Deep Reinforcement Learning with Communication Transformer for Adaptive Live Streaming in Wireless Edge Networks

Shuoyao Wang, *Member, IEEE*, Suzhi Bi, *Senior Member, IEEE*, and Ying-Jun Angela Zhang, *Fellow, IEEE*

***Abstract*—The emerging mobile edge computing (MEC) technology has been recently applied to improve the Quality of Experience (QoE) of network services, such as live video streaming. In this paper, we study an energy-aware adaptive live streaming scheme in wireless edge networks. In particular, we aim to design a joint uplink transmission and edge transcoding algorithm maximizing the video followers' QoE, while minimizing the energy consumption of the video streamer. We formulate the problem as a Markov decision process (MDP), and propose a deep reinforcement learning (DRL) based framework, named SACCT, to determine the streamer's encoding bitrate, the uploading power as well as the edge transcoding bitrates and frequency. We decompose the MDP problem into inter-frame and intra-frame problems to address the key design challenges that arise from continuous-discrete hybrid action space, time-varying state and action spaces, and unknown network variation. By doing so, SACCT integrates model-based optimization and model-free DRL to determine the intra-frame continuous resource allocation decisions and the inter-frame discrete bitrate adaptation decisions, respectively. To integrate both the numerical features (e.g., channel gain) and the categorical features (e.g., bitrate), we propose a communication Transformer (CT) as a backbone of SACCT by representing network states as communication tokens and running Transformers to model multi-scale dependencies. Extensive simulations manifest that compared with state-of-the-art approaches, SACCT can provide 128.23% (on average) extra reward. As such, by leveraging joint uplink adaption and edge transcoding, the proposed scheme enables an intelligent wireless network edge with QoE-assured and energy-aware live streaming services.**

***Index terms*—** Adaptive video streaming, Mobile edge computing, Intelligent resource allocation, Transformer, Deep reinforcement learning

This work is supported in part by the National Natural Science Foundation of China (Project number 62101336 and 61871271), the Tencent Rhinoceros BirdsScientific Research Foundation for Young Teachers of Shenzhen University, General Research Funding (Project number 14208017 and 14201920) from the Research Grants Council of Hong Kong, the Guangdong Province Pearl River Scholar Funding Scheme 2018, and the Key Project of Department of Education of Guangdong Province (No. 2020ZDZX3050), and National Key Research and Development Program under Project 2019YFB1803305.

S. Wang and S. Bi are with the College of Electronic and Information Engineering, Shenzhen University, Shenzhen, China (E-mail:sywang@szu.edu.cn, bsz@szu.edu.cn). S. Wang is also with the Guangdong Province Engineering Laboratory for Digital Creative Technology, Shenzhen, China, and the Shenzhen Key Laboratory of Digital Creative Technology, Shenzhen, China. S. Bi is also with the Peng Cheng Laboratory, Shenzhen, China, 518066.

Y. J. Zhang is with the Department of Information Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong (E-mail:yjzhang@ie.cuhk.edu.hk). Y. J. Zhang is also with the Shenzhen Research Institute, The Chinese University of Hong Kong, China.

I. Introduction

A. Motivations and Summary of Contributions

Live video streaming services, such as Twitch and DouYu, receive much attention due to the strong growth in popularity and profits. The COVID-19 pandemic has further resulted in an increasing demand for online streaming and entertainment services. Live video is expected to grow 15-fold by 2022 and reach a 17% share of all Internet traffic [1]. With the rapid increase in 5G-enabled mobile devices, supporting high Quality of Experience (QoE) streaming has become evermore critical in wireless networks.

Adaptive bitrate (ABR) video streaming [2] has been recognized as a prominent video quality adaptation technique to enhance the QoE for users. With ABR a video is divided into several sequential segments and each segment is encoded with different bitrates. For each segment, the video user can request an appropriate bitrate version based on factors like screen resolution, current and predicted channel conditions. To cope with the increasing congestion in the network due to the growth of devices and data, the recent development of *mobile edge computing* (MEC) technology provides a promising solution to enhance ABR streaming under time-varying wireless channels. Specifically, MEC provides storage and computing capabilities at servers located at the edge of radio access network, e.g., cellular base station and WiFi access point. Compared to the conventional ABR streaming implemented at cloud servers, MEC-enabled ABR streaming enjoys significant performance improvement, such as low latency [3], low bandwidth cost [4], improved QoE [5], and targeted services [6].

In this paper, we investigate an end-to-end ABR live streaming scheme in an MEC network in Fig. 1. The mobile streamer, such as the channel owners on Twitch and Tiktok, encodes and uploads the source video to the edge through a wireless uplink. The followers, such as the channel viewers on Twitch and Tiktok, download the video from the same edge through a wireless downlink. While showing potential, the research [3]-[6] to date has mostly focused on downlink bitrate adaption optimization problem. Nonetheless, the stream quality in an end-to-end live video delivery is also constrained by the streamer's uplink [7]. Thus a joint end-to-end optimization over both uplink and downlink would significantly reduce the end-to-end latency as well as the energy consumption [8]. Unfortunately, most existing approaches, investigating solely discrete or continuous action space downlink problems, have failed to address the

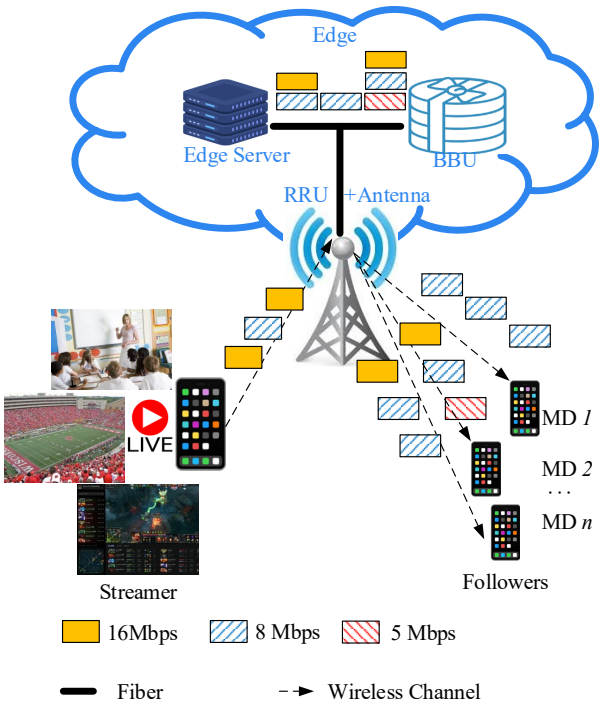


Figure 1: MEC-enabled live video streaming network.

end-to-end transmission problem with joint continuous power allocation and discrete bitrate adaptation.

In particular, we aim to design a decision-making online algorithm to maximize the followers' QoE while minimizing the provider[1] energy consumption. The algorithm is "online", in the sense that the decisions are made without the knowledge of the future channel and follower dynamics. In each time frame, the algorithm jointly determines the optimal encoding bitrate, uploading power, transcoding bitrates, and transcoding frequency decisions. To tackle the problem, we propose a Soft Actor-Critic framework [9] with Communication Transformers (SACCT) that combines the advantages of convex optimization, Transformer, and deep reinforcement learning (DRL). Under fast-varying channel fading and dynamic follower arrivals, SACCT can significantly improve the QoE and reduce the energy consumption. To the authors' best knowledge, this is the first work that jointly considers the uplink bitrate adaptation and edge transcoding for ABR streaming in wireless edge networks. The main contributions of the paper are:

- *Joint adaptive encoding and transcoding*: we formulate a Markov decision process (MDP) problem to maximize the long-term QoE for followers while minimizing the energy consumption of the live streaming network. In particular, the formulation aims to jointly determine the optimal encoding bitrate, uploading power, transcoding bitrates, and transcoding frequency decisions in each time frame, without the knowledge of the future channel and follower dynamics.
- *Problem decomposition*: to cope with the hybrid action space, i.e., the continuous power and frequency allocation decisions together with the discrete bitrate decisions, we decompose the MDP problem into an intra-frame power-frequency resource allocation sub-problem and an inter-frame bitrates adaptation sub-problem.
- *Integrated optimization and DRL*: based on the decomposition, we propose a novel SACCT framework that adopts a soft actor-critic (SAC) learning structure with an embedded convex optimizer. Specifically, the actor module consists of a convex optimizer that determines the optimal streamer uploading power and transcoding frequency, and a stochastic deep neural network that decides the optimal encoding and transcoding bitrates. Compared to the conventional SAC structure, the proposed approach takes advantage of communication system information to acquire power-frequency resource allocation, reduces the output dimension of the deep network, and thus enjoys higher reward and faster convergence during the DRL training process.
- *Integrated DRL and Transformer*: we propose a communication Transformer (CT) as a backbone of both the actor module and the critic module. Compared with conventional deep networks that rely on recursive processing, CT encodes the historical information to capture the multi-scale dependencies, which is more robust against system randomness. Moreover, CT also encodes the hybrid space network states into a unique continuous feature space, leading to a general framework for feature extraction in communication systems.

Compared to state-of-the-art (SOTA) benchmark algorithms and the Twitch solutions, the proposed SACCT achieves a higher QoE and a lower energy consumption under various network conditions.

B. Related Works

1) Adaptive Video Streaming in Wireless Edge Networks: As one of the most frequently launched applications in wireless networks, ABR live streaming service has attracted a great deal of attention. For instance, [10] investigated a joint power allocation and subcarrier assignment scheme for video quality optimization. A model predictive control approach was proposed in [11] to make bitrate adaptation decisions based on channel predictions. Enabled by the recent advance in MEC, online transcoding is a promising solution to satisfy the specific requirement of mobile devices and match the network dynamics. Ref. [12] presented an MEC-enabled video transcoding scheme assuming that the channel quality does not vary with time, in which both bitrate adaptation and spectrum resource allocation are optimized to maximize the user QoE.

To cope with time-varying channel conditions, DRL has recently appeared as a promising alternative to solve bitrate adaptation and segment transcoding problems in wireless edge networks. For instance, [5] proposed a deep Q-network (DQN) based scheme to simultaneously optimize energy consumption and QoE for an MEC-enabled video streaming. To resolve the costly computation problem and balance the exploration-experience tradeoff, recent works have applied the policy-based approaches for ABR streaming optimization [3][6][13]-[16]. For example, [13] proposed an actor-critic network with

[1]For simplicity, we call both the streamer and edge as the provider in the remaining part of this paper.

two actor modules based on periodical experience replay for multi-path video streaming. Ref. [14] investigated the joint computational resource assignment and bitrate adaption optimization problem with the Asynchronous Advantage Actor-Critic algorithm. To evolve the models in real-time, [6] studied a federated reinforcement learning based adaptation framework for mobile video telephony. However, the aforementioned DRL methods suffer a low convergence rate problem. Alternatively, [3] proposed a soft actor-critic network to maximize the video bitrate while minimizing time delays and bitrate variations. The research to date has mostly focused on downlink bitrate adaption optimization problem, where the aforementioned methods investigate fully discrete or continuous action space optimization problem. Such approaches, however, have failed to address the uplink transmission problem with joint continuous power allocation and discrete bitrate adaptation.

2) Transformer: Transformer architectures consist of an encoder module and a decoder module with several encoders/decoders of the same architecture. Each encoder and decoder is composed of a multi-head self-attention layer and a feed-forward neural network. As opposed to recurrent networks (e.g., Long short-term memory LSTM [17]) that process sequence elements recursively and thus are dominated by the most recent context, the Transformer architectures can attend to complete sequences and learn long-range relationships.

Transformer is originally designed in 2017 for natural language processing (NLP) tasks [18], and has demonstrated exemplary performance on a broad range of language tasks, e.g., classification, translation, and question answering. For instance, [19] pre-trains the BERT (Bidirectional Encoder Representations from Transformers) model from unlabeled text by jointly conditioning on bi-direction context. Since 2020, the breakthroughs from Transformer networks in NLP have sparked great interest in the computer vision (CV) community to adapt these models for vision and multi-modal learning tasks. For example, [20] applied a pure Transformer directly to sequences of images and attains SOTA performance on multiple image recognition benchmarks. Apart from classification, Transformer models have been successfully adopted to address more CV problems, such as segmentation [21] and super-resolution [22].

Overall, Transformers enable modeling the historical bitrate selection behaviors and channel variations between the past and current states. Furthermore, the multi-head self-attention layer is more robust than recursively processing against the system randomness in a single frame, which can be very significant in wireless channels. Accordingly, we propose a new paradigm, CT, to represent and process network-level understanding in wireless edge networks. Critically, we propose a tokenizer to convert live streaming network states into communication tokens that describe the past and current network conditions. After tokenization, we feed the communication tokens as input into the Transformer to model multi-scale dependencies among network states and actions.

The rest of the paper is organized as follows. We introduce the network model and formulate the problem as an MDP problem in Section II and III. Through rigorous analysis in Section IV, we decompose the MDP problem into intra-frame resource allocation and inter-frame bitrate adaptation sub-problems. In Section V, we propose a DRL-based framework to solve the sub-problems. The simulation results are presented in Section VI. Finally, the paper is concluded in Section VII.

Notations	
$\mathcal{R}$	The set of the bitrates (Mbps)
β	The playback time of each segment (second)
$r_t^e \in \mathcal{R}$	The encoding bitrate at time t (Mbps)
e_t^e	The encoding energy consumption at time t (Watt×sec)
W	The orthogonal channel bandwidth allocated to the streamer (Hz)
h_t	The wireless channel gain between the streamer and edge at time t
p_t	The uploading power at time t (Watt)
$p^{\max}$	The maximum uploading power (Watt)
N_0	The power spectral density of additive white Gaussian noise (Watt/Hz)
U_t	The uploading transmission rate for time t (bits/second)
τ_t^u	The uploading time at time t
e_t^u	The uploading energy consumption at time t (Watt×sec)
τ_t^c	The transcoding time at time t
e_t^c	The transcoding energy consumption at time t (Watt×sec)
f_t	The edge CPU frequency at time t (Hz)
$\mathcal{J}_t$	The set of followers at the beginning of time t
$\mathcal{J}_t^a$	The set of followers that arrive at the network at time t
$\mathcal{J}_t^d$	The set of followers that departure from the network at time t
$D_{i,t}$	The downlink capacity at time t of the user i (Mbps)
$\eta_{\{1,2,3\},i}$	The user preference parameters of the user i
$\kappa_{\{1,2\}}$	The energy efficiency parameters
$\mathcal{A}_t$	The action space at time t
$\mathcal{S}_t$	The state space at time t
s_t	The state at time t
a_t	The action at time t
s_t^c	The categorical state at time t
a_t^c	The categorical action at time t
s_t^h	The description of the past and current network conditions at time t
M	The state history length
s_t^{Token}	The communication token at time t
$b_{r,t}$	The binary decision on transcoding bitrate r or not at time t
ψ	The scale to obtains the bounded actions

II. System Model

A. Adaptive Bitrate Streaming

In particular, we consider a typical ABR streaming protocol in the live streaming system. Its key features are summarized as follows.

1) Video Segmentation: A source video is partitioned into a sequence of small segments, each containing a piece of the source video with a fixed playback time β (e.g., 2 seconds). Without loss of generality, we set the time frame length as β in this paper.

2) Multi-Bitrate Encoding: A segment is encoded or transcoded into multiple copies with different bitrates, so that a user can select the most suitable bitrate for each segment. For notation simplicity, we denote the finite bitrate set as $\mathcal{R} = \{R_1, R_2, ..., R_{|\mathcal{R}|}\}$ (unit: bits/second). Without loss of generality, we assume that $R_1 > R_2 > ... > R_{|\mathcal{R}|}$. In this paper, the streamer encodes live streaming at each time t with $r_t^e \in \mathcal{R}$ bitrate and the edge can transcode it into $\{r | r < r_t^e, r \in \mathcal{R}\}$ based the user preferences if necessary.

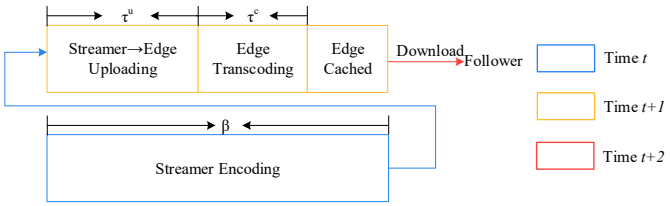


Figure 2: Timeline of the MEC-enabled live video streaming.

B. Network Model

As shown in Fig. 1, we consider a wireless edge network over a time horizon that is divided into T equal time frames. The network consists of a mobile streamer, a Remote Radio Unit (RRU), and a Building Base band Unite (BBU). The RRU and streamer have a single antenna each. Followers arrive at and departure from the network at random times. In particular, an edge server is co-located with the BBU. The timeline of processing each segment is shown in Fig. 2. At the streamer side, the streamer encodes the live streaming segment-by-segment and uploads the segments sequentially to the network edge (Section II.C). At the edge side, the server transcodes the segments into multiple bitrate versions (Section II.D) and the followers download the segment from the edge according to their preference (Section II.E).

C. Video Encoding and Uploading

At the beginning of time frame t, the streamer starts to upload the $(t-1)$-th segment[2] to the edge and encodes the tth segment. Meanwhile, the followers start to download the $(t-2)$-th segment. Correspondingly, the encoding energy consumption at time t is [23]

$$e_t^e = \kappa_1 r_{t-1}^e \beta, \tag{1}$$

where r_{t-1}^e and $\kappa_1 > 0$ denote the encoding bitrate and energy efficiency parameter, respectively. In this paper, we consider that the streamer accesses the uplink spectrum through FDMA to avoid mutual interferences with other users. Let W and h_t denote the orthogonal channel bandwidth allocated to the streamer, and the wireless channel gain between the streamer and edge at time t, respectively. The uploading transmission rate (bits/second) for time frame t is

$$U_t = W \log_2 \left(1 + \frac{p_t h_t}{N_0 W}\right), \tag{2}$$

where p_t and N_0 denote the uploading power at time t and the power spectral density of the noise, respectively. Due to the physical limitation of the on-device battery and antenna, p_t is upper bounded by $p^{\max}$. Then, the uploading time for the $(t-1)$-th video segment at time t is

$$\tau_t^u = \frac{r_{t-1}^e \beta}{U_t}. \tag{3}$$

For notational simplicity, we define

$$g(x) = N_0 W \left(2^{\frac{x}{W}} - 1\right). \tag{4}$$

[2]Since there are multiple segments been encoding, transcoding, and transmission at the same time, we clarify that the tth segment means the segment encoded at time t in this paper.

It follows from equation (3) that the transmit power and the energy consumption are

$$p_t = \frac{1}{h_t} g\left(\frac{r_{t-1}^e \beta}{\tau_t^u}\right), \quad e_t^u = p_t \tau_t^u = \frac{\tau_t^u}{h_t} g\left(\frac{r_{t-1}^e \beta}{\tau_t^u}\right), \tag{5}$$

respectively. Accordingly, e_t^u is convex with respect to τ_t^u.

D. Video Transcoding

After the $(t-1)$-th segment is completely uploaded to the edge, the edge server transcodes the segment into multiple bitrate versions if necessary. Without loss of generality, let $\mathcal{J}_t$ denote the cached bitrate set of the $(t-1)$-th segment after transcoding. We denote by $\mu(r_1, r_2)$ as the required CPU cycles to transcode one segment from r_1 to r_2. Correspondingly, the total required CPU cycles at time t can be expressed as $\sum_{r^c \in \mathcal{J}_t} \mu(r_{t-1}^e, r^c)$. Let f_t denote the edge CPU frequency at time t. Then, the transcoding time and energy consumption at time t are modeled as [24]

$$\tau_t^c = \frac{\sum_{r^c \in \mathcal{J}_t} \mu(r_{t-1}^e, r^c)}{f_t}, \quad e_t^c = \kappa_2 \left(f_t\right)^3 \tau_t^c, \tag{6}$$

respectively. Here, parameter $\kappa_2 > 0$ denotes the transcoding energy efficiency parameter. It follows that the transmission and transcoding times are constrained by

$$\tau_t^u + \tau_t^c \leq \beta \tag{7}$$

to avoid follower playback re-buffering and latency accumulation.

E. Video Downloading

In live streaming applications, it is very common for the mobile devices to dynamically join and quit [25]. That is, the followers arrive at and departure from the network at a random time. We denote by $\mathcal{I}_t^a$ and $\mathcal{I}_t^d$ as the set of followers that arrive at and departure from the network at time t, respectively. Then, the set of followers at the beginning of time $t+1$ can be expressed as

$$\mathcal{I}_{t+1} = \mathcal{I}_t \cup \mathcal{I}_t^a \setminus \mathcal{I}_t^d, \tag{8}$$

where $\mathcal{I}_0 = \varnothing$.

At the beginning of time $t+1$, all the follower $i \in \mathcal{I}_{t+1}$ requests the $(t-1)$-th segment and selects the bitrate from the available set on the edge, based on network conditions and user preferences. Specifically, follower i decides its bitrate request $r_{t+1,i}$ at time $t+1$ to maximize its own QoE by solving

$$r_{t+1,i}|_{\mathcal{J}_t} = \arg \max_{r \in \mathcal{J}_t} \quad q_{i,t+1}(r), \tag{9}$$

where

$$q_{i,t+1}(r) = \eta_{1,i} r - \eta_{2,i} |r - r_{i,t}| - \eta_{3,i} \frac{r}{D_{i,t+1}} \tag{10}$$

denotes the QoE gain with bitrate r for the follower i at time $t+1$ [16]. Here, $D_{i,t+1}$ and $\eta_{k\in\{1,2,3\},i}$ denote the downlink capacity at time $t+1$ and the user preference parameters of the user i, respectively. The three terms in (10) represent the quality gain of the video play, the degradation loss of the quality, and the re-buffering time loss, respectively.

Unfortunately, the network edge cannot afford to produce all possible bitrate versions due to the limited computation resources [26]. As a consequence, the edge optimizes the transcoding set $\mathcal{J}_t$ at time t to maximize the summation of the follower QoEs at time $t+1$, denoted as

$$q(\boldsymbol{r}_t, \mathcal{J}_t) = \sum_{i \in \mathcal{I}_{t+1}} q_{i,t+1}(r_{t+1,i}|_{\mathcal{J}_t}), \tag{11}$$

where $\boldsymbol{r}_t = \left(r_{t,1}, r_{t,2}, ..., r_{t,|\mathcal{I}_t|}\right)$.

III. Problem Formulation

In this paper, we aim to design an online algorithm to maximize the total follower QoE gain while minimizing the provider energy consumption. In particular, we make online decisions in the sense that in each time frame, we optimize the encoding and transcoding bitrates, the uploading power as well as the transcoding frequency decisions based on the observations up to the current time frame, without knowing the future random parameters, such as wireless channel conditions, follower arrivals and departures, and bitrate requests.

At the beginning of time t, the live streaming network determines the online decisions based on its observation of the past and current events, including the current channel h_t, the last encoding bitrate r^e_{t-1}, and the current request bitrates $\boldsymbol{r}_t$. The decision, in turn, affects the last encoding bitrate for future time frames. Thus, the optimal decision is naturally a solution to an MDP, which is defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$. In the following, we discuss the state $\mathcal{S}$, action $\mathcal{A}$, transition function $\mathcal{P}$, reward function $\mathcal{R}$, and discounting factor γ of the considered MDP.

A. State

The system state at time t is described by $s_t = \left(h_t, r^e_{t-1}, \boldsymbol{r}_t\right)$. Since the bitrate requests vary with the number of followers, the state space is also time-varying. For example, when the number of followers is 3, the state space is $\mathcal{S}_t = \mathcal{H} \times \mathcal{R} \times \mathcal{R}^3$; when the number is 1, the state space is $\mathcal{S}_t = \mathcal{H} \times \mathcal{R} \times \mathcal{R}$.

B. Action

Based on s_t, the live streaming network decides the encoding and transcoding bitrates, uploading power, and transcoding frequency at time t. As such, the action at time t can be expressed as $a_t = (r^e_t, p_t, \mathcal{J}_t, f_t)$. Notice that the action space is also time-varying due to the varying $\mathcal{J}_t$. For instance, when the edge does not transcode at time t, i.e., $|\mathcal{J}_t| = 1$, the action space is expressed as $\mathcal{A}_t = \mathcal{R} \times \{(p, f)|\tau^u_t + \tau^c_t \leq \beta, p \leq p^{max}\}$; when the edge transcodes the segment into another 2 bitrates, i.e., $|\mathcal{J}_t| = 3$, the action space is $\mathcal{A}_t = \{(r^e, r^c_1, r^c_2)|r^e \in \mathcal{R}, r^c_1 < r^e, r^c_2 < r^c_1\} \times \{(p, f)|\tau^u_t + \tau^c_t \leq \beta, p \leq p^{max}\}$.

C. Transition Probability

A keen observation on s_t shows that the last encoding bitrate r^e_{t-1} is deterministic given the last action a_{t-1}. Moreover, the channel gain is independent with the encoding and transcoding processes. Therefore, the transition probability can be expressed as:

$$\begin{aligned} P(s_{t+1}|s_t, a_t) &= P(\boldsymbol{r}_{t+1}, h_{t+1}|h_t, \boldsymbol{r}_t, \mathcal{J}_t) \\ &= P(h_{t+1}|h_t) P(\boldsymbol{r}_{t+1}|h_t, \boldsymbol{r}_t, \mathcal{J}_t). \end{aligned} \tag{12}$$

D. Reward Function

The reward function is closely related to the objective of the live streaming network, i.e., the QoE and energy consumption. Without loss of generality, we formulate the reward function at time t as the difference between the QoE and energy consumption at time t. That is

$$v_t(s_t, a_t) = q(\boldsymbol{r}_t, \mathcal{J}_t) - \omega \left[e^e_t + e^u_t + e^c_t\right], \tag{13}$$

where ω determines the energy consumption penalty rate. Since the user preference is unknown to the edge, the edge observes the QoE $q(\boldsymbol{r}_t, \mathcal{J}_t)$ from the followers feedback.

E. Problem Formulation and Discount Factor

Ultimately, the live streaming network aims to find the optimal policy π by solving the following MDP problem:

$$\begin{aligned} \text{(P1)} \quad & \underset{\pi}{\text{maximize}} \quad V_\pi(s_0) = \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t v_t(s_t, a_t)|\pi, s_0\right], \\ & \text{subject to} \quad a \in \mathcal{A}_t, \forall a \sim \pi(s_t), \forall t, \end{aligned} \tag{14}$$

where s_0, $\pi : \mathcal{S} \to \mathcal{A}$, and $\gamma \in (0, 1]$ are the initial network state, the policy that maps a state to an action, and the discount factor that balances the instantaneous and future rewards, respectively.

The time-varying state and action spaces of (P1) renders standard dynamic programming algorithms inapplicable. As a naive solution method, we can project all the state and action spaces into a huge state and action space by padding zeros. In this case, if the channel and bitrate requests transition probabilities, i.e., $P(h_{t+1}|h_t)$ and $P(\boldsymbol{r}_{t+1}|h_t, \boldsymbol{r}_t, \mathcal{J}_t)$, are explicitly and accurately known, (P1) can be solved by dynamic programming, such as sample average approximation and large-scale linear programming [27]. In practice, however, precise distributional information is rarely available. Besides, the huge state and action space causes curse of dimensionality in the dynamic programming methods. To address this issue, we decompose the MDP problem into intra-frame and inter-frame problems in Section IV, and propose a DRL-based framework to solve the decomposed problems efficiently in Section V.

IV. Problem Decomposition

According to the Bellman Equation, the value function V_π in (P1) can also be expressed as:

$$\begin{aligned} \text{(P2)} \quad & V_\pi(s_t) = \\ & \max_{a_t} \; v_t(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1}|s_t, a_t}\left[V_\pi(s_{t+1})\right], \\ & \text{s.t.} \quad \frac{r^e_{t-1}\beta}{W \log_2\left(1 + \frac{p_t h_t}{N_0 W}\right)} + \frac{\sum_{r^c \in \mathcal{J}_t} \mu(r^e_{t-1}, r^c)}{f_t} \leq \beta, \\ & \quad p_t \leq p^{max}. \end{aligned} \tag{15}$$

To tackle the hybrid discrete and continuous action space of $\mathcal{A}_t$, in the following, we decompose (P2) into intra-frame resource allocation and inter-frame bitrate adaptation problems. The bitrate adaptation problem is an inter-frame problem that decides the discrete encoding bitrates, which are coupled in sequential time frames due to the fact that the bitrate decision affects the future ones in subsequent time frames for a smooth playback, i.e., $\eta_2|r - r_{-1}|$ in (10). On the other hand, the resource allocation problem is an intra-frame problem in the sense that the continuous resource allocation is only related to the energy consumption in the current frame.

A. Intra-frame Problem

We note that the resource allocation problem to optimize $\{p_t, f_t\}$ given $\{r_t^e, \mathcal{J}_t\}$ can be efficiently calculated with a standard convex optimizer. In particular, given the value of $\{r_t^e, \mathcal{J}_t\}$ in (P2), we can express the optimal resource allocation problem at time t as:

$$\begin{aligned}(\text{P3}) \max_{p_t, f_t} \quad & q(\boldsymbol{r}_t, \mathcal{J}_t) - \omega\Big(\kappa_1 r_t^e \beta + \frac{\tau_t^u}{h_t} g(\frac{r_{t-1}^e \beta}{\tau_t^u}) \\ & + \kappa_2 \left(f_t\right)^3 \tau_t^c\Big) + \gamma \mathbb{E}_{s_{t+1}|s_t, a_t}\left[V_\pi(s_{t+1})\right], \\ \text{s.t.} \quad & \frac{r_{t-1}^e \beta}{W \log_2\left(1 + \frac{p_t h_t}{N_0 W}\right)} + \frac{\sum_{r^c \in \mathcal{J}_t} \mu(r_{t-1}^e, r^c)}{f_t} \leq \beta, \\ & p_t \leq p^{max}.\end{aligned} \tag{16}$$

According to the QoE gain function (9) and the transition probability (12), $q(\boldsymbol{r}_t, \mathcal{J}_t)$, $\kappa_1 r^e \beta$, s_{t+1}, and $P(\boldsymbol{r}_t|h_t, \boldsymbol{r}_{t-1}, \mathcal{J}_t)$ are constants given $\{r_t^e, \mathcal{J}_t\}$. Moreover, $P(h_{t+1}|h_t)$ and $P(s_{t+1}|s_t, a_t)$ are independent of (p_t, f_t). Therefore, the expectation $\mathbb{E}_{s_{t+1}|s_t, a_t}\left[V_\pi(s_{t+1})\right]$ is independent of (p_t, f_t). Hence, the optimal solution (p_t^*, f_t^*) can be derived by solving the following intra-frame energy minimization problem

$$\begin{aligned}(\text{P4}) \max_{p, f} \quad & -\omega \frac{\tau^u}{h} g(\frac{r_{-1}^e \beta}{\tau^u}) - \omega \kappa_2 \left(f\right)^3 \tau^c, \\ \text{s.t.} \quad & \frac{r_{-1}^e T}{W \log_2\left(1 + \frac{p h}{N_0 W}\right)} + \frac{\sum_{r^c \in \mathcal{J}_t} \mu(r_{-1}^e, r^c)}{f} \leq \beta, \\ & p \leq p^{max}.\end{aligned} \tag{17}$$

We drop the subscript t in (P4) for simplicity of exposition, for the above problem optimizes the resource allocation within a particular time frame. For notational simplicity, we denote $\kappa_3 = \sum_{r^c \in \mathcal{J}_t} \mu(r_{-1}^e, r^c)$, which is a constant given $(r_t^e, \mathcal{J}_t)$. Similar to (3) and (6), we derive

$$p = \frac{N_0 W}{h}\left(2^{\frac{r_{-1}^e \beta}{\tau^u W}} - 1\right), f = \frac{\kappa_3}{\tau^c}. \tag{18}$$

Notice that both $p \to \tau^u$ and $f \to \tau^c$ are one-to-one mappings. Substituting (18) into (P4) and replacing the decision variables with (τ^u, τ^c), we have

$$\begin{aligned}(\text{P5}) \min_{\tau^u, \tau^c} \quad & \frac{\tau^u}{h} g(\frac{r_{-1}^e \beta}{\tau^u}) + \kappa_2 \frac{\left(\kappa_3\right)^3}{\left(\tau^c\right)^2}, \\ \text{s.t.} \quad & \tau^u + \tau^c \leq \beta, \\ & \frac{1}{h} g(\frac{r_{-1}^e \beta}{\tau^u}) \leq p^{max},\end{aligned} \tag{19}$$

which is a convex minimization problem with linear and convex constraints. Thanks to the convexity, the function $f(r_t^e, \mathcal{J}_t)$ can be efficiently calculated with a standard convex optimizer, e.g., an interior point optimizer.

B. Inter-frame Problem

For notational simplicity, we denote the immediate reward at time t when taking the discrete encoding rate action $(r_t^e, \mathcal{J}_t)$ as

$$\begin{aligned}v_t(s_t, (r_t^e, \mathcal{J}_t)|p_t^*, f_t^*) = & \, q(\boldsymbol{r}_t, \mathcal{J}_t) - \\ & \omega\left(\kappa_1 r_t^e \beta + \frac{\tau_t^u}{h_t} g(\frac{r_{t-1}^e \beta}{\tau_t^u|p_t^*}) + \kappa_2 \left(f_t\right)^3 \tau_t^c|f_t^*\right),\end{aligned} \tag{20}$$

where (p_t^*, f_t^*) is the optimal solution to Problem (P3). Accordingly, solving (P2) is equivalent to finding the optimal bitrate adaptation decisions:

$$\begin{aligned}(\text{P6}) \max_{r_t^e, \mathcal{J}_t} \quad & v_t(s_t, (r_t^e, \mathcal{J}_t)|p_t, f_t) + \gamma \mathbb{E}_{s_{t+1}|s_t, r_t^e, \mathcal{J}_t}\left[V_\pi(s_{t+1})\right], \\ \text{s.t.} \quad & \frac{r_{t-1}^e \beta}{W \log_2\left(1 + \frac{p_t h_t}{N_0 W}\right)} + \frac{\sum_{r^c \in \mathcal{J}_t} \mu(r_{t-1}^e, r^c)}{f_t} \leq \beta, \\ & p_t \leq p^{max}.\end{aligned} \tag{21}$$

Leveraging the DRL technique, we will propose an SACCT algorithm in Section V to construct a policy π that maps from the state s_t to the optimal bitrate adaptation decisions $(r_t^e, \mathcal{J}_t)$ in (P6).

V. The SACCT Algorithm

In this section, we employ a DRL-based framework to solve (P6), which is data-driven and requires no distributional information about the wireless channel and followers in the future. Note that the conventional DRL methods cannot be directly applied to our problem. This is because the dimensions of the state and action space, being proportional to the cardinality of $|\mathcal{I}_t|$, keep varying with the random arrival and departure of the followers. To address the problem, we propose a novel DRL framework with an embedded Communication Transformer (CT). Our proposed framework is based on the SOTA soft actor-critic algorithm [9], and leverages the technique of Transformers [18]. The schematics of the proposed SACCT algorithm are shown in Fig. 3.

A. Soft Value Functions

The objective of the live streaming network is to find the policy π to maximize the long-term expected reward. Inspired by [9], to ensure that the agent can explore continually, an entropy term $\mathcal{H}(\pi(a|s)) = -\log \pi(a|s)$ is added to the reward in this paper, where $\pi(a|s)$ denotes the probability of taking action a given state s under policy π. The soft objective with the expected entropy is written as

$$J(\pi) = \mathbb{E}^\pi\{\sum_{t=0}^{\infty} \gamma^t \left[v_t(s_t, a_t) - \alpha \log \pi(a_t|s_t)\right]\}, \tag{22}$$

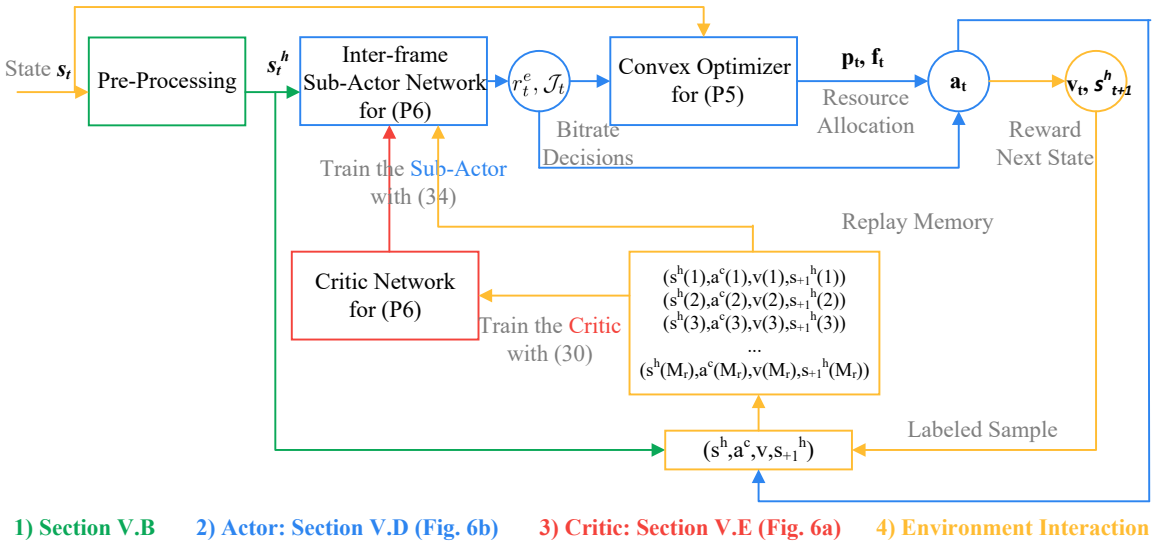


Figure 3: Soft actor-critic with communication Transformer (SACCT) framework overview.

where α is the temperature parameter that controls the stochasticity of the optimal policy. We define the soft Q-value function given the initial state-action pair (s, a) as

$$Q^{\pi}(s,a) = \mathbb{E}^{\pi}\{\sum_{t=0}^{\infty}\gamma^t\left[v_t(s_t,a_t) - \alpha\log\pi(a_t|s_t)\right]|s_0 = s, a_0 = a\}. \tag{23}$$

In the actor-critic framework, the learning process alternates between policy improvement, i.e., actor module with parameter ϕ, and policy evaluation, i.e., critic module with parameter θ, in the direction of maximizing $Q_\theta^{\pi_\phi}(s,a)$. Both the parameters are randomly initialized following the standard normal distribution at time 0.

B. Pre-Processing

To tackle the time-varying state space, we first leverage categorical state representation. In particular, we transform the numerical state into the categorical state:

$$s_t \rightarrow s_t^c = [h_t, r_{t-1}^e, n_{1,t}, ..., n_{|\mathcal{R}|,t}], \tag{24}$$

where $n_{k,t} = |\{i|r_{t,i} = R_k, i \in \mathcal{I}_t\}|$ for $k = 1, ..., |\mathcal{R}|$ is the number of the requests for R_k at time t. The space of the numerical state is $\mathcal{S}^c = \mathcal{H} \times \mathcal{R} \times \{0, 1, 2, \dots\}^{|\mathcal{R}|}$, which is unified in all the frames. Accordingly, we combine the past M categorical states to obtain a description of the past and current network conditions[3]

$$s_t^h = (s_t^c, ..., s_{t-M+1}^c). \tag{25}$$

Similarly, we also transform the numerical action into the categorical action:

$$a_t \rightarrow a_t^c = [r_t^e, b_{1,t}, ..., b_{|\mathcal{R}|,t}], \tag{26}$$

where $b_{k,t} = \begin{cases} 1, & \text{if } R_k \in \mathcal{J}_t, \\ 0, & \text{otherwise,} \end{cases}$ for $k = 1, ..., |\mathcal{R}|$. Similarly, the space of the numerical action is $\mathcal{A}^c = \mathcal{R} \times \{0, 1\}^{|\mathcal{R}|}$, which is unified in all the frames. Here, s_t^h and (s_t^h, a_t^c) are the inputs of the actor network and the critic network, respectively.

C. Communication Transformer

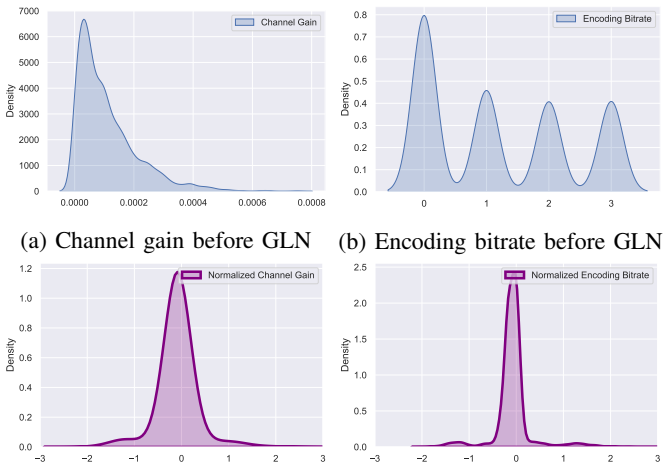
Different from the conventional DRL algorithms that use a multi-layer perceptron neural network (MLP) [9] or recurrent neural network (RNN) [16] as the network architecture, this work proposes a CT to learn the non-local semantic concepts. As shown in Fig. 5, the proposed CT consists of a tokenizer to generate the communication tokens and a Transformer to encode the tokens into the encoded state.

Our intuition is that a sentence with a few "words" (or communication tokens) suffices to describe the current network conditions. The network condition momentum rather than the absolute value of the network conditions is important. This motivates a departure from the sequential array representation; instead, we propose a group layer normalization (GLN) to convert the state into a compact set of semantic tokens. We then feed these tokens into a Transformer, a multi-head self-attention module widely used in NLP to capture token interactions.

CT encodes the historical information (e.g.,the network variation and follower user preferences), and captures multi-scale dependencies. Unlike MLPs and RNNs, the proposed CT can better handle the following three challenges:

- CT adaptively encodes the historical information according to the current network condition by attending to the frames with different weights, instead of treating all historical frames equally;
- CT encodes semantic communication concepts from the bitrate as well as the transmission and transcoding latency, which are category and numerical features, respectively. This is a general framework to address the

[3]The longer the state history length M, the better the decision can be made by the agent. However, a large M will induce a large state space and decelerate the learning speed of algorithm. Without loss of generality, we set $M = 30$ in our simulation.



(a) Channel gain before GLN (b) Encoding bitrate before GLN

(c) Channel gain after GLN (d) Encoding bitrate after GLN

Figure 4: Illustration of the proposed GLN.

heterogeneous data challenges for machine learning tasks in communication systems;

- CT establishes the multi-scale relationships among the historical channel conditions and follower behaviors through the multi-head computation in token-space. This is also a general framework to cope with sequential decision problems, such as MDP problems.

1) Tokenizer with GLN: The object of the tokenizer is to convert live streaming network states into communication tokens that describe the past and current network conditions. Formally, we define

$$s_t^{\text{Token}} = \left(\frac{h_t - \mu_t^h}{\sigma_t^h}, \frac{r_{t-1}^e - \mu_t^e}{\sigma_t^e}, \frac{n_{1,t} - \mu_t^n}{\sigma_t^n}, ..., \frac{n_{|\mathcal{R}|,t-M+1} - \mu_t^n}{\sigma_t^n}\right) \tag{27}$$

with

$$\begin{aligned}
\mu_t^h &= \frac{1}{M}\sum_{i=0}^{M-1} h_{t-i},\ \sigma_t^h = \sqrt{\frac{1}{M}\sum_{i=0}^{M-1}(h_{t-i} - \mu_t^h)^2}, \\
\mu_t^e &= \frac{1}{M}\sum_{i=1}^{M} r_{t-i}^e,\ \sigma_t^e = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(r_{t-i}^e - \mu_t^e)^2}, \\
\mu_t^n &= \frac{1}{M|\mathcal{R}|}\sum_{i=0}^{M-1}\sum_{j=1}^{|\mathcal{R}|} n_{j,t-i}, \\
\sigma_t^n &= \sqrt{\frac{1}{M|\mathcal{R}|}\sum_{i=0}^{M-1}\sum_{j=1}^{|\mathcal{R}|}(n_{j,t-i} - \mu_t^n)^2}.
\end{aligned} \tag{28}$$

The conventional layer normalization suffers gradient vanishing and gradient exploding problems. Through the GLN (Fig. 4), the heterogeneous features are aligned with the same mean and variance. Therefore, the Transformer can integrate heterogeneous features simultaneously.

2) Transformer: After tokenization, we then need to model interactions between these communication tokens. To achieve multi-head self-attention, the communication tokens serve as

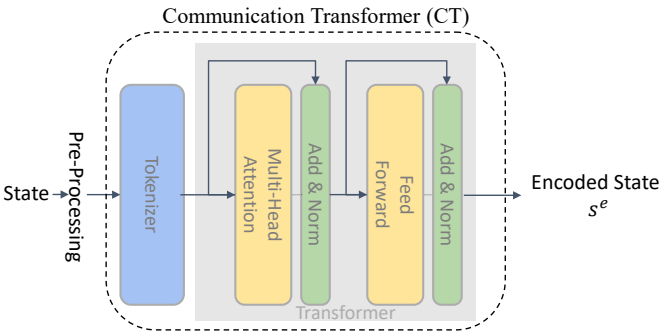


Figure 5: Communication Transformer.

the input to the Transformer. Afterwards, the Transformer returns the description of the past and current network conditions as the encoded state, which is denoted as $s_t^e \in \mathbb{R}^{d_{\text{dim}}}$. Here, d_{dim} is a hyper-parameter. In particular, we employ a standard Transformer encoder. Each layer has a standard architecture, and consists of a normalization multi-head self-attention module and a normalization feed forward network. We refer to the open source implementation for the detailed definition of the architecture, following the one described in [18]. An advantage of this intentionally simple setup is that scalable NLP Transformer architectures and their efficient implementations can be used almost out of the box.

D. Critic Module

Based on CT, we build up the critic network as shown in Fig. 6a. In particular, the critic module takes (s^h, a^c) as inputs, and returns the corresponding expectation of $Q^\pi(s^h, a^c)$, including both the long-term reward and entropy. On the left hand side of Fig. 6a, the CT backbone encodes s^h into the encoded state s^e. On the right hand side, the embedding layer [28] encodes a^c into the encoded action $a^e \in \mathbb{R}^{d_{\text{dim}}}$. The encoded a^e and s^e are further concatenated. Then, the final evaluation $Q^\pi(s^h, a^c)$ is computed by a 2-layer perceptron with GeLU activation function [19] and hidden dimension d_{dim}.

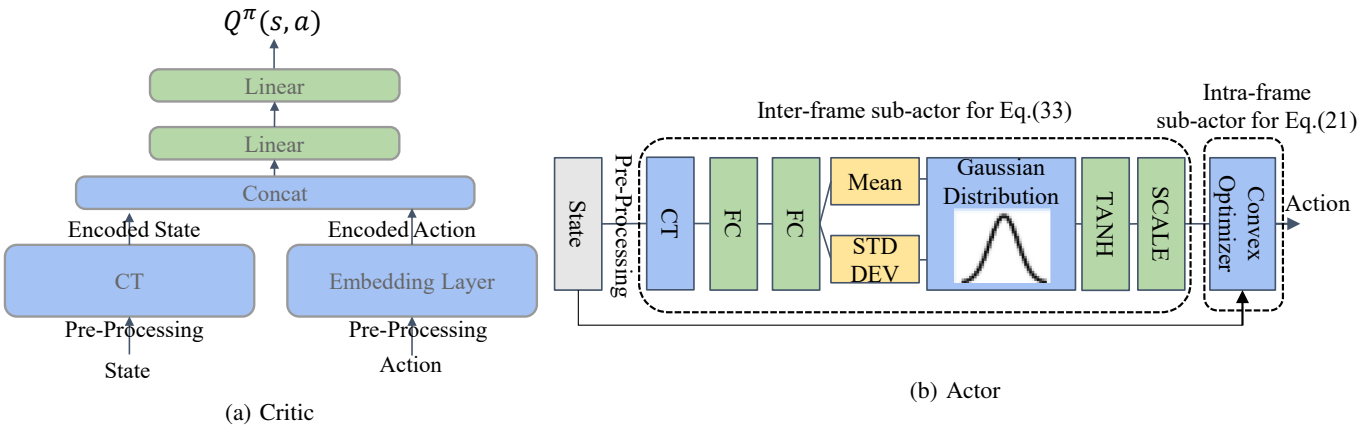
Specifically, the soft Q-value is estimated as $Q^\pi(s^h, a^c) \approx Q_\theta(s^h, a^c)$. Similar to the conventional value-based deep reinforcement learning algorithm, the soft Q-function parameters θ can be trained to minimize the soft Bellman residual

$$L_c(\theta) = \mathbb{E}_{s_t^h, a_t^c}\left[\frac{1}{2}\left(Q_\theta(s_t^h, a_t^c) - \hat{Q}_{\bar{\theta}}(s_t^h, a_t^c)\right)^2\right], \tag{29}$$

with

$$\begin{aligned}
&\hat{Q}_{\bar{\theta}}(s_t^h, a_t^c) = v_t(s_t^h, a_t^c) + \\
&\gamma \mathbb{E}_{a_{t+1}^c \sim \pi_\phi(s_{t+1}^h)}\left[Q_{\bar{\theta}}(s_{t+1}^h, a_{t+1}^c) - \alpha \log\left(\pi_\phi\left(a_{t+1}^c | s_{t+1}^h\right)\right)\right].
\end{aligned} \tag{30}$$

The update makes use of the target soft $\hat{Q}$ in (30) with parameters $\bar{\theta}$ that are obtained as an exponentially moving average of the soft Q weights, which has been shown to stabilize training [29]. According to the experience replay technique, we maintain a replay memory with length M_r and stores the newly generated M_r tuples $\{(s_\tau^h, a_\tau^c, v_\tau(s_\tau^h, a_\tau^c), s_{\tau+1}^h)\}_{\tau=t-M_r+1,...,t}$ at each time step. In particular, with an initially empty memory,



(a) Critic

(b) Actor

Figure 6: The critic and actor network structures.

Algorithm 1 Soft Actor-Critic with Communication Transformer (SACCT)

Initialization: critic network parameter θ, actor network parameter ϕ

Output: Control actions $\{r_t^e, p_t, \mathcal{J}_t, f_t\}$

1: **for** $t = 1, 2, \ldots, K$ **do**
2: Observe the environment (state) $s_t = (h_t, r_{t-1}^e, \boldsymbol{r}_t)$ and compute s_t^h using (24)(25);
3: Calculate the mean and standard deviation $(\boldsymbol{\mu}^t, \boldsymbol{\sigma}^t)$ for the unbounded inter-frame action $\left(\hat{r_t^e}, \hat{b}_{1,t}, ..., \hat{b}_{|\mathcal{R}|,t}\right)$ with ϕ_t;
4: Generate the unbounded action and project it to the bounded action $(r_t^e, b_{1,t}, ..., b_{|\mathcal{R}|,t})$;
5: Compute (p_t, f_t) by optimizing Problem (P5) given $(r_t^e, b_{1,t}, ..., b_{|\mathcal{R}|,t})$;
6: $a_t \leftarrow a_t^c$
7: Perform a_t, receive the reward $v_t(s_t, a_t)$, observe s_{t+1} and stored the tuple $(s_t^h, a_t^c, v_t(s_t^h, a_t^c), s_{t+1}^h)$ in the replay memory;
8: **if** the replay memory is full **then**
9: Sample a batch from the replay memory and update the critic network and the actor network according to $L_c(\theta)$ in (29) and $L_a(\phi)$ in (34), respectively;
10: **end if**
11: $t \leftarrow t + 1$.
12: **end for**

we start training after collecting M_r samples. When the replay memory is full, in each iteration, we sample a mini-batch from the replay memory and update the critic network by minimizing its average $L_c(\theta)$ in (29).

E. Actor Module

The actor module consists of an intra-frame sub-actor and an inter-frame sub-actor. As shown in Fig. 6b, the inter-frame sub-actor takes s_t^h as input, and returns the encoding and transcoding bitrates $(r_t^e, b_{1,t}, \ldots, b_{|\mathcal{R}|,t})$. The intra-frame sub-actor takes $(r_t^e, b_{1,t}, \ldots, b_{|\mathcal{R}|,t}, s_t)$ as input, and returns the transmission power and transcoding frequency (p_t, f_t).

To balance the exploration-exploitation tradeoff, the inter-frame sub-actor consists of a CT-MLP network and an action generator. Taking s_t^h as the input, the CT-MLP outputs the mean $\boldsymbol{\mu}_t^a$ and the standard deviation $\boldsymbol{\sigma}_t^a$, where $\boldsymbol{\mu}_t^a(i)$ and $\boldsymbol{\sigma}_t^a(i)$ are the mean and variance of the i-th element of the action generator. Then, the generator generates the unbounded actions with the Gaussian distributions as the following

$$\begin{aligned} \hat{r_t^e} &\sim \mathcal{N}\left(\boldsymbol{\mu}_t^a(1), \boldsymbol{\sigma}_t^a(1)\right), \\ \hat{b}_{1,t} &\sim \mathcal{N}\left(\boldsymbol{\mu}_t^a(2), \boldsymbol{\sigma}_t^a(2)\right), \\ &\ldots \\ b_{|\mathcal{R}|,t} &\sim \mathcal{N}\left(\boldsymbol{\mu}_t^a(1+|\mathcal{R}|), \boldsymbol{\sigma}_t^a(1+|\mathcal{R}|)\right). \end{aligned} \tag{31}$$

Afterwards, the actor applies $\tanh(\cdot)$ and scaling operations to the unbounded inter-frame actions and obtains the bounded actions $\left(r_t^e, b_{1,t}, ..., b_{|\mathcal{R}|,t}\right)$[4], i.e.,

$$\begin{aligned} r_t^e &= \lfloor \frac{|\mathcal{R}|}{2} \times \tanh(\hat{r_t^e}) + \frac{|\mathcal{R}|}{2} \rceil, \\ b_{k,t} &= \lfloor 0.5 \times \tanh(\hat{b}_{k,t}) + 0.5 \rceil, \forall k = 1, ..., |\mathcal{R}|. \end{aligned} \tag{32}$$

Overall, the input-output relation of the actor is expressed as

$$\pi_\phi : s_t^h \rightarrow a_t^c \sim \lfloor \boldsymbol{\psi} \times \tanh\left(\mathcal{N}(\boldsymbol{\mu}_t^a | s_t^h, \boldsymbol{\sigma}_t^a | s_t^h)\right) + \boldsymbol{\psi} \rceil, \tag{33}$$

where $\boldsymbol{\psi} = \left(\frac{|\mathcal{R}|}{2}, \underbrace{0.5, ..., 0.5}_{|\mathcal{R}|}\right)$.

The well-known universal approximation theorem claims that an MLP with a sufficient number of neurons can accurately approximate any continuous mappings if proper activation functions are applied at the neurons. Similar to [9], the optimal policy is learned by minimizing

$$\begin{aligned} L_a(\phi) = \mathbb{E}_{s_t^h, a \sim \left(\text{Standard}\left(\pi_\phi(s_t^h) + \epsilon\right)\right)} \big[& \alpha \log \pi_\phi(a | s_t^h) \\ & - Q_\theta^\pi(s_t^h, a) \big], \end{aligned} \tag{34}$$

where Standard$(\cdot)$ is the standardization function to ensure that the summation of the probabilities is 1, and the noise vector $\epsilon \sim \mathcal{N}(0, 1)$ is sampled from the standard normal distribution [9]. Similarly to the critic network, in each iteration, we sample a mini-batch from the replay memory and update the actor network by minimizing its average $L_a(\phi)$ in (34).

Finally, the intra-frame sub-actor takes the inter-frame action as well as the network state as the input, and outputs both the inter-frame and intra-frame actions by solving (P5). The details of the algorithm are shown in Algorithm 1.

[4]The operator $\lfloor x \rceil$ returns the nearest integer of x.

VI. Performance Evaluation

In this section, we investigate the performance of our proposed SACCT framework. All the computations are executed on a machine with an AMD EPYC 7742 CPU, a NVIDIA A100 Tensor Core GPU, and 64GB RAM. The DRL algorithms are constructed using Pytorch for computational speed boost. In the following, we first introduce the experiment setup. We evaluate SACCT to answer the following questions:

- How does SACCT coordinate the transcoding and transmission tasks in an on-line manner, especially under highly dynamic wireless channel and random followers? (Section VI.B)
- Do the proposed problem decomposition technique improve the QoE and reduce the energy consumption? (Section VI.C)
- How does CT contribute to the QoE improvement and the energy consumption reduction? (Section VI.D)

The complete source code implementing SACCT and the benchmark algorithms are available at https://github.com/wsyCUHK/SACCT.

A. Experiment Setup

Unless otherwise stated, we consider a Rayleigh fading channel model, where the channel gain $h_t = \bar{h}\xi$. Here, $\bar{h}$ denotes the average channel gain determined by the location of the user i, and $\xi \sim \text{Exponential}(1)$ denotes an independent exponential random variable of unit mean. Specifically, $\bar{h}$ follows the free-space path loss model, i.e.,

$$\bar{h} = G_A \left(\frac{3 \times 10^8}{4\pi f_c d} \right)^{d_e}, \tag{35}$$

where G_A and d denote the antenna gain and the distance from the streamer to the transmit antenna, respectively. Here, d_e denotes the path loss exponent. In addition, we set $G_A = 3$ and $d_e = 2.8$ in our simulation. The distance d from the streamer to the base station is generated from a truncated Gaussian distribution as $d = \min\left(\max\left(X, 10\text{m}\right), 70\text{m}\right)$, where $X \sim \mathcal{N}(40\text{m}, 10\text{m})$ is a Gaussian random variable.

There are four video layers of the video, with $R_1 = 16$ Mbps (2K), $R_2 = 8$ Mbps (1080P), $R_3 = 5$ Mbps (720P), and $R_4 = 1$ Mbps (360P) [16]. The video transcoding CPU cycles are modeled as $\mu(r_1, r_2) = \eta_4 |r_1 - r_2|$ [5]. The downlink capacity $D_{i,t}$ is generated from a truncated Gaussian distribution as $D_{i,t} = \min\left(\max\left(Y, 1\text{Mbps}\right), 9\text{Mbps}\right)$, where $Y \sim \mathcal{N}(5\text{Mbps}, 2\text{Mbps})$ is a Gaussian random variable. The followers are divided into three types, namely laptop, smart phone, and tablet followers. For each type, the follower arrivals follow a Poisson distribution and each follower departures from the live streaming with a fixed probability. The setting of the arrival rates, departure rates, and QoE parameters for the three types are listed in Table I. The values of the other parameters are also listed in Table I, which are equal for all the followers. For performance comparison, we consider four benchmark methods.

- Twitch Event [30]: for the members of Twitch Partnership Program, the streamer uploads the streaming with the highest bitrate, a.k.a., source quality. The server transcodes the segments into multiple bitrate versions. In our experiment, the streamer uploads R_1 segments and the edge server transcodes them into $R_2, ..., R_{|\mathcal{R}|}$.
- Twitch Starter [30]: for the new streamers, the default encoder of Twitch provides fixed bitrate encoding and deliver the streaming to the followers without transcoding. In our experiment, the streamer uploads R_2 segments without any transcoding at the edge server.
- SAC with decomposition (SAC-w) [4]: the provider uses Soft Actor-Critic framework to estimate the state function and find the inter-frame policy that maximizes the defined reward. Similar to SACCT, the intra-frame actions in SAC-w are determined by the proposed convex optimizer.
- SAC without decomposition (SAC-wo): the provider uses Soft Actor-Critic framework to estimate the state function and find both the inter-frame and intra-frame action policies that maximize the defined reward.

According to [14], the critic and actor networks are both two-layer MLP networks for SAC baseline in our simulation.

Table I: Simulation Parameters

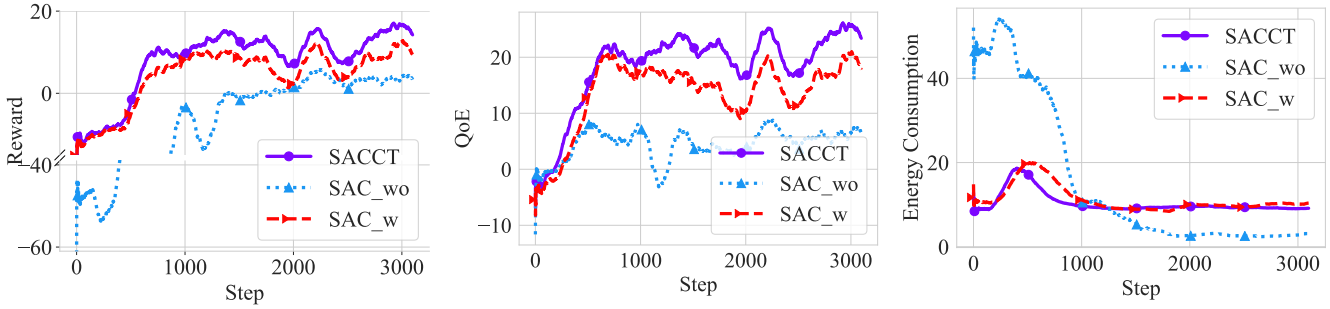
Type	Laptop	Smart Phone	Tablet
Arrival Rate	0.05	λ(default=0.1)	0.1
Departure Rate	0.05	0.1	0.1
η_1	2	1	2
η_2	5	0.05	5
η_3	2	0.5	0.5
$\kappa_2 = 10^{-26}/(\text{Hz})^3$	$\beta = 2$ second		$M_r = 30$
$M = 30$	$\eta_1 = 0.3/(\text{MHz})^2$		$d_{\text{dim}} = 128$
$N_0 = 10^{-10}$W/Hz	$W = 2$ MHz		$\kappa_1 = 10^{-9}$ W/Hz

Table II: Mean and variance after Step 2000 in Fig. 7

Metric		Mean	Variance	Var/Mean
Reward	SACCT	**12.25**	7.45	**0.60**
	SAC-w	8.84	7.49	0.84
	SAC-wo	2.06	3.78	1.83
QoE	SACCT	21.54	6.95	**0.32**
	SAC-w	16.42	8.49	0.51
	SAC-wo	5.32	2.54	0.47
Energy	SACCT	9.29	0.03	**3.46e-3**
	SAC-w	7.58	0.29	3.86e-3
	SAC-wo	3.26	1.26	0.38

B. Instance-Level Evaluation

Before the evaluation of the learned policy, we first evaluate the convergence of the proposed SACCT in Fig. 7. For fair comparison, we first randomly generate 20 random seeds and train SACCT model under each seed for 10,000 time frames. Then, we train the benchmark algorithms, i.e., SAC-w and SAC-wo, under the same random seeds for 10,000 time

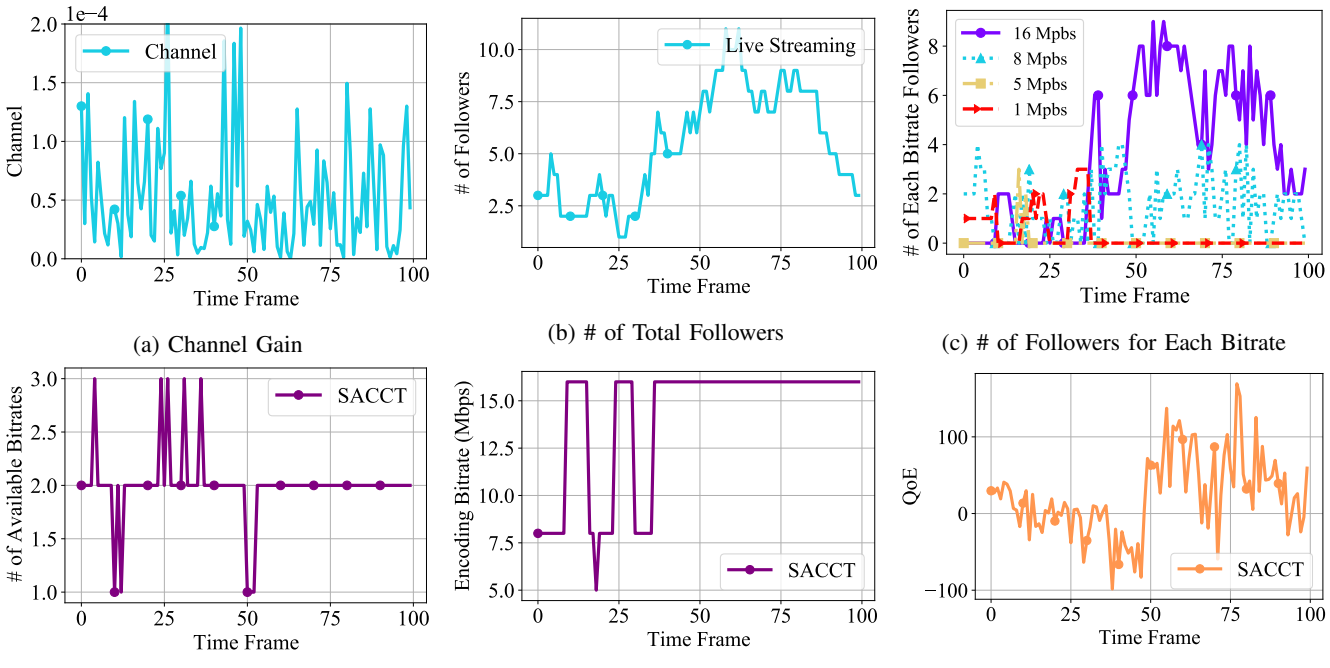


(a) Average Reward (b) Average QoE (c) Average Energy Consumption

Figure 7: The convergence performance.

frames. We set the episode numbers as 200 and the batch size as 256 for both SACCT and the baseline algorithms. Based on the above parameter setting, we plot the reward, QoE, and energy consumption achieved by SACCT and the baseline algorithms, as the time processed. In Fig. 8f, each point is an average performance of 20 random seeds, while the value of each random seed is a moving-window average of 200 time frames. Moreover, we also plot the mean and variance comparsion of the three methods after gettting converged in Tab. II.

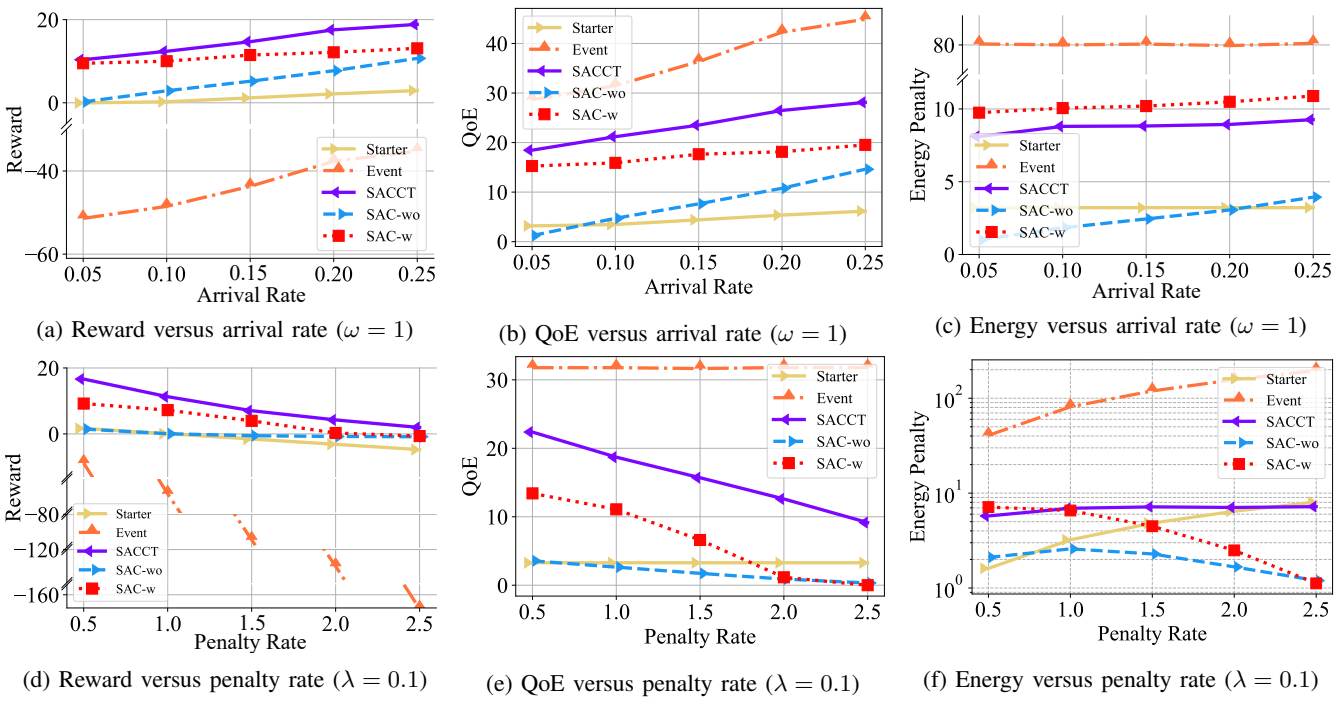
It can be seen from Fig. 7a that both SACCT and the baseline algorithms become more experienced as training proceeds and collects higher expected average rewards. Thanks to the decomposition technique and convex optimizer, SACCT and SAC-w outperform SAC-wo both in terms of the convergence speed and average reward. We also observe from Tab. II that SACCT acheives the highest reward and the lowest normalized variances, i.e., variance/mean, of all the three metrics, highlighted in bold. By employing the Transformer backbone, SACCT outperforms SAC-w in terms of the average reward and robustness. In particular, we can observe in Fig. 7b-7c, all methods improve the total QoE by proper bitrate uplink adaptation and edge transcoding from time 0 to 500. Intuitively, the energy consumption increases accordingly. Thanks to the optimality ensured by the convex optimizer, the energy consumptions under SACCT and SAC-w are significantly less than SAC-wo. Afterwards, SACCT and SAC-w continuously improve the QoE as well as the energy consumptions, while SAC-wo focuses on reducing the energy consumption at the cost of poor QoE. After 1000 time frames, SACCT and SAC-w converge to the learned policies, where the QoEs vary marginally with the random arrival and departure and the average energy consumptions are almost constants. Overall, under dynamic wireless channel and follower arrival/deparutre,



(a) Channel Gain (b) # of Total Followers (c) # of Followers for Each Bitrate

(d) # of Available Bitrates (e) Encoding Bitrate (f) Energy Consumption Penalty

Figure 8: A video streaming instance with learned policy by the SACCT.



(a) Reward versus arrival rate ($\omega = 1$)

(b) QoE versus arrival rate ($\omega = 1$)

(c) Energy versus arrival rate ($\omega = 1$)

(d) Reward versus penalty rate ($\lambda = 0.1$)

(e) QoE versus penalty rate ($\lambda = 0.1$)

(f) Energy versus penalty rate ($\lambda = 0.1$)

Figure 9: The performance versus arrival rate and energy penalty rate.

SACCT owns a fast convergence and can achieve both a higher QoE gain and a lower energy consumption.

The learned policy by the proposed SACCT is shown in Fig. 8. When the number of followers is small, we observe from 10 to 18 that the provider selects a high encoding bitrate under a large channel gain in Fig. 8e and 8a. From 19 to 25, we observe that the provider selects a low bitrate under a small channel gain in Fig. 8e and 8a. When the number of follower increases from 20 to 40, the provider prefers to transcode the segments into multiple bitrate versions (Fig. 8d). When the number of followers is large, the provider selects a high encoding bitrate in regardless of the channel gain. Overall, *the learned policy adapts the segment bitrates based on the dynamic channel conditions and followers, and thus achieves a high QoE in most of the test frames.*

C. System-Level Evaluation

In the following, we compare the average reward, QoE gain, and energy consumption under different network setups. Specifically, we investigate the performance metrics versus different smart phone user arrival rates λ and energy consumption penalty rates ω, respectively. In Fig. 9a-9c, we fix $\omega = 1$/watt and vary arrival rate from 0.05 to 0.25. In Fig. 9d-9f, we fix $\lambda = 0.1$ and vary ω from 0.5/watt to 2.5/watt.

From Fig. 9a, the proposed SACCT outperforms the four baseline schemes and the gap widens when the arrival rate increases from 0.05 to 0.25. Compared to the improvements from SAC-wo to SAC-w, the improvements from SAC-w to SACCT are $9.25\% \sim 108.66\%$ on the reward. This is because when the number of followers is small, the "conservative" methods, such as Twitch Starter, enjoy a low energy consumption; when the number of followers are large, the "conservative" methods suffer a low QoE. In contrast, SACCT encodes low bitrate segments and seldom transcodes the segments for energy saving when the number of followers is small; when the number is large, SACCT encodes high bitrate segments and transcodes the segments into popular bitrates for QoE improvement (Fig. 9c). Consequently, we can observe from Fig. 9b-9c that the QoE gain under SACCT enjoys a linear growth when the arrival rate increases from 0.05 to 0.25, while the energy consumption increases relative slowly, especially when the arrival rate is larger than 0.1.

In Fig. 9d-9f, we investigate the performance metrics when $\lambda = 0.1$ and the penalty rate ω increases from 0.5 to 2.5, where the proposed SACCT earns the best reward for each ω. Specifically, we observe from Fig. 9f that the rewards under SACCT and the benchmark methods monotonically decrease when ω increases from 0.5 to 2.5. The reasons are as follows: i) SACCT adopts the transcoding and transmission decisions to keep a near-constant energy consumption penalty when the penalty rate is increasing. As a result, although the QoE decreases linearly with the penalty rate, the rate of reward decline is getting slower and slower when the penalty rate increases from 0.5 to 2.5. ii) The Twitch Starter and Event baselines intuitively share a constant QoE performance when the arrival rate is fixed, and suffer a linear penalty growth with the penalty rate. Therefore, the Twitch baselines suffer a linear reward decline. iii) SAC-w selects very light transcoding and transmission decisions when the penalty rate is high, which is believed as an overreaction. Therefore, SAC-w degrades to SAC-wo and thus suffers a significant performance loss when the penalty rate is high.

Overall, the experiments show that SACCT can adapt the encoding, uploading, and transcoding to the wireless channel

Table III: Comparisons of reward, QoE, and energy consumption versus different machine learning architectures.

Model			$\lambda = 0.1, \omega = 1$			$\lambda = 0.2, \omega = 1$			$\lambda = 0.1, \omega = 2$		
Architecture	# of Layer	# of Head	Reward	QoE	Energy	Reward	QoE	Energy	Reward	QoE	Energy
SACCT	1	1	11.54	19.47	7.93	15.08	22.72	7.64	3.66	12.66	9.00
SACCT	1	2	11.82	20.31	8.48	16.45	24.71	8.25	3.34	12.73	9.38
SACCT	1	3	12.06	20.36	8.29	16.35	24.50	8.14	3.65	15.05	11.40
SACCT	1	6	12.06	20.97	8.90	17.27	25.45	8.18	3.88	13.36	9.47
SACCT	2	6	12.08	19.99	7.90	17.00	25.27	8.26	**4.74**	13.73	8.98
SACCT	3	6	**12.26**	21.23	8.96	**17.44**	25.53	8.09	4.16	13.27	9.10
SACCT	4	6	12.09	20.63	8.53	16.06	23.61	7.54	3.96	15.43	11.47
FC+FC (SAC-w)			10.40	16.35	5.94	11.14	17.14	6.00	1.41	14.02	12.61
Linear Attention+FC			9.94	19.62	9.67	11.44	18.56	7.12	2.13	13.60	11.47
Convolution Layer+FC			10.74	19.56	8.81	13.86	21.24	7.38	2.05	13.16	11.11
LSTM+FC			11.07	19.01	7.93	14.30	22.45	8.14	2.37	14.13	11.76

and the bitrate request under different arrival rates and penalty rates. The performance advantage is especially notable when the network is highly dynamic, i.e., the arrival rate λ is large. Compared with the SOTA DRL algorithm SAC, SACCT always achieves higher reward, QoE, and energy consumption in our experiments. The widening gap with increasing λ indicates that *the proposed CT can embed the heterogeneous communication features, encode historical information, and capture multi-scale dependencies.* In addition, the current Twitch Event solution suffers a significant energy consumption, especially when the channel is poor. The current Twitch Starter solution shares a low QoE performance, especially when the streamer is popular. This demonstrates that SACCT enables *QoE-assured adaptive live streaming services with energy consumption requirements, leading to a higher profit for the service providers.*

D. Transformer-Level Evaluation

In this subsection, we further show the impact of the propose CT. All the points in the table are the average performance of 20 random seeds after convergence. In Table III, we exhibit the comparison by replacing the propose CT with different machine learning architectures. The major findings from the architecture evaluation results can be summarized as follows:

- The proposed CT outperforms all the baselines, indicating that CT enjoys a more principled way to leverage heterogeneous local and non-local network informations for bitrate adaptation, transmission and transcoding resource allocation.
- We can see that the CTs with different number of the Transformer layers achieve the similar performance, up to some random variations (12.06 to 12.26 with the number of head$=6$, $\lambda = 0.1$ and $\omega = 1$). It indicates that the proposed architecture is robust and can be safely extended to other Transformer-based backbones. Due to the computation efficiency, we adopt one Transformer layer with 6-head self-attention in the proposed SACCT framework.
- Among these baselines, we can find that the overall performance order is as follows: CT, LSTM, convolution network, and dense network. It indicates that the better performances can be achieved by modeling *the long-term and short-term dependencies*. On one hand, the performance increases with the number of self-attention heads in SACCT, which validates the intuition that both the longer-term dependencies and the shorter-term dependencies are important for decision making, i.e., multi-head attention pays attention to different parts of the tokens and thus improves the overall performance. On the other hand, the 1-head-1-layer CT network outperforms the LSTM network, which validates *the heterogeneous features are well aligned through the proposed group layer normalization*.

Recall that one of the main challenge in wireless live streaming is the continuous fast channel variation and discrete bitrate requests dynamics, which suffers gradient vanishing and gradient exploding problems in conventional machine learning frameworks. The experiment has demonstrated that CT can efficiently integrate the heterogeneous information. Moreover, thanks to the robustness, CT is a general machine learning backbone to improve the network services in wireless edge networks, not limited to live streaming services.

VII. Conclusions and Potential Applicability

In this paper, we have studied an uplink transmission and edge transcoding joint optimization problem for ABR live streaming in a wireless edge network. We formulated an MDP problem that maximizes the long-term QoE for the followers while minimizing the energy consumption of the provider. To tackle the problem, we proposed a DRL-based framework with convex optimizer and Transformer, combining the advantage of convex optimization and machine learning, called SACCT. Moreover, the proposed CT, a backbone of SACCT, represents network states as communication tokens and runs Transformers to densely model the historical informations, which successfully integrates both the numerical and categorical

features. Our extensive simulations have demonstrated that the proposed algorithm outperforms baseline solutions in terms of both QoE gain and energy consumption. The results indicated that uplink adaptation enables QoE-assured live streaming services with lower energy consumption.

The proposed CT not only integrates the numerical and categorical features but also relates the spatially distant concepts, which is a general framework and can be adapted to numerous signal processing problems in communication networks. For example, it is interesting to extend the proposed SACCT framework to design *computation offloading* strategy in wireless edge networks, where the offloading decision is binary (discrete) and the resource allocation decisions are continuous. Likewise, CT can also be extended to optimize the node selection and cache replacement in mobile networks. In collaborative *edge caching* problems, the cache state and node selection decision are both discrete, and the content popularity and link rate are continuous. In addition, CT can further be extended to design *edge video analytics* system where the video is delivered for computer vision tasks and thus can be compressed before transmission. In this case, the integer and continuous variables correspond to the video compression rate and wireless channel gain, respectively.

By carefully setting the integer variables to represent the offloading, node selection, compression decisions, CT is applicable to jointly optimize the integer decisions and continuous resource allocations. Since the discrete action space can be extremely large in some applications, CT might be polished to improve the sample efficiency and convergence rate. We hope CT would become an important backbone of future machine learning design in communication systems.

REFERENCES

[1] Cisco Visual Networking Index Report. Accessed on 10th Jan 2021. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.pdf

[2] Y. Sani, A. Mauthe, and C. Edwards, "Adaptive bitrate selection: A survey," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2985–3014, 2017.

[3] F. Fu, Y. Kang, Z. Zhang, F. R. Yu, and T. Wu, "Soft actor-critic drl for live transcoding and streaming in vehicular fog computing-enabled iov," *IEEE Internet of Things Journal*, pp. 1–1, 2020.

[4] K. Du, A. Pervaiz, X. Yuan, A. Chowdhery, Q. Zhang, H. Hoffmann, and J. Jiang, "Server-driven video streaming for deep learning inference," in *Proceedings of ACM SIGCOMM'20*, New York, NY, USA, 2020, p. 557–570.

[5] Y. Guo, F. R. Yu, J. An, K. Yang, C. Yu, and V. C. M. Leung, "Adaptive bitrate streaming in wireless networks with transcoding at network edge using deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 3879–3892, 2020.

[6] H. Zhang, A. Zhou, J. Lu, R. Ma, Y. Hu, C. Li, X. Zhang, H. Ma, and X. Chen, "Onrl: Improving mobile video telephony via online reinforcement learning," in *Proceedings of ACM MobiCom '20*, New York, NY, USA, 2020.

[7] Z. Wang, Y. Cui, X. Hu, X. Wang, W. T. Ooi, and Y. Li, "Multilive: Adaptive bitrate control for low-delay multi-party interactive live streaming," in *Proceedings of IEEE INFOCOM*, 2020, pp. 1093–1102.

[8] C. Zhang and J. Liu, "On crowdsourced interactive live streaming: A twitch.tv-based measurement study," in *Proceedings of ACM NOSSDAV 2015*, New York, NY, USA, 2015, p. 55–60.

[9] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proceedings of ICML 2018*, 2018, pp. 1861–1870.

[10] M. Rugelj, U. Sedlar, M. Volk, J. Sterle, M. Hajdinjak, and A. Kos, "Novel cross-layer QoE-aware radio resource allocation algorithms in multiuser ofdma systems," *IEEE Transactions on Communications*, vol. 62, no. 9, pp. 3196–3208, 2014.

[11] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over HTTP," in *Proceedings of ACM SIGCOMM'15*, vol. 45, no. 4, New York, NY, USA, Aug. 2015, p. 325–338.

[12] D. Wang, Y. Peng, X. Ma, W. Ding, H. Jiang, F. Chen, and J. Liu, "Adaptive wireless video streaming based on edge computing: Opportunities and approaches," *IEEE Transactions on Services Computing*, vol. 12, no. 5, pp. 685–697, 2019.

[13] Y. Guan, Y. Zhang, B. Wang, K. Bian, X. Xiong, and L. Song, "Perm: Neural adaptive video streaming with multi-path transmission," in *Proceedings of IEEE INFOCOM*, 2020, pp. 1103–1112.

[14] J. Luo, F. R. Yu, Q. Chen, and L. Tang, "Adaptive video streaming with edge caching and video transcoding over software-defined mobile networks: A deep reinforcement learning approach," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 1577–1592, 2020.

[15] Y. Zhang, Y. Zhang, Y. Wu, Y. Tao, K. Bian, P. Zhou, L. Song, and H. Tuo, "Improving quality of experience by adaptive video streaming with super-resolution," in *Proceedings of IEEE INFOCOM*, 2020, pp. 1957–1966.

[16] Y. Zhang, P. Zhao, K. Bian, Y. Liu, L. Song, and X. Li, "Drl360: 360-degree video streaming with deep reinforcement learning," in *Proceedings of IEEE INFOCOM*, 2019, pp. 1252–1260.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of NeurIPS 2017*, Red Hook, NY, USA, 2017, p. 6000–6010.

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of ACL 2019*, Minneapolis, Minnesota, Jun. 2019, pp. 4171–4186.

[20] B. Wu, C. Xu, X. Dai, A. Wan, P. Zhang, M. Tomizuka, K. Keutzer, and P. Vajda, "Visual transformers: Token-based image representation and processing for computer vision," 2020. [Online]. Available: https://arxiv.org/abs/2006.03677

[21] L. Ye, M. Rochan, Z. Liu, and Y. Wang, "Cross-modal self-attention network for referring image segmentation," in *Proceedings of CVPR 2019*, 2019, pp. 10 502–10 511.

[22] F. Yang, H. Yang, J. Fu, H. Lu, and B. Guo, "Learning texture transformer network for image super-resolution," in *Proceedings of CVPR 2020*, 2020, pp. 5791–5800.

[23] S. Yang and G. Sun, "Power-aware adaptive video streaming from the set-top-box to mobile devices," in *Proceedings of IEEE ICCE-TW 2016*, 2016, pp. 1–2.

[24] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[25] Y. Chen, Y. Liu, B. Zhang, and W. Zhu, "On distribution of user movie watching time in a large-scale video streaming system," in *Proceedings of IEEE ICC 2014*, 2014, pp. 1825–1830.

[26] J. Yang, J. Luo, D. Meng, and J. Hwang, "Qoe-driven resource allocation optimized for uplink delivery of delay-sensitive vr video over cellular network," *IEEE Access*, vol. 7, pp. 60 672–60 683, 2019.

[27] S. Wang, S. Bi, and Y. A. Zhang, "Demand response management for profit maximizing energy loads in real-time electricity market," *IEEE Transactions on Power Systems*, vol. 33, no. 6, pp. 6387–6396, 2018.

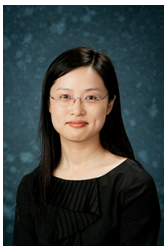
[28] "Embedding — PyTorch 1.8.1 documentation," accessed on 1st June 2021. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.nn.Embedding.html

[29] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[30] Twitch. Twitch Partner Program Overview. Accessed on 9th Jan 2021. [Online]. Available: https://help.twitch.tv/s/article/partner-program-overview?language=en_US

Shuoyao Wang (S'18–M'20) received the B.Eng. degree (with first class Hons.) and the Ph.D degree in information engineering from The Chinese University of Hong Kong, Hong Kong, in 2013 and 2018, respectively. From 2018 to 2020, he was an senior researcher with the Department of Risk Management, Tencent, Shenzhen, China. Since 2020, he has been with the College of Electronic and Information Engineering, Shenzhen University, Shenzhen, China, where he is currently an Assistant Professor. His research interests mainly involve machine learning and optimization theory in Multimedia Communications, Wireless Communications, and Image Processing.

Suzhi Bi (S'10–M'14–SM'19) received the B.Eng. degree in communications engineering from Zhejiang University in 2009, and the Ph.D. degree in information engineering from The Chinese University of Hong Kong in 2013. From 2013 to 2015, he was a post-doctoral research fellow with the Department of Electrical and Computer Engineering, National University of Singapore. Since 2015, he has been with the College of Electronics and Information Engineering, Shenzhen University, China, where he is currently an Associate Professor. His research interests mainly involve in the optimizations in wireless information and power transfer, mobile computing, and wireless sensing. He received 2019 IEEE ComSoc Asia-Pacific Outstanding Young Researcher Award, and Best Paper Awards of IEEE SmartGridComm 2013 and IEEE/CIC ICCC 2021. He is an Associate Editor of IEEE Wireless Communications Letters.



Ying-Jun Angela Zhang (S'00–M'05–SM'10–F'20) is a Professor at Department of Information Engineering, The Chinese University of Hong Kong. Her research interests are in optimization and learning in wireless communication systems and smart power grids. She is now an Associate Editor-in-Chief of IEEE Open Journals of the Communication Society and a Member of IEEE ComSoc Fellow Evaluation Committee. Previously, she has served as the Chair of the Executive Editorial Committee of IEEE Transactions on Wireless Communications and a Founding Chair of IEEE ComSoc Smart Grid Communications Technical Committee. She is the co-recipient of the 2011 Marconi Prize Paper Award in Wireless Communications, the 2013 SmartGridComm Best Paper Award, and the 2014 IEEE ComSoc Asia Pacific Board Outstanding Paper Award. She won the 2006 Hong Kong Young Scientist Award as the only winner from engineering science. She is a Fellow of IEEE and a Distinguished Lecturer of IEEE ComSoc.