结构　DRLA算法的网络结构如图2所示,隐藏层数为1,采用128个卷积核和全连接网络进行特征提取,其中卷积核大小为4,步长为1. 结合了卷积网络的局部感知和全连接网络的全局感知特性有效进行特征提取,从而提高了算法的泛化性和鲁棒性.

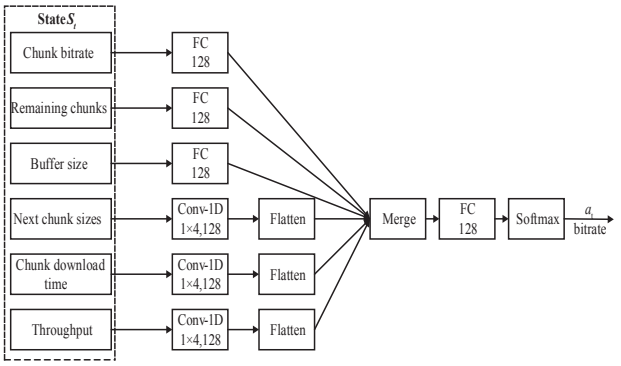


图2　DRLA算法框架

我们做了大量对比实验,通过改变神经网络结构,比较不同神经网络结构对QoE值的影响. 表1首先是固定隐藏层的数量,改变卷积核和隐藏层神经元的数量. 这些参数的使用是并行的,例如神经元的数量为4,卷积核的数量也为4. 神经元和卷积核的数量如表1所示,数量从4增加到128. 当数量超过16时,QoE值逐渐趋向于平稳. 当数量达到128时(DRLA算法的设置值),QoE值和稳定性达到最优.

由表2可知,当神经元和卷积核的数量固定为128,改变隐藏层的数量. 我们发现当隐藏层的数量为1时,性能最好,这也是本文算法所采用的. 随隐藏层数增加,QoE值逐渐减小. 增加隐藏层数花费了大量的训练时间,但没有提高算法的性能.

表1　不同卷积核和神经元的数量对QoE值和算法稳定性的影响

卷积核和神经元的数量	QoE值	稳定性
4	3.749	±1.194
16	4.593	±1.279
32	5.123	±1.310
64	5.499	±1.370
128	**5.495**	**±1.349**

表2　不同隐藏层的数量对QoE值和算法稳定性的影响

隐藏层的数量	QoE值	稳定性
1	**5.495**	**±1.349**
2	5.275	±1.421
3	5.173	±1.310
4	4.972	±1.290
5	4.795	±1.216

训练过程　DRLA算法在训练过程中使用RL actor-critic方法,训练过程如下:

步骤1：状态s_t作为输入，输入状态包括6个变量，分别是网络吞吐量（throughtput）、视频块的下载时间（chunk download time）、下一个视频块的大小（next chunk sizes）、缓冲区的容量（buffer size）、缓冲区剩余视频块的数量（remaining chunks）和视频块的码率（chunk bitrate）.

步骤2：在接收到状态s_t后，Agent基于策略π选择相应的动作a_t，概率分布定义为$\pi:\pi(s_t,a_t)\rightarrow[0,1]$，$\pi(s_t,a_t)$是动作$a_t$在状态$s_t$下可能采取的动作概率. 由于有许多的动作、状态和奖励，因此采用神经网络参数θ来调整所有的策略参数.

步骤3：在采取每个动作a_t之后，环境反馈给Agent对应a_t的奖励r_t，目标是从环境中获得累计奖励最大. 因此，奖励是根据式QoE的各项参数进行设置，反映QoE的各项指标.

步骤4：Agent对码率决策的轨迹进行采样，然后使用经验数据值计算优势函数A作为无偏估计，Actor网络每次更新如下：

$$\theta\leftarrow\theta+\sum_{t=1}\frac{\pi_\theta}{\pi_{\text{old}}}\nabla\log\pi_\theta A(s_t,a_t)+\beta\nabla_\theta H(\pi_\theta) \tag{14}$$

其中β约束熵的更新幅度；策略π_θ更新概率大小取决于优势函数A，优势函数A越大表明带来的奖励值越大，则增加该方向的动作概率.

步骤5：从经验数据中计算优势函数A，需要对状态价值函数$v^{\pi_\theta}(s_t)$进行估计，即从状态s_t开始遵循策略π_θ的累计期望奖励. Critic网络的作用是从经验数据中观察到奖励值r_t并对$v^{\pi_\theta}(s_t)$进行估计，利用Reinforce算法训练Critic网络参数θ_v：

$$\theta_v\leftarrow\theta_v-\alpha\sum_t\nabla_{\theta_v}\left(v^{\pi_\theta}(s_t,\theta_v)-b_t\right)^2 \tag{15}$$

其中α是Critic网络的学习率. 优势函数A表示为$v^{\pi_\theta}(s_t,\theta_v)-b_t$，其中$b_t$为基线函数，$b_t$采用文献[18]的输入依赖（input-dependent）方法，能够孤立奖励函数，消除外部因素产生的方差.

步骤6：最后Actor网络用于ABR算法码率决策，q_t^i为选择视频块i的码率，采用softmax概率分布函数输出下一个视频块的概率值p_i：

$$p_i=\frac{\exp(q_t^i)}{\sum_{i=1}^{n}\exp(q_t^i)} \tag{16}$$

5 实验结果与分析

5.1 实验环境与参数设置

选择2台PC机，配置为Intel Xeon E5-2620 CPU、64 GB内存、64位Ubuntu20.04、64位Windows7操作系统作为实验平台，Python3.5、TensorFlow1.5、Apache2、Google Chrome和FFmpeg等开发工具.

本文使用FCC[23]，HSDPA[24]和4G LTE[25]网络带宽数据集，数据集特征如下：

FCC数据集 是用户在火车、汽车和公交车观看视频所产生，粒度为5 s，超过100万条轨迹吞吐量范围为0~111 Mbit/s.

HSDPA数据集 用户在地铁、电车、火车、公共汽车和渡轮产生的，粒度为1 s，轨迹条数为86，吞吐量范围为0~3 Mbit/s.

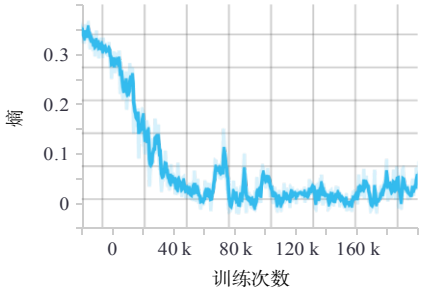
4G LTE数据集 静态、行人、汽车、公共汽车和火车等移动模式，粒度为1 s，持续时间为15 min，轨迹条数为135，吞吐量范围为0~173 Mbit/s.

在本次实验中，选取的网络吞吐量大于0.2 Mbps小于6 Mbps，选取相对较低值的网络吞吐量是为了让训练生成的ABR算法模型更优[5]. 本次实验选取2000条网络轨迹，其中使用80%用于训练，20%用于测试.

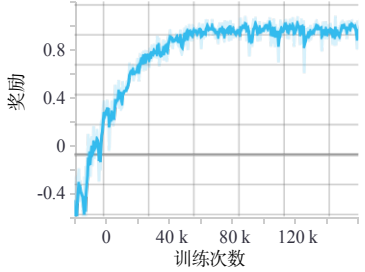
式(1)的QoE参数设置：N为8，α为1，λ为4.3，μ为1. 训练过程中熵因子从1到0.1迭代超过16万轮，每轮大小为100，γ=0.99，使用了Relu激活函数和Adam优化器. 在整个实验过程中，根据损失函数的变化，调整Actor和Critic网络的学习率，分别调整为0.0001和0.001，环境反馈给Agent的奖励值是QoE值.

5.2 结果分析

实验结果使用TensorBoard深度学习工具可视化DRLA与Pensieve训练过程中的熵（entropy）和奖励（reward）值，如图3和4所示.

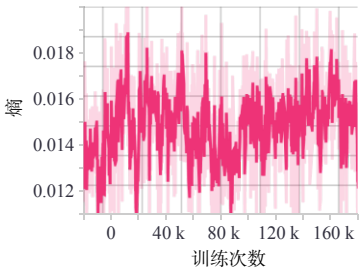


(a) 熵损失值

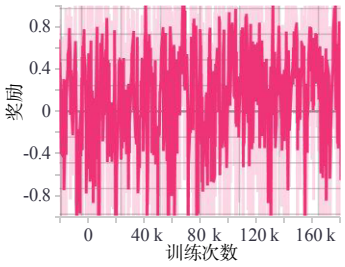


(b) 奖励值

图3 DRLA训练情况



(*a*) 熵损失值



(*b*) 奖励值

图4　Pensieve训练情况

从图3与4对比的奖励值和熵损失函数值可以看出DRLA收敛、稳定. DRLA通过优化ABR策略并限制新旧策略的更新幅度,利用基线函数减少方差,使得累计奖励值不断增加. 而Pensieve由于Agent在探索过程中新旧策略的更新幅度是随机的,没有消除外部因素产生的方差,所以导致奖励值波动很大且不稳定. 从累计奖励值可以看出,DRLA收敛之后平均奖励值是0.861,而Pensieve是0.768,平均奖励值差距是9.3%,表明DRLA获得的期望累计奖励值更高. 由奖励函数的收敛情况可知,DRLA训练效率更高,训练到6万轮收敛,而Pensieve训练到16万轮仍然没有收敛.

5.3　DRLA与现有ABR算法比较

为评估DRLA,比较了DRLA与现有最先进ABR算法的QoE,QoE的表达式为式(1). 实验评估:3类网络带宽数据集(FCC,HSDPA和4G LTE);视频:切分为48个视频块,每个视频块时长约为4 s,共计时长为193 s;H.264/MPEG-4编码:{300,750,1200,1850,2850,4300} kbps,对应视频播放器的显示分辨率为{240,360,480,720,1080,1440}p;视频播放器:Google Chrome;视频服务器:Apache2.

(1) BOLA算法[6]:基于缓冲区占用率的ABR算法,只单独考虑了缓冲区占用率使用李雅普诺夫来优化请求的码率.

(2) ABRL算法[11]:基于强化学习的ABR算法,为了有利于部署和实现,将神经网络ABR策略翻译为线性策略,通过拟合线性模型做出码率决策

CDF(Cumulative Distribution Function)概率分布图能够评估QoE总体变化情况,越往右表明QoE越高. 如图5所示,DRLA的QoE平均高于ABRL的10.3%,BOLA的23.7%. 在开始的一段奖励中,差距在15%以上,表明DRLA算法播放视频的平均质量高、重新缓冲时间少和不同码率之间切换频率少. 在复杂的网络条件下,当网络带宽变得很差时,由于DRLA学会调节缓冲区占用率,通过低码率的视频补偿当前的低带宽,一旦网络带宽增加,就会请求高码率的视频来提升QoE. 但是BOLA只考虑缓冲区占用率,没有充分利用网络带宽信息,则不能根据可用网络带宽请求最佳码率的视频. 而ABRL将神经网络翻译为线性策略,这使得决策时泛化能力下降,则不能适应不同的网络带宽.

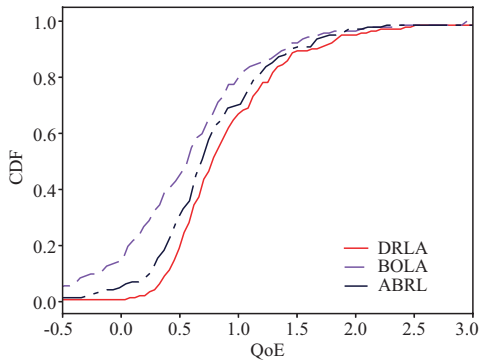


图5　FCC数据集上的QoE概率分布情况

为进一步说明DRLA算法的泛化性,即QoE更高,训练过程中我们也与最先进的ABR算法Comyco[26](基于缓冲区占用率与网络吞吐量的ABR算法,采用深度强化学习模仿学习方法,在实现过程中我们也采取了这种方法进行训练)进行了比较,实验结果如图6所示.

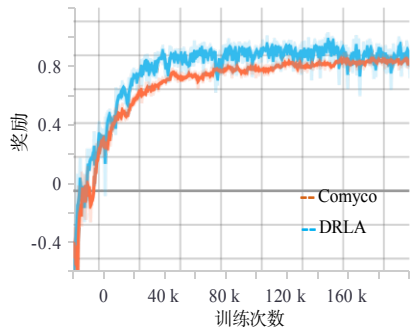


图6　DRLA与Comyco累计奖励值变化情况

由图6可知,DRLA的累计奖励值高于Comyco的3.4%~7.8%,由于DRLA的熵损失函数值H能够逐渐减少,输入依赖基线函数b_t消除方差,有利于Agent向高奖励值方向探索,因此QoE更高.

实验过程中我们也在HSDPA与4G LTE数据集上

进行了测试. 实验结果如图7和图8所示.

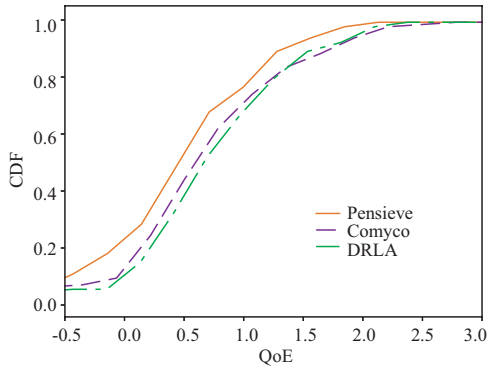


图7 HSDPA 数据集上的QoE概率分布情况

由图7可知,DRLA的QoE比Comyco与Pensieve分别提高了4.6%和17.4%. 在整个QoE概率分布情况中,Pensieve的QoE均低于另外2类算法. 由于Pensieve没有解决因大量不同的网络吞吐量产生的策略梯度方差,因此难以获得较好的ABR策略. DRLA与Comyco利用基线函数解决了方差问题,整体QoE均表现较好. 但DRLA的QoE更高,由于DRLA优化了策略,能适应大量不同的网络吞吐量,在没有遇见的网络吞吐量情况下也能平衡QoE的各项指标,最大化QoE,减少重新缓冲时间与不同码率之间的切换频率.

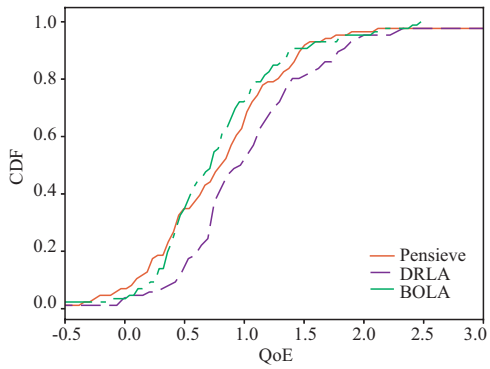


图8 4G LTE 数据集上的QoE概率分布情况

由图8可知,DRLA的QoE比Pensieve与BOLA分别提高了12.4%和16.8%. DRLA在整个概率分布图中均高于另外2类算法,表明在所有ABR策略决策中,DRLA能够平衡视频码率与卡顿时长以最大化QoE. 因为DRLA消除了策略梯度方差且优化了ABR策略,在不同的网络吞吐量条件下,ABR具有泛化性. 而Pensieve没有解决训练过程中因回报值差异产生的方差,BOLA采用传统启发式方法不能适应大量的网络带宽数据集,ABR决策时泛化能力较弱.

5.4 DRLA在实际环境中测试

在5.3章节仿真测试中,训练集与测试集是同一数据集,为进一步说明DRLA算法在实际网络环境下的泛化性,将算法部署在dash.js[27]播放器上,采用Python BaseHTTPServer和SocketServer实现,通过在真实的4G和WiFi网络条件下进行多次测试. 实验过程中,视频客户端与ABR算法均运行在Window7笔记本电脑上,通过HTTP协议访问另外一台Linux视频服务器. 根据图1的ABR算法设计原理,视频播放器向服务器请求码率时首先需经过ABR算法,然后由ABR算法向服务器发出请求视频的信号. 因此,ABR算法与视频客户端之间存在往返时延. 在WiFi网络条件下的平均往返时延为16.67 ms,4G网络条件下的平均往返时延为77.14 ms. 实验过程中DRLA对比了Pensieve和BOLA,在真实的网络条件下,对收集得到的QoE数据集进行归一化处理,实验结果如图9所示.

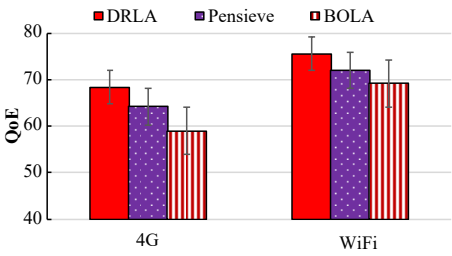


图9 不同网络条件下平均QoE

由图9可知,在WiFi网络条件下的QoE总体高于4G网络,由于WiFi网络带宽相对稳定且往返时延低,能够持续请求高码率视频,因此QoE较高. DRLA在4G和WiFi网络条件下,QoE分别提升了3.8%~9.4%和5.2%~8.5%. 在模拟环境中训练生成的DRLA算法具有较强的泛化性,在未知的网络条件下也能最大化视频质量并减少卡顿时间.

6 结论

本文提出的DRLA算法,利用基线减少方差,通过优化策略得到最优策略,与现有的ABR算法进行了比较. 实验结果表明:DRLA能够减少训练时间,鲁棒性更强;QoE提升了3.4%~23.7%. DRLA为自适应流媒体系统提供理论与实践依据,优化视频分发,提升了用户视频观看体验.

参考文献

[1] 曹燕, 董一鸿, 邬少清, 陈华辉, 钱江波, 潘善亮. 动态网络表示学习研究进展[J]. 电子学报, 2020, 48(10): 2047-

2059.
CAO Yan, DONG Yi-hong, WU Shao-qing, CHEN Hua-hui, QIAN Jiang-bo, PAN Shan-liang. Dynamic network representation learning: A review[J]. Acta Electronica Sinica, 2020, 48(10): 2047-2059. (in Chinese)

[2] DOBRIAN F, SEKAR V, AWAN A, et al. Understanding the impact of video quality on user engagement[J]. ACM SIGCOMM, 2011, 41(4): 362-373.

[3] KRISHNAN S, SITARAMAN R. Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs[J]. IEEE/ACM Transactions on Networking, 2013, 21(6): 2001-2014.

[4] STOCKHAMMER T. Dynamic adaptive streaming over HTTP standards and design principles[C]//ACM Conference on Multimedia Systems. Scottsdale Arizona, USA: ACM, 2011: 133-144.

[5] MAO H, NETRAVALI R, ALIZADEH M. Neural adaptive video streaming with pensieve[C]//ACM Special Interest Group on Data Communication. Los Angeles, USA: ACM, 2017: 197-210.

[6] SPITERI K, URGAONKAR R, SITARAMAN R K. BOLA: Near-optimal bitrate adaptation for online videos[J]. IEEE/ACM Transactions on Networking, 2020, 28(4): 1698-1711.

[7] SUN Y, YIN X, JIANG J, et al. CS2P: Improving video bitrate selection and adaptation with data-driven throughput prediction[C]//ACM SIGCOMM Conference. Florianopolis, Brazil: ACM, 2016: 272-285.

[8] 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, 2014, 22(1): 326-340.

[9] YIN X, JINDAL A, SEKAR V, et al. A control-theoretic approach for dynamic adaptive video streaming over HTTP [C]//ACM Conference on Special Interest Group on Data Communication. London, UK: ACM, 2015: 325-338.

[10] CLAEYS M, LATRÉ S, FAMAEY J, et al. Design and optimisation of a Q-learning-based HTTP adaptive streaming client[J]. Connection Science, 2014, 26(1): 25-43.

[11] MAO H, CHEN S, DIMMERY D, et al. Real-world video adaptation with reinforcement learning[C]//International Conference on Machine Learning. California, USA: ACM, 2019: 1-10.

[12] LETHAM B, BAKSHY E. Bayesian optimization for policy search via online-offline experimentation[J]. Journal of Machine Learning Research, 2019, 20(145): 1-30.

[13] MNIH V, BADIA A P, MIRZA M, et al. Asynchronous methods for deep reinforcement learning[C]//International Conference on Machine Learning. New York: ACM, 2016: 1928-1937.

[14] LEKHARU A, MOULII K Y, SUR A, et al. Deep learning based prediction model for adaptive video streaming [C]//International Conference on COMmunication Systems & NETworkS. Bengaluru, India: IEEE, 2020: 152-159.

[15] GADALETA M, CHIARIOTTI F, ROSSI M, et al. D-DASH: A deep Q-learning framework for DASH video streaming[J]. IEEE Transactions on Cognitive Communications and Networking, 2017, 3(4): 703-718.

[16] HUO L, WANG Z, XU M, et al. A meta-learning framework for learning multi-user preferences in QoE optimization of DASH[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2019, 30(9): 3210-3225.

[17] SUTTON R S, BARTO A G. Reinforcement Learning: An Introduction[M]. 2th ed. London: MIT Press, 2018.

[18] MAO H, VENKATAKRISHNAN S B, SCHWARZKOPF M, et al. Variance reduction for reinforcement learning in input-driven environments[C]//International Conference on Learning Representations. New Orleans, USA: ACM, 2019: 1-20.

[19] HAMMERSLEY J. Monte Carlo Methods[M]. 4th ed. USA: Halsted Press, 2013.

[20] 郭宪,方勇纯. 深入浅出强化学习: 原理入门[M]. 北京: 电子工业出版社, 2018.

[21] ENGSTROM L, ILYAS A, SANTURKAR S, et al. Implementation matters in deep RL: A case study on ppo and trpo[C]//International Conference on Learning Representations. New Orleans: ACM, 2019: 1-14.

[22] SCHULMAN J, LEVINE S, ABBEEL P, et al. Trust region policy optimization[C]//International conference on machine learning. Lille, France: ACM, 2015: 1889-1897.

[23] BAUER S, CLARK D, LEHR W. Gigabit broadband measurement workshop report[J]. ACM SIGCOMM Computer Communication Review, 2020, 50(1): 60-65.

[24] RIISER H, VIGMOSTAD P, GRIWODZ C, et al. Commute path bandwidth traces from 3G networks: analysis and applications[C]//ACM Multimedia Systems Conference. Boston Massachusetts: ACM, 2013: 114-118.

[25] RACA D, QUINLAN J J, ZAHRAN A H, et al. Beyond throughput: A 4G LTE dataset with channel and context metrics[C]//Proceedings of the 9th ACM Multimedia Sys-

tems Conference. New York: ACM, 2018: 460-465.
[26] HUANG T, ZHOU C, YAO X, et al. Quality-aware neural adaptive video streaming with lifelong imitation learning[J]. IEEE Journal on Selected Areas in Communications, 2020, 38(10): 2324-2342.
[27] SPITERI K, SITARAMAN R, SPARACIO D. From theory to practice: Improving bitrate adaptation in the DASH reference player[J]. ACM Transactions on Multimedia Computing, Communications, and Applications, 2020, 15 (2): 1-29.

作者简介

易 令 男,1994年出生于贵州省铜仁市,硕士. 主要研究方向为自适应流媒体,深度强化学习.
E-mail: 2821210016@qq.com

李泽平(通讯作者) 男,1964年出生于贵州省贵阳市,博士,毕业于电子科技大学. 现为贵州大学计算机科学与技术学院教授. 主要研究方向为网络分发,视频流.
E-mail: zpli1@gzu.edu.cn