Deep Reinforcement Learning with Importance Sampling for QoE enhancement in Edge-Driven Video Delivery Services

Mandan Naresh†, Vikramjeet Das†, Manik Gupta and Paresh Saxena[1*]

Dept. of Computer Science and Information Systems, BITS Pilani, Hyderabad, India.

*Corresponding author(s). E-mail(s): psaxena@hyderabad.bits-pilani.ac.in;
Contributing authors: p20180420@hyderabad.bits-pilani.ac.in;
f20180280@hyderabad.bits-pilani.ac.in; manik.gupta@hyderabad.bits-pilani.ac.in;
†These authors contributed equally to this work.

Abstract

Adaptive bitrate (ABR) algorithms are used to adapt the video bitrate based on the network conditions to improve the overall video quality of experience (QoE). Further, with the rise of multi-access edge computing (MEC), a higher QoE can be guaranteed for video services by performing computations over the edge servers rather than the cloud servers. Recently, reinforcement learning (RL) and asynchronous advantage actor-critic (A3C) methods have been used to improve adaptive bit rate algorithms and they have been shown to enhance the overall QoE as compared to fixed-rule ABR algorithms. However, a common issue in the A3C methods is the lag between behavior policy and target policy. As a result, the behavior and the target policies are no longer synchronized with one another which results in suboptimal updates. In this work, we present the deep reinforcement learning with an importance sampling based approach focused on edge-driven video delivery services to achieve an overall better user experience. We refer to our proposed approach as ALISA: Actor-Learner Architecture with Importance Sampling for efficient learning in ABR algorithms. ALISA incorporates importance sampling weights to give higher weightage to relevant experience to address the lag issues incurred in the existing A3C methods. We present the design and implementation of ALISA, and compare its performance to state-of-the-art video rate adaptation algorithms including vanilla A3C and other fixed-rule schedulers. Our results show that ALISA provides up to 25%-48% higher average QoE than vanilla A3C, whereas the gains are even higher when compared to fixed-rule schedulers.

Keywords: Deep Reinforcement Learning, Edge Computing, Quality of Experience (QoE), Adaptive Bit Rates (ABR), and Actor-critic methods.

1 Introduction

The global edge computing market is projected to grow to a multi-billion-dollar scale at a CAGR of 38.4%, reaching $61.14 billion by 2028 [25]. This growth is largely expected to be driven by the deep integration of artificial intelligence (AI) systems on edge devices, helping them to make important decisions in a split second. AI has the potential to improve a multitude of mobile and edge services like video streaming, online gaming, voice over IP, smart home applications, remote health monitoring, etc. In particular, video streaming is projected to be a major contributor to global Internet traffic, with each user streaming 35 GB

of video streaming data per month on an average, contributing to about 77% of entire mobile data traffic [8]. Hence, it is of prime importance to improve the quality of experience (QoE) of video streaming for users.

Dynamic Adaptive Streaming over HTTP (DASH) [28] has become a prominent standard of streaming video content over the best effort Internet. In general, ABR algorithms have been widely explored for improving QoE in DASH-based video streaming [4]. ABR algorithms dynamically change the video bitrate based on the underlying network conditions such as buffer occupancy and observed throughput to provide a higher QoE for the users. These algorithms, however, follow a fixed set of rules to make decisions and are often optimized for specific scenarios. This makes such methods difficult to generalize to a wide variety of network conditions prevalent in today's ever evolving networks.

With the rise of multi-access edge computing (MEC) that offers low latency, high bandwidth and context awareness by performing the computations on network edge devices rather than a centralized cloud [10], there have been few recent works in the direction of DASH-based edge-driven video services [3, 11, 16, 34]. In [11], a real-time method for QoE estimation is proposed for edge-driven mobile video applications. Such estimation provides the service provider to optimize the video quality and bit-rate for assuring better QoE for users in dynamic environments. In [16], authors proposed a novel caching method to store the highest video bit rate video at the edge server instead of storing video with different bit-rates. The encoding offloading is transferred to edge-server from the cloud server and the proposed methods reduces the video load time and increases the cache hit-rate as compared to traditional store and forward caching mechanism. Additionally, the ML-driven content fetching for DASH is also proposed in [3] to improve the cache-hit ratio and to reduce the backhaul link utilization. Recently, an edge-based adaptive bit rate (ABR) algorithm is presented in [34] where edge server makes decision on video chunk quality and adapt rate based on the client's network conditions.

Reinforcement learning (RL) [31] is an area of machine learning concerned with how agents ought to take actions in an environment to maximize some notion of cumulative reward. Several recent works have explored the integration of such RL methods into video streaming and obtained encouraging results [19, 27] wherein, the goal is to achieve a high QoE. RL techniques with asynchronous advantage actor-critic (A3C) methods [21] have shown several advantages over fixed-rule based ABR algorithms. Several researchers [19, 27] have used a vanilla A3C method to generate adaptive bit rates for improving overall QoE. The A3C [22] agent consists of multiple actors and a central learner with a critic. Each actor generates experience based on its own behavior policy independently and in parallel with other actors. The individual experiences are then sent to the central learner which updates the target policy (policy that the A3C agent aims to learn) according to the generated experience. However, in many cases, A3C agents require a large amount of data to learn a suitable policy. Increasing the number of actors is a common solution to process the large amount of data while maintaining a low compute time. However, in such cases, each actor's behavior policy starts lagging behind the central learner's target policy [9]. As a result, the behavior and the target policies are no longer synchronized which results in suboptimal updates. To alleviate the problem, the use of importance sampling weights is suggested in [9] where weights are assigned to give more importance to relevant experience, and less importance to experience that is less probable [9]. In this paper, we propose deep reinforcement learning with important sampling for QoE enhancement in edge-driven video delivery services. Our solution is refereed to as ALISA: Actor-Learner architecture with Importance Sampling for enhancing QoE in ABR algorithms. The proposed method is capable of generating adaptive bit rates by using RL based actor-learner architecture without considering any pre-programmed model and assumptions of the underlying systems. The novel contribution of the current work is to integrate deep reinforcement learning with importance sampling to train, learn and generate adaptive bit rates at the edge server. The research contributions of this work are as follows:

- First, we propose the integration of actor-critic methods with importance sampling

weights [9] to generate ABR for edge-driven video delivery services. By assigning importance sampling weights and consequently, assigning more importance to relevant experience, our solution learns faster and provides a better overall QoE than state-of-the-art ABR algorithms.

- Second, we present the performance of our proposed approach using different data sets including traces from FCC [1], Norway [26], OBOE [2] and live video streaming [32]. We present a comprehensive study using three different variants of QoE metrics formulated as rewards for utilizing deep reinforcement learning. Finally, we also present the comparison over different network characteristics considering both lossless and lossy scenarios.

- Third, we present the comparison of our proposed approach with several state-of-the-art ABR algorithms. This includes a comparison with basic implementation of A3C, vanilla A3C [19] and the comparison with other non-RL ABR algorithms including RB [30], BOLA [29], RobustMPC [33], etc. Our results show that ALISA provides up to 25%-48% higher average QoE than vanilla A3C, whereas the gains are even higher when compared to the fixed-rule schedulers.

The remainder of the paper is organized as follows. Section 2 presents the related work HTTP-based video services and ABR algorithms. Section 3 presents the relevant background on reinforcement learning and actor-critic methods. Further, Section 4 presents the problem statement, the integration of importance sampling weights, proposed algorithm and system design. We present the experimental setup and results in Section 5 and Section 6, respectively. Finally, we conclude our work in Section 7.

2 Related Work

In this paper, we focus on the HTTP-based video delivery services with ABR generation at the edge server as shown in Figure 1. Such systems are primarily implemented using DASH [28]. We focus on edge-driven ABR approach, for example in [34], where ABR decisions are performed by edge

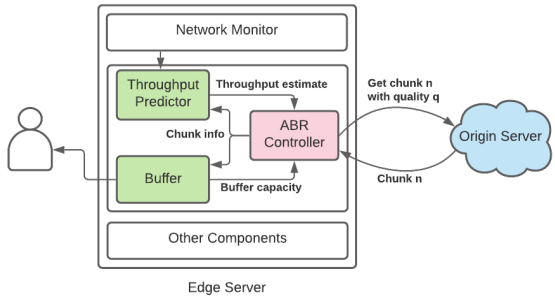

Fig. 1 HTTP based edge-drive video delivery services.

server instead of clients hence the edge computing power and storage capacity can be efficiently utilized. In such systems, the videos are stored in discrete chunks on the server. After connecting to the server, the edge server requests each chunk with a specific video bit rate from the server based on the client's requests. The edge-server uses an ABR algorithm to select the video bit rate based on the network conditions and sends it to the server. Several ABR algorithms have been proposed [13, 24, 29, 33] to generate adaptive bit rates for video distribution over wireless networks. The algorithms can be primarily classified into being rate-based or buffer-based.

Rate-based algorithms [30] predict the bitrate of the future chunk as the maximum supported bitrate based on available network bandwidth and based on past chunk history. Many of the rate-based algorithms, however, suffer from bias in the system [18], leading to overestimation of the available bitrate. In contrast, buffer-based algorithms [13, 29] make predictions based on the client's buffer occupancy.

However, since most of the proposed strategies operate based on pre-defined rules, they suffer from several drawbacks. First, these algorithms are susceptible to sudden changes in network conditions which can lead to inaccurate predictions. Second, several methods can be used to achieve a higher QoE, however, there is a tradeoff between such methods. For example, selecting the highest supported bitrate for every chunk can lead to loss of smoothness due to fluctuations in video resolution. Finally, the bitrate selection for a present chunk can often affect the bitrate selection for the future chunks. For example, downloading chunks

using the highest possible bitrate can results in a lower bit rate and quality for the future chunks to prevent rebuffering.

Recently, there has been focus on the development of a class of ABR algorithms that rely on application of reinforcement learning. There have been several attempts to use Q-Learning for this task [6, 7]. In these works, tabular Q-Learning is applied, which makes it infeasible to expand it to larger state spaces. Some works [6] also assume the Markovian property to hold, however, this is not always a good assumption to make because the predicted bitrate need not only depend on the last seen chunk, instead, it can be affected by multiple historical chunks. To solve the issue of the Q-Learning task becoming intractable with increasing feature space, actor-critic methods for ABR generation have been explored in [14, 19, 27]. In these papers, the A3C agent is used to generate ABRs, and achieve better QoE than most of the other fixed-rule based ABR algorithms. However, there are some issues with A3C [5, 9, 12, 17, 20] where one of them is the lagging behind of an actor's behavior policy as compared to the central learner's target policy. This impacts the performance of A3C agent in the existing RL-based video delivery systems, resulting in lower sample-efficiency and learning of a suboptimal policy. In this paper, we propose and investigate the use of assigning importance sampling weights [9] to the experiences based on their relevance and overcome a significant drawback with existing implementations of A3C agents with HTTP based video delivery systems.

3 Background

In this section, we present a brief overview of reinforcement learning and actor-critic methods.

3.1 Reinforcement Learning

A reinforcement learning solution aims to learn a mapping from the state space to the action space by repeated interaction between the RL agent and the environment. The RL problem is modeled as a Markov decision process with agents states and actions. Let us consider a discrete system where at each time step $t \in \{0, 1, 2, ...\}$, the RL agent observes its state s_t, takes an action a_t, moves to state s_{t+1} and receives a reward r_{t+1}. Further, for a sequence of states and actions, the discounted cumulative reward is defined as $R(\tau) = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$ where τ is the sequence of states and actions, i.e. $\{(s_t, a_t), (s_{t+1}, a_{t+1}), ...\}$ and $\gamma \leq 1$ is a discount factor. The agent selects action based on a policy,

$$\pi : \pi_\theta(s_t, a_t) \rightarrow [0, 1] \tag{1}$$

where $\pi_\theta(s_t, a_t)$ is the probability that action a_t is taken in state s_t and θ are the policy parameters upon which the actions are based. Following the policy π, the value function $V(s)$ for a state s is defined as

$$V(s) = \mathbb{E}_\pi[R(\tau)|S_t = s] \tag{2}$$

The goal of an RL agent is to find the optimal policy π^* that maximizes the overall discounted reward. The optimal policy is given by,

$$\pi^*(s_t) = \underset{a}{\text{argmax}}[R(s_t, a_t) + \gamma V(s_{t+1})] \tag{3}$$

3.2 Actor-Critic Methods

As an improvement to the value-based methods, actor-critic methods have been proposed [22], which makes an update using gradient ascent at every step without having to wait till the end of an episode. The policy update with respect to its parameters θ is defined in terms of the gradient operator ∇ as follows,

$$\Delta\theta = \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) \hat{q}_w(s_t, a_t) \tag{4}$$

where α is the actor learning rate, $\hat{q}_w(s_t, a_t)$ is the critic function that indicates how good an action a_t is in state s_t. The parameters w of the critic function are updated as follows,

$$\Delta w = \xi(R(s_t, a_t) + \gamma \hat{q}_w(s_{t+1}, a_{t+1}) - \hat{q}_w(s_t, a_t)) \nabla_w \hat{q}_w(s_t, a_t) \tag{5}$$

where ξ is the critic learning rate.

However, a policy network trained in this manner may have high variance, which can cause instability during training. To mitigate this issue, the advantage actor-critic (A2C) [22] framework

introduces the advantage function to determine the advantage of the action taken in state s_t as compared to the average value of actions in s_t. The advantage function is defined as the temporal difference (TD) error:

$$A(s_t, a_t) = r_t + \gamma V(s_{t+1}) - V(s_t) \tag{6}$$

The final gradient-based update for the actor is as follows,

$$\Delta\theta = \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) A(s_t, a_t) + \beta \nabla_\theta H(\pi_\theta(.|s_t)) \tag{7}$$

where $H(\pi_\theta(.|s_t))$ is the entropy factor which promotes random actions and β is the regularization term. The entropy term is defined as

$$H(\pi_\theta(.|s_t)) = -\sum_a \pi_\theta(a|s_t) \log(\pi_\theta(a|s_t)) \tag{8}$$

The value of β is initially set to a high value to promote exploration early on, and it is reduced as training progresses. To enhance training speed, Asynchronous Advantage Actor-Critic (A3C) framework [22] is proposed to simulate multiple actors in parallel and asynchronously. These actors synchronize their parameters with the central learner at regular intervals.

4 Problem Statement and Proposed Solution

In this section, we present the problem formulation and the proposed solution including the system design details for deep RL based edge-driven video delivery system.

4.1 The Issue

Reinforcement learning agents often require a large amount of experience to model the environment effectively and accurately. Increasing the number of actors during training is a common technique to achieve this goal. However, there is an inherent issue with this method. As shown in Figure 2, the central learner first synchronizes its weights with all the actors (Step 1) and then actors provide their experience to the central learner (Step 2). However, there are situations when updates may lag from some of the actors. For example, as shown in Figure 2, the central learner updates its target policy even before its receives the experience from the actor on the right side (Step 3). Therefore, the behavior policy corresponding to this experience lags behind the target policy, and the experience may not be as relevant to update the current target policy. This issue is only aggravated by the presence of more actors, and learning shifts off-policy as actors generate experience with an older version of the behavior policy. The lack of synchronization between the behavior and target policies results in suboptimal updates. Ultimately, this leads to learning an overall suboptimal policy. We aim to develop methods that counteract the lagging of the behaviour policy behind the target policy during training, which in turn will help us achieve better performance on unseen test data.

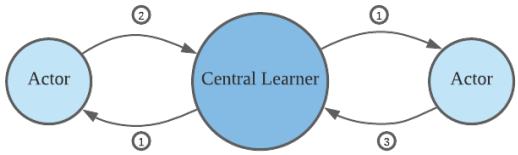

Fig. 2 Illustration of A3C lagging issue. The figure shows three steps. Step (1): Each actor synchronizes its weights with the central learner; Step (2): One of the actors provides experience to the central learner, which updates the weights of the target policy; Step (3): The other actor provides experience to the central learner. The behavior policy for the experience is not synchronized with the latest version of the target policy (updated in step 2), hence the experience is based on an older policy.

4.2 Integration of Importance Sampling and ALISA Algorithm for Policy Update

Importance sampling is a commonly used method to address the issue of a data distribution mismatch. It enables us to estimate the expected value of a function $f(x)$, where x follows a probability density function q on the domain $\mathcal{D}$, using values sampled from a different distribution r on the same domain $\mathcal{D}$ as,

$$\mathbb{E}(f(X)) = \mathbb{E}_r\left(\frac{f(X)q(X)}{r(X)}\right) \quad (9)$$

Importance sampling transforms the data drawn from the distribution r such that it appears to be sampled from the distribution q. This effectively addresses the distribution mismatch issue, which in our case occurs due to the lagging of the behavior policy behind the target policy.

To overcome the distribution mismatch between the target policy π and the behavior policy μ, we employ importance sampling. Correlating to the notation in Equation (9), we have $q \equiv \pi$ and $r \equiv \mu$. Similar to the authors of [9], we use the n-step V-trace target to correct for the off-policy shift. The n-step V-trace target now serves as an estimate of the value function V for the target policy π using an older version of the behavior policy μ. The n-step V-trace target is defined as,

$$v_j \doteq V(s_j) + \sum_{t=j}^{j+n-1} \gamma^{t-j} \left(\prod_{i=j}^{t-1} c_i\right) \delta_t V \quad (10)$$

where,

- $\delta_t V \doteq \rho_t(r_t + \gamma V(s_{t+1}) - V(s_t))$ is the temporal difference.
- $\rho_t \doteq min\left(\overline{\rho}, \frac{\pi(a_t|s_t)}{\mu(a_t|s_t)}\right)$ and $c_i \doteq min\left(\overline{c}, \frac{\pi(a_i|s_i)}{\mu(a_i|s_i)}\right)$ are the importance sampling weights. The importance sampling weights ρ_t and c_i are used to give importance to experience which is more relevant to the target policy than the behavior policy. Here, π denotes the target policy and μ denotes the behavior policy.
- $\overline{\rho}$ and $\overline{c}$ are lower threshold values for their corresponding importance sampling weights, which we set to 1 throughout our work.
- $\overline{\rho_t}$ denotes how much more probable the action a_t taken in state x_t is according to the target policy compared to the behavior policy.
- $\prod_{i=j}^{t-1} c_i$ denotes how much more probable the predicted path from state s_j to s_{t-1} is according to the target policy compared to the behavior policy.

Subsequently, the V-trace targets are used in place of V for gradient computation. The n-step V-trace target can also be defined recursively as

$$v_j = V(s_j) + \delta_j V + \gamma c_j(v_{j+1} - V(s_{j+1})) \quad (11)$$

which we use during implementation throughout our work. As a result of these calculations, actions which are more likely to be taken according to the target policy contribute more to the V-trace target. Hence the importance sampling weights help the reinforcement learning model to focus on the experience which is more relevant and leads to better parameter updates and assign less importance to suboptimal experience. Using the above definition of the V-trace target, Algorithm 1 outlines ALISA's policy update algorithm with the importance sampling weights where n equals to the length of the episode. The V-trace target calculation with ALISA takes as input the information related to an episode consisting of the sequence of states (s), the sequence of action probabilities according to the behavior policy (a_b), the sequence of rewards (r) along with the metaparameters $\overline{\rho}$ and $\overline{c}$ and the actor and critic models. As a consequence of importance sampling, actions which are more likely to be taken by the current target policy contribute more to the gradients compared to actions which are likely to be taken by earlier lagging versions of the target policy, but not the current one. This algorithm guides the RL model to focus on experience which matters more, and assign less importance to other less relevant experience.

4.3 ALISA: System Design

ALISA uses the familiar DASH framework and deep reinforcement learning algorithms like A3C as key components of its design and improves on them to achieve better QoE for video streaming. Figure 3 shows the main components of our system. The user streams a video on their devices which connects to the edge server. The edge server contains several components. The main component is the ABR controller, which handles the process of choosing actions. It observes several state parameters such as the bandwidth, bitrate selection history and the buffer occupancy and decides the action to take i.e. the bitrate selection

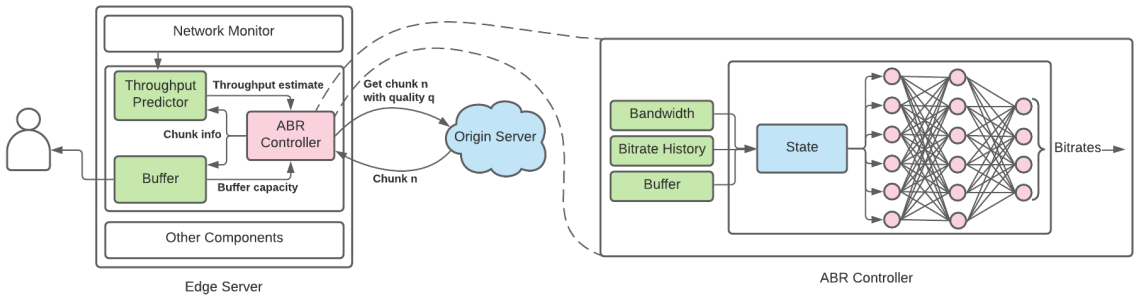

Fig. 3 ALISA: System Design

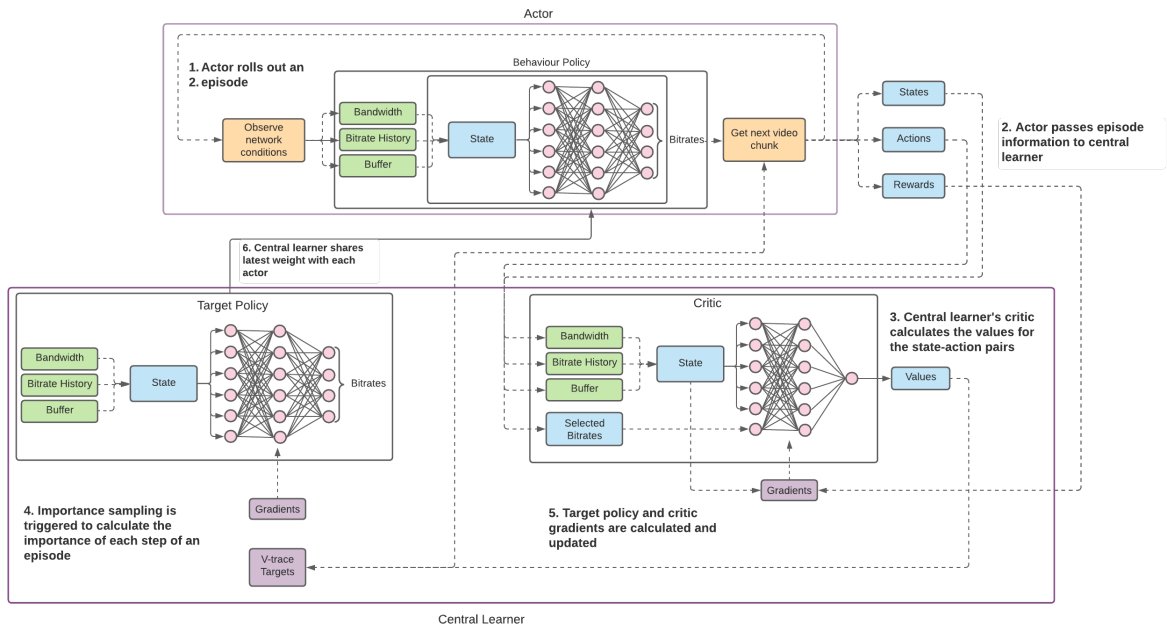

Fig. 4 Detailed flow design for training the RL based ABR controller

for the next chunk. At each step, it also observes some reward (QoE) as a result of its actions.

We now describe the training process used for the ABR controller. As shown in Figure 4 the training setup consists of several actors being coordinated by a single central learner. The actor contains a behavior policy as its parameters, while the central learner maintains the target policy and the critic parameters. All three of the behaviour policy, target policy and the critic function are modelled as a neural network. The training process can be considered as the repetition of the following steps until convergence:

1. First, an actor simulates an episode and generates a batch of experience consisting of the states, the corresponding actions taken by the actor and the rewards received as a result.

2. The experience is then passed back to the central learner.

Algorithm 1 Calculation of V-trace targets using ALISA for an episode

Input:
s: Sequence of states
$\boldsymbol{a_b}$: Sequence of action probabilities according to behavior policy
$\boldsymbol{r}$: Sequence of rewards
$\overline{\boldsymbol{\rho}}$: Lower threshold for ρ
$\overline{\boldsymbol{c}}$: Lower threshold for c
actor: target policy from central learner
critic: critic model
Result:
v: V-trace targets
$V \leftarrow$ critic values for s
$p_b \leftarrow$ behavior policy probabilities for optimal action
$a_t \leftarrow$ target policy probabilities for s
$p_t \leftarrow$ target policy probabilities corresponding to optimal actions of behavior policy, computed using a_t and p_b
$\rho \leftarrow min\left(\overline{\rho}, \frac{p_t}{p_b}\right)$
$\delta_t V \leftarrow \rho\left(\mathrm{r} + \gamma\ V_{+1} - V\right)$
$c \leftarrow min\left(\overline{c}, \rho\right)$
$v \leftarrow V + \delta_t V$
foreach *position* $\boldsymbol{p}$ *from rear of v_trace* **do**
  $v_p \mathrel{+}= \gamma c_p * (v_{p+1} - V_{p+1})$
end

3. The central learner calculates the values for each step of the experience using the critic parameters.

4. The central learner calculates the $V-$trace targets after incorporating the importance sampling strategy discussed in Algorithm 4.2.

5. The critic gradients are computed using the observed states and their corresponding rewards, while the target policy gradients are computed using the observed states, the corresponding actions, the obtained rewards and the $V-$trace targets. The target policy and the critic network are now updated using back propagation.

6. Finally, the central learner shares the latest version of the target policy with each actor, which set their behaviour policy to the newest target policy to generate the next batch of experience.

ALISA effectively decouples the processes of acting and learning, while also correcting for the off-policy shift that can occur as a result. This has significant implications for learning ABR algorithms. Since the large amount of video is streamed for the users across the world and with the upcoming advancements in edge computing, the ALISA's architecture allows to use the massive amount of data to continuously fine-tune the ABR algorithm and adapt to the ever changing network conditions, all without putting the privacy of the users at risk. While the edge devices continuously make decisions to select the bitrate, the decision choices can be passed on to the central learner on a remote cloud server, where learning can take place in a federated [15] manner. The latest policy can be synchronized between the edge devices and remote server at regular intervals. In this setting, the policy of the central learner may lag behind the policy using which the edge devices select bitrates. This distribution mismatch is effectively handled by importance sampling integrated into ALISA.

5 Experimental Details and Training Methodology

In this section, we describe the experimental setup together with the performance metrics used to evaluate ALISA's performance.

5.1 Experimental Setup

We use the Python based framework proposed in [19], to generate and test our ABR algorithms. The client requests chunks of data from the edge server and provides it with parameters pertaining to the observed network conditions like bandwidth, buffer occupancy and bitrate history. We have integrated importance sampling as described in Section 4 and assigned weights to the target and the behavior policies. To emulate network conditions for effectively testing our trained RL model, the MahiMahi [23] framework has been used, which is a record-and-replay HTTP framework for this task.

We use four datasets as part of our training set: the broadband dataset provided by the FCC [1], the mobile dataset collected in Norway [26],

the OBOE traces [2] and the live video streaming dataset [32], which have been pre-processed according to the MahiMahi format.

Subsequently, we compare ALISA with the following state-of-the-art ABR algorithms:

- Vanilla A3C [19]: uses vanilla A3C without any additional techniques to train the agent for delivering adaptive bit rates.
- Rate-Based (RB) [30]: RB predicts the maximum supported bitrate based on the harmonic mean of past observed throughput.
- Buffer-based (BB) [13]: BB selects bitrate based on client's buffer occupancy.
- BOLA [29]: Bitrate selection is done exclusively based on buffer occupancy, using Lyapunov optimization.
- RobustMPC[33]: MPC uses buffer occupancy observations and throuhgput predictions similar to RB. Additionally, RobustMPC accounts for error between predicted and observed throuhgputs by normalizing throughput estimates by the maximum error seen in the past 5 chunks.

To quantify the performance of the ABR algorithms, we use the formulation of QoE:

$$QoE = \sum_{n=1}^{N} q(b_n) - \mu \sum_{n=1}^{N} T_n - \sum_{n=1}^{N-1} |q(b_{n+1}) - q(b_n)| \tag{12}$$

where b_i and $q(b_i)$ represent the bit-rate and quality, respectively, for chunk i. A higher bit rate means a higher quality and a higher QoE. However, there are also penalties due to rebuffering time T_i (represented by second term) and fluctuations in video quality (represented by the final term) that hinders the overall smoothness. In this paper, we evaluate the performance of the proposed approaches with three QoE variants [19] that depend on the above general QoE metric:

- QoE_{lin}: $q(b_n) = b_n$ where value of rebuffer penalty is $\mu = 4.3$,
- QoE_{log}: $q(b_n) = log(b/b_{min})$ that considers the fact that the marginal improvement in quality decreases at higher bitrates with $\mu = 2.66$ and
- QoE_{HD}: assigns a higher value to higher quality bitrates and lower values to lower quality bitrates with $\mu = 8$.

5.2 Dataset details

We use the following data sets for training, validation and testing. Our selection of data sets for training, validation, and testing is in line with the previous experimental setups in [19], [2], [32]. For training, we have used three different data sets. The first data set consists of 127 traces, out of which 59 belongs to the FCC [1] dataset while the remaining 68 belongs to the Norway HSDPA [26] dataset. The second data set consists of 428 OBOE traces [2] and the third data set consists of 100 live video streaming traces [32]. To demonstrate the benefits of ALISA over different trained models, we have generated three different trained model corresponding to three difference data sets described above. For all the three trained model, we have used the same validation data set, i.e., 142 Norway traces. Finally, for all the three trained models, after validation, the testing is performed using 205 traces from the FCC dataset and 250 traces from the Norway HSDPA dataset.

5.3 Training Methodology

We train the three models for each configuration, one each for QoE_{lin}, QoE_{log} and QoE_{HD} as reward metrics. For each model, we use a consistent set of hyperparameters throughout. The discount factor γ is set to 0.99. The learning rates are set to 0.0001 and 0.001 for the actor and critic respectively. Additionally, we also set both importance sampling thresholds $\bar{\rho}$ and $\bar{c}$ to 1. We train multiple models for different configurations of entropy weights. First, we train several models with a constant entropy weight for 100,000 epochs. Next, we use a decaying entropy weight where the entropy is gradually decreased over 100,000 epochs.

5.4 Testing Methodology

We select the model with the highest validation QoE for testing. We perform testing under both lossless and lossy conditions simulated using the MahiMahi [23] framework. We perform tests under packet loss percentages of 0%, 0.1%, 0.5%, 1% and 2%, where random packets are dropped

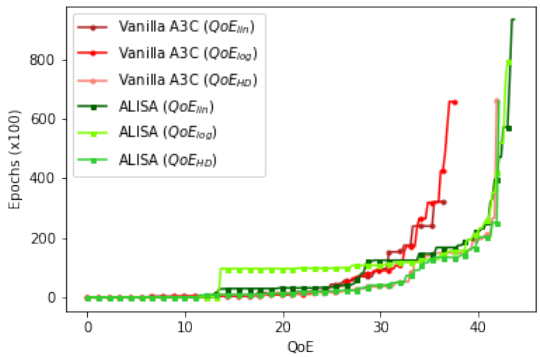

Fig. 5 Maximum training QoE obtained versus epochs elapsed. ALISA is able to obtain a higher QoE faster for all the three variants of QoE metric.

from the video stream. We evaluate all models on the three different QoE metrics discussed in Section 5.1.

6 Results

In this section, we present the results and comparison of ALISA with other state-of-the-art ABR algorithms.

6.1 Convergence Speed

ALISA takes advantage of the importance sampling strategy during training. As a result, it is often able to achieve a higher QoE compared to Vanilla A3C in a shorter time. Figure 5 presents the plots of the epochs elapsed versus the maximum QoE achieved till then. These plots are generated during the training using the first data set, i.e., 127 traces from FCC and Norway data sets. Our results show that by the time ALISA obtains a fairly high QoE of over 40, Vanilla A3C is only able to obtain a highest QoE of approximately 35. This demonstrates ALISA's advantage in learning and adapting to newer conditions faster, resulting in shorter training times. Similar results are observed during the training using OBOE and live video streaming traces.

6.2 Comparison with state-of-the-art ABR algorithms

6.2.1 Results with training and validation data sets

We perform a comprehensive set of training on the three data sets and report our results on QoE_{lin}, QoE_{log} and QoE_{HD} metrics. Table 1 presents the rewards obtained after training and validation on the FCC and Norway traces, OBOE traces and the live video streaming traces. We also investigate the use of different values for entropy regularization β. Specifically, we consider the following values: 0, 0.001, 0.01, 0.1, 0.25, 0.5, 0.75, and 1. We have also explored the use of variations in β during the training. For example, in Table 1, $\{2, 1.5, 1, 0.5, 0.1\}$ refers to the scenario where $\beta = 2$ for the first 20000 epochs, $\beta = 1.5$ for the next 20000 epochs, and so on, till $\beta = 0.1$ for the last 20000 epochs. Similarly, we have also used constant entropy regularization, where 0.1×5, in Table 1, refers to scenario where $\beta = 0.1$ for all 100000 iterations.

We note that the models do not converge well on training with a very low or very high entropy. We found a constant entropy of 0.1 to work well for QoE_{lin} and QoE_{log} metrics, while 0.75 worked well for QoE_{HD}. We have also trained multiple times using a decaying entropy regularization. We start from a high value, and gradually decrease our entropy weight every 20,000 epochs, gradually going down to 0.1. We find that decaying entropy regularization is more effective in almost all the cases as seen from Table 1 since after a few epochs, a high exploration is not required to achieve an optimal policy.

6.2.2 Results with Test Data Sets

We also compare ALISA to several other state-of-the-art ABR algorithms such as RB, BB, BOLA and RobustMPC described in the previous section. We have also compared ALISA with Vanilla A3C, an RL based basic A3C approach that does not utilize the importance sampling weights. Table 2 presents the comparison when there are no losses in the network. Our results show that ALISA achieves a higher QoE on all metrics over all different configurations. ALISA obtains upto 25% higher QoE than RB, 230%

Entropy values	FCC and Norway Traces			OBOE Traces			Live Video Streaming Traces		
	QoE_{lin}	QoE_{log}	QoE_{HD}	QoE_{lin}	QoE_{log}	QoE_{HD}	QoE_{lin}	QoE_{log}	QoE_{HD}
0 (×5)	13.61	13.62	13.57	30.76	30.8	39.51	13.61	30.75	29.97
0.001 (×5)	13.61	13.62	13.61	42.47	30.78	39.52	13.61	30.87	13.57
0.01 (×5)	13.61	13.61	13.61	38.79	39.25	39.44	30.53	30.93	37.84
0.1, 2, 0.1, 2, 0.1	41.57	38.91	42.21	41.43	44.03	38.4	43.27	38.54	41.36
0.1 (×5)	43.88	43.29	37.88	42.89	38.37	38.55	44.1	30.76	27.96
0.25 (×5)	40.91	43.29	37.61	42.85	43.2	39.57	43.22	44.03	37.69
0.5 (×5)	38.11	37.14	**42.42**	36.79	37.99	39.17	37.86	38.35	41.71
0.75 (×5)	31.11	32.11	41.99	29.85	30.69	40.97	32.23	32.83	42.28
1, 0.75, 0.5, 0.25, 0.1	43.22	44.09	42.4	**44.34**	**44.94**	40.61	**44.76**	**45.96**	41.66
1 (×5)	27.46	25.8	40.98	25.07	25.15	41.11	26.41	26.07	41.17
2, 1.5, 1, 0.5, 0.1	43.86	44.19	41.27	43.16	43.73	**41.16**	44.45	45.54	41.76
3, 2, 1, 0.5, 0.1	**43.92**	44.31	42.11	43.95	43.36	40.56	44.25	44.54	42.32
4, 2, 1, 0.5, 0.1	43.15	43.4	41.8	42.99	44.93	39.33	43.92	45.93	**42.66**
5, 2, 1, 0.5, 0.1	41.94	**44.8**	42.06	43.23	42.33	40.99	43.52	45.12	41.64

Table 1 Average QoE after training ALISA with all the three datasets and all the three variants of QoE metrics.

higher QoE than BB, 30% higher QoE than BOLA, 25% higher QoE than RobustMPC and 20% higher QoE compared to Vanilla A3C when tested under lossless conditions. This performance translates to lossy conditions as well. We note that ALISA is able to obtain upto 25%, 28%, 48% and 48% higher QoE compared to vanilla A3C under losses of 0.1%, 0.5%, 1% and 2% respectively. We summarize the remained of our testing QoE metrics for a random packet loss percentage of 0.1%, 0.5%, 1% and 2% in Table 3, Table 4, Table 5 and Table 6, respectively. These results indicate that ALISA achieves a significantly better performance than many other fixed-rule based ABR algorithms and also vanilla A3C. Further, we also visualize the different components of the QoE metric from equation (12) to understand how ALISA performs better than other ABR algorithms. Figure 7 shows how ALISA can consistently achieve higher bitrates than other methods. This increases the first component of QoE. From Figure 8, we note that ALISA maintains a lower buffer size than vanilla A3C, leading to an decrease in the second component in equation (12) hence ALISA provides a lower rebuffer penalty. Overall, this leads to a higher quality of experience for ALISA over other ABR algorithms.

7 Conclusion

We show how importance sampling and a structured entropy selection significantly improves the performance of vanilla A3C methods on the task of generating ABR algorithms for edge-driven video delivery services. By employing these methods as part of our proposed system, ALISA, we are able to consistently achieve an improvement in QoE of 25-48% and higher in certain cases. We also test our methods on a wider variety of conditions in terms of packet losses and find similar improvements. Finally, we also visualize and compare the bitrate selection and buffer size of ALISA with other ABR algorithms and find that ALISA performs better on both aspects, leading to an improved quality of experience. The future work includes the investigation of advanced hybrid cloud-edge architectures for the deployment of ALISA. Further, we also aim to investigate ALISA in a federated setup to utilize distributed training across multiple decentralized edge devices.

8 Acknowledgments

This work has been supported by TCS foundation, India under the TCS research scholar program, 2019-2023 and by DST-SERB Government of India Start-Up Research Grant (SRG/2019/002027).

9 Declarations

Conflict of interest: The authors declare that they have no conflict of interest.

Algorithm	FCC and Norway Traces			OBOE Traces			Live video streaming Traces		
	Linear	Log	HD	Linear	Log	HD	Linear	Log	HD
ALISA	**43.03**	**42.37**	**256.29**	**42.5**	**41.79**	**237.27**	**46.57**	**44.36**	228.63
Vanilla A3C	39.62	35.26	239.08	37.52	37.01	194.29	39.12	41.68	**234.72**
BB	12.02	12.78	84.24	14.08	20.00	80.36	13.81	20.26	63.08
RB	35.62	36.45	139.82	36.15	37.97	138.02	37.44	37.35	120.52
BOLA	34.25	35.3	141.04	35.04	37.09	139.1	35.82	36.05	121.02
RobustMPC	39.93	40.44	195.52	40.21	38.03	188.65	40.59	38.99	177.58

Table 2 Average QoE achieved on all the datasets with all the three variants of QoE metrics under emulation with no packet losses.

Algorithm	FCC and Norway Traces			OBOE Traces			Live video streaming Traces		
	Linear	Log	HD	Linear	Log	HD	Linear	Log	HD
ALISA	**44.57**	**43.75**	**244.38**	**44.46**	**42.07**	**237.91**	**45.76**	**41.89**	233.96
Vanilla A3C	41.99	35.06	235.12	39.34	36.69	194.37	36.62	39.08	**241.10**
BB	16.61	20.97	73.16	17.08	19.81	0.64	15.40	19.58	68.76
RB	38.40	38.73	132.98	39.05	37.64	35.58	37.15	37.65	130.14
BOLA	37.57	37.21	129.27	37.84	36.03	34.81	35.95	36.08	126.24
RobustMPC	41.89	37.30	189.53	42.80	37.97	187.43	40.97	37.56	182.75

Table 3 Average QoE achieved on all the datasets with all the three variants of QoE metrics under emulation of random packet drops with 0.1% probability.

Algorithm	FCC and Norway Traces			OBOE Traces			Live video streaming Traces		
	Linear	Log	HD	Linear	Log	HD	Linear	Log	HD
ALISA	37.34	**43.86**	**236.15**	39.2	**42.24**	**231.37**	**42.53**	**43.33**	227.66
Vanilla A3C	35.43	34.1	214,59	36.44	36.22	201.32	34.98	40.80	**231.19**
BB	10.59	18.74	66.13	13.97	18.79	69.6	12.12	18.41	64.51
RB	32.91	35.89	108.86	34.19	35.35	117.44	33.99	34.92	111.23
BOLA	32.39	34.59	108.98	34.12	33.97	112.67	33.78	33.77	110.80
RobustMPC	**37.99**	38.2	177.87	**39.48**	38.46	184.96	38.62	38.09	179.55

Table 4 Average QoE achieved on all the datasets with all the three variants of QoE metrics under emulation with 0.5% probability.

Algorithm	FCC and Norway Traces			OBOE Traces			Live video streaming Traces		
	Linear	Log	HD	Linear	Log	HD	Linear	Log	HD
ALISA	**34.87**	**38.18**	**198.34**	**35.35**	**44.86**	**181.71**	**36.97**	**38.2**	187.68
Vanilla A3C	29.68	29.22	174.53	29.04	30.18	138.71	29.50	29.22	**188.78**
BB	2.90	10.11	19.07	2.91	10.57	12.67	6.43	10.11	-15.75
RB	27.62	28.58	78.17	27.38	28.72	77.99	28.00	28.58	25.38
BOLA	26.93	27.74	77.30	27.59	27.95	77.12	26.89	27.74	25.00
RobustMPC	32.92	33.13	149.21	32.62	33.47	142.51	33.74	33.13	145.87

Table 5 Average QoE achieved on all the datasets with all the three variants of QoE metrics under emulation with 1% probability.

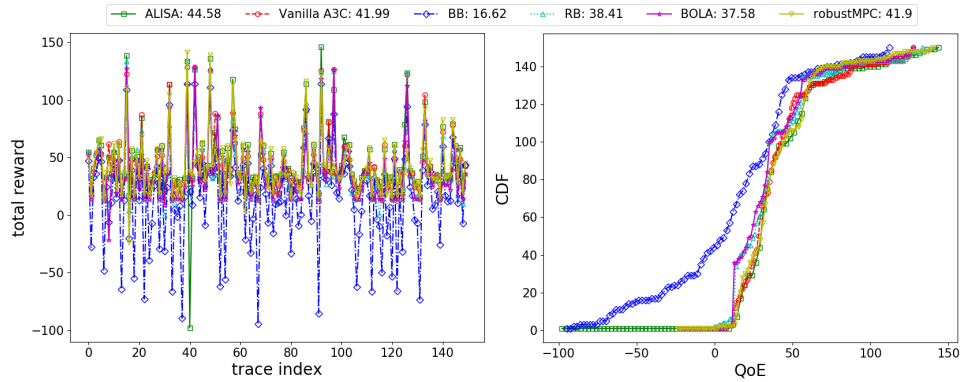

Fig. 6 Comparison of ALISA over other ABR algorithms with the QoE_{linear} metric: (left) average reward over all test traces; (right) CDF vs QoE plot

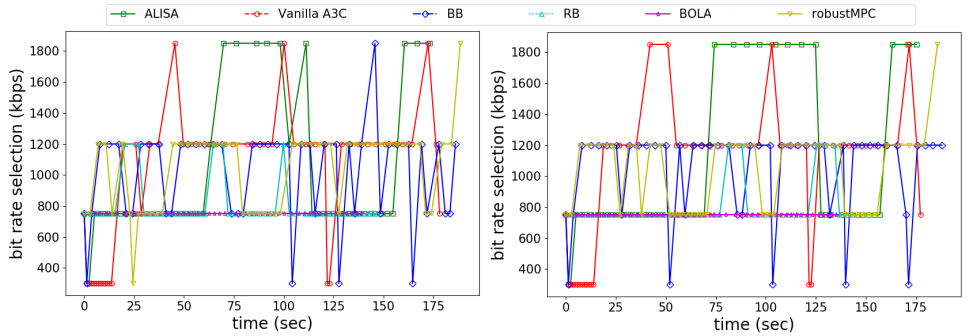

Fig. 7 Comparison of bit rate selection for ALISA over other ABR algorithms for two sample traces.

Ethical approval : This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent For this type of study formal consent is not required.

Funding details: This work has been supported by TCS foundation, India under the TCS research scholar program, 2019-2023 and by DST-SERB Government of India Start-Up Research Grant (SRG/2019/002027).

Authorship contributions: MN and VD contributed equally to this manuscript. MG and PS supervised the process. The manuscript was first written by MN and VD and then edited by MG and PS.

References

[1] 2016. Federal communications commission. 2016. raw data - measuring broadband america.

[2] Akhtar, Z. August 20–25, 2018, Budapest, Hungary. Oboe: Auto-tuning video abr algorithms to network conditions. *Oboe: Auto-tuning Video ABR Algorithms to Network Conditions* .

[3] Behravesh, R., D.F. Perez-Ramirez, A. Rao, D. Harutyunyan, R. Riggio, and R. Steinert 2020. Ml-driven dash content pre-fetching in mec-enabled mobile networks. In *2020 16th International Conference on Network and Service Management (CNSM)*, pp. 1–7. IEEE.

[4] Bentaleb, A., B. Taani, A.C. Begen, C. Timmerer, and R. Zimmermann. 2019. A survey on bitrate adaptation schemes for streaming

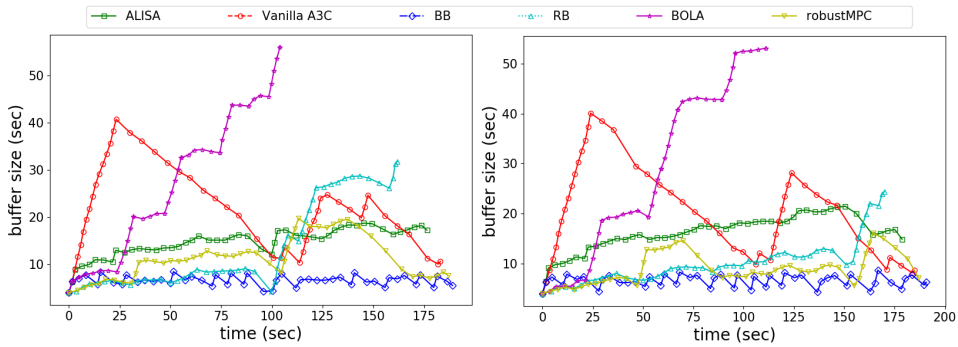

Fig. 8 Comparison of buffer size selection for ALISA over other ABR algorithms for two sample traces.

Algorithm	FCC and Norway Traces			OBOE Traces			Live video streaming Traces		
	Linear	Log	HD	Linear	Log	HD	Linear	Log	HD
ALISA	**27.60**	**28.98**	**128.53**	**25.62**	**26.67**	**122.05**	**28.73**	**30.83**	**122.47**
Vanilla A3C	24.28	21.48	111.57	22.35	22.98	82.05	22.45	28.56	121.41
BB	-13.44	-6.74	-41.18	-15.48	-6.28	-37.73	-14.49	-4.15	-38.48
RB	16.09	17.35	13.44	17.27	17.75	14.88	17.21	18.22	14.21
BOLA	17.89	18.59	16.05	18.08	18.01	16.08	17.48	18.76	16.49
RobustMPC	24.43	20.72	99.98	25.28	21.66	99.14	24.98	21.69	97.09

Table 6 Average QoE achieved on all the datasets with all the three variants of QoE metrics under emulation with 2% probability.

media over http. *IEEE Communications Surveys Tutorials 21*(1): 562–585. https://doi.org/10.1109/COMST.2018.2862938 .

[5] Chen, S., X.F. Zhang, J.J. Wu, and D. Liu 2018. Averaged-a3c for asynchronous deep reinforcement learning. In L. Cheng, A. C. S. Leung, and S. Ozawa (Eds.), *Neural Information Processing*, Cham, pp. 277–288. Springer International Publishing.

[6] Chiariotti, F., S. D'Aronco, L. Toni, and P. Frossard 2016. Online learning adaptation strategy for dash clients. In *Proceedings of the 7th International Conference on Multimedia Systems*, MMSys '16, New York, NY, USA. Association for Computing Machinery.

[7] Claeys, M., S. Latré, J. Famaey, T. Wu, W.V. Leekwijck, and F.D. Turck. 2014. Design and optimisation of a (fa)q-learning-based http adaptive streaming client. *Connection Science 26*(1): 25–43. https://doi.org/10.1080/09540091.2014.885273 .

[8] Ericsson. 2021, June. Mobile data traffic outlook. https://www.ericsson.com/en/mobility-report/dataforecasts/mobile-traffic-forecast.

[9] Espeholt, L., H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, et al. 2018. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *Proceedings of the International Conference on Machine Learning (ICML)*.

[10] Filali, A., A. Abouaomar, S. Cherkaoui, A. Kobbane, and M. Guizani. 2020. Multi-access edge computing: A survey. *IEEE Access* 8: 197017–197046. https://doi.org/10.1109/ACCESS.2020.3034136 .

[11] Ge, C. and N. Wang 2018. Real-time qoe estimation of dash-based mobile video applications through edge computing. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 766–771. IEEE.

[12] Holliday, J. and T. Le 2020, 07. Follow then forage exploration: Improving asynchronous advantage actor critic. pp. 107–118.

[13] Huang, T.Y., R. Johari, N. McKeown, M. Trunnell, and M. Watson 2014. A buffer-based approach to rate adaptation: Evidence from a large video streaming service. In *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, New York, NY, USA, pp. 187–198. Association for Computing Machinery.

[14] Jin, H., Q. Wang, S. Li, and J. Chen 2020. Joint qos control and bitrate selection for video streaming based on multi-agent reinforcement learning. In *2020 IEEE 16th International Conference on Control Automation (ICCA)*, pp. 1360–1365.

[15] Konečný, J., H.B. McMahan, F.X. Yu, P. Richtarik, A.T. Suresh, and D. Bacon 2016. Federated learning: Strategies for improving communication efficiency. In *NIPS Workshop on Private Multi-Party Machine Learning*.

[16] Kumar, S., D.S. Vineeth, et al. 2018. Edge assisted dash video caching mechanism for multi-access edge computing. In *2018 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, pp. 1–6. IEEE.

[17] Labao, A.B., M.A.M. Martija, and P.C. Naval. 2021. A3c-gs: Adaptive moment gradient sharing with locks for asynchronous actor–critic agents. *IEEE Transactions on Neural Networks and Learning Systems 32*(3): 1162–1176. https://doi.org/10.1109/TNNLS.2020.2980743 .

[18] Li, Z., X. Zhu, J. Gahm, R. Pan, H. Hu, A.C. Begen, and D. Oran. 2014. Probe and adapt: Rate adaptation for http video streaming at scale. *IEEE Journal on Selected Areas in Communications 32*(4): 719–733. https://doi.org/10.1109/JSAC.2014.140405 .

[19] Mao, H., R. Netravali, and M. Alizadeh 2017. Neural adaptive video streaming with pensieve. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, SIGCOMM '17, New York, NY, USA, pp. 197–210. Association for Computing Machinery.

[20] Medini, T., X. Luan, and A. Shrivastava 2018. A4c: Anticipatory asynchronous advantage actor-critic.

[21] Mnih, V. et al. 2016. 2016. asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*. 19281937. V. Mnih et al Asynchronous methods for deep reinforcement learning. In International Conference on Machine Learning. 19281937.

[22] Mnih, V., A.P. Badia, M. Mirza, A. Graves, T.P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. *CoRR* abs/1602.01783. https://arxiv.org/abs/1602.01783 .

[23] Netravali, R., A. Sivaraman, S. Das, A. Goyal, K. Winstein, J. Mickens, and H. Balakrishnan 2015. Mahimahi: Accurate record-and-replay for http. USENIX ATC '15, USA, pp. 417–429. USENIX Association.

[24] Pimentel-Niño, M.A., P. Saxena, and M.A. Vazquez Castro 2013. Qoe driven adaptive video with overlapping network coding for best effort erasure satellite links. In *31st AIAA International Communications Satellite Systems Conference*, pp. 5668.

[25] Research, G.V. 2021, May. Edge computing market worth $61.14 billion by 2028 cagr: 38.4%. https://www.grandviewresearch.com/press-release/global-edge-computing-market.

[26] Riiser, H., P. Vigmostad, C. Griwodz, and P. Halvorsen 2013. Commute path bandwidth traces from 3g networks: Analysis and applications. MMSys '13, New York, NY, USA, pp. 114–118. Association for Computing Machinery.

[27] Saxena, P., M. Naresh, M. Gupta, A. Achanta, S. Kota, and S. Gupta 2020. Nancy: Neural adaptive network coding

methodology for video distribution over wireless networks. In *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, pp. 1–6.

[28] Sodagar, I. 2011. The mpeg-dash standard for multimedia streaming over the internet. *IEEE multimedia 18*(4): 62–67 .

[29] Spiteri, K., R. Urgaonkar, and R.K. Sitaraman 2016. Bola: Near-optimal bitrate adaptation for online videos. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pp. 1–9.

[30] Sun, Y., X. Yin, J. Jiang, V. Sekar, F. Lin, N. Wang, T. Liu, and B. Sinopoli. 2016. Cs2p: Improving video bitrate selection and adaptation with data-driven throughput prediction. *Proceedings of the 2016 ACM SIGCOMM Conference* .

[31] Sutton, R.S. and A.G. Barto. 2018. *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book.

[32] Yi, G. October 21–25, 2019, Nice, France. The acm multimedia 2019 live video streaming grand challenge. *The ACM Multimedia 2019 Live Video Streaming Grand Challenge* .

[33] Yin, X., A. Jindal, V. Sekar, and B. Sinopoli 2015. A control-theoretic approach for dynamic adaptive video streaming over http. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, SIGCOMM '15, New York, NY, USA, pp. 325–338. Association for Computing Machinery.

[34] Zhang, A., Q. Li, Y. Chen, X. Ma, L. Zou, Y. Jiang, Z. Xu, and G.M. Muntean. 2021. Video super-resolution and caching–an edge-assisted adaptive video streaming solution. *IEEE Transactions on Broadcasting* .