Risk-Aware Contextual Learning for Edge-Assisted Crowdsourced Live Streaming

Xingchi Liu*, Mahsa Derakhshani†, Lyudmila Mihaylova*, Sangarapillai Lambotharan†
*Department of Automatic Control and Systems Engineering, University of Sheffield, S1 3JD, UK
†Signal Processing and Networks Research Group, Wolfson School of Mechanical, Electrical and Manufacturing Engineering, Loughborough University, LE11 3TU, UK
Email: xingchi.liu@sheffield.ac.uk; M.Derakhshani@lboro.ac.uk; l.s.mihaylova@sheffield.ac.uk
S.Lambotharan@lboro.ac.uk.

Abstract—**This paper proposes an edge-assisted crowdsourced live video transcoding approach where the transcoding capabilities of the edge transcoders are unknown and dynamic. The resilience and trustworthiness of highly unstable transcoders in decision making are characterized with mean-variance-based measures to avoid making highly risky decisions. The risk level of each device's situation is assessed and two upper confidence bounds of the variance of transcoding performance are presented. Based on the derived bounds and by leveraging the contextual information of devices, two risk-aware contextual learning schemes are developed to efficiently estimate the transcoding capabilities of the edge devices. Combining context awareness and risk sensitivity, a novel transcoding task assignment and viewer association algorithm is proposed. Simulation results demonstrate that the proposed algorithm achieves robust task offloading with superior network utility performance as compared to the linear upper confidence bound and the risk-aware mean-variance upper confidence bound-based algorithms. In particular, an epoch-based task assignment strategy is designed to reduce the task switching costs incurred in assigning the same transcoding task to different transcoders over time. This strategy also reduces the computational time needed. Numerical results confirm that this strategy achieves up to 86.8% switching costs reduction and 92.3% computational time reduction.**

Index Terms—**Reinforcement learning, edge computing, task offloading, risk-awareness, contextual learning**

I. Introduction

The revolution of the mobile Internet driven by the powerful mobile devices and social networks has greatly enriched the sources of video platforms. As an outcome of the revolution, crowdsourced live streaming platforms (CLSP) such as Twitch, TikTok, and Periscope have emerged as a new type of video platforms, that not only serve tremendous viewers all over the world but also receive live videos from various sources in the crowd [1], allowing a growing number of people to broadcast their live videos over the Internet.

This work was supported in part by the Royal Academy of Engineering under the Leverhulme Trust Research Fellowship scheme (Derakhshani-LTRF1920\16\67), and the Engineering and Physical Sciences Research Council (EPSRC) through EP/R006385/1 "Communications Signal Processing Based Solutions for Massive Machine-to-Machine Networks (M3NETs)" as well as EP/T013265/1 "ShiRAS. Towards Safe and Reliable Autonomy in Sensor Driven Systems" and USA National Science Foundation under Grant NSF ECCS 1903466 supporting ShiRAS. For the purpose of open access, the author has applied a Creative Commons Attribution (CC BY) licence to any Author Accepted Manuscript version arising

However, due to the heterogeneity of broadcasters' devices, different quality versions of live videos are created [2]. As a result, there is a strong need to transcode the original live videos into several industrial standard representations and to serve viewers with a set of proper versions of representations. Providing the adaptive bit rate (ABR) service [3] can bring massive computational demands due to real-time processing requirements. For instance, until 2022, there are about 9.2 million broadcasters and 140 million viewers, which are active on Twitch monthly, and there are an average of more than 100,000 live channels concurrently at any point in time [4].

Therefore, instead of building private data centres to facilitate ABR, cloud computing has become a natural solution to perform transcoding because of its powerful computing ability and the 'pay as you go' feature. Furthermore, the emergence of cloud computing releases CLSP from building large, expensive private data centres. In such a system, the CLSP controller will decide the number of representations that need to be transcoded for each broadcaster based on parameters such as viewer capacity, playback delay, bandwidth consumption etc. The original live videos will be directly transmitted to the cloud data centre for transcoding. When multiple versions are generated in the cloud, content delivery networks will be utilized to deliver proper versions of live videos to the corresponding viewers.

On the downside, in current CLSPs, the cloud transcoding is not able to provide the ABR service to most of the broadcasters. For instance, in Twitch.TV, only the premium broadcasters have access to the ABR service, and for the rest of the broadcasters, only the original versions are available for their viewers [1]. The reason behind is that a general cloud instance can only deal with at most two transcoding tasks simultaneously. Therefore, an enormous cost will be incurred when a large number of original live videos are scheduled for transcoding. Moreover, in cloud transcoding systems, the cloud data centre can be far from the viewers or the broadcasters, which can cause high latency. This problem can be further magnified considering the fact that most of CLSPs enable an interactive live chat service which is also latency-sensitive as compared to the traditional live streaming platforms.

The development of edge computing (also know as fog computing) has brought a potential transcoding solution for CLSP. Since edge computing [5] is more suitable for real-time processing and low-latency applications, it can be treated as a viable replacement [6], [7] to address the shortcomings of the cloud transcoding. Moreover, edge-assisted transcoding can lead to lower latency and avoid the network traffic traversing through the core network since different versions of videos can be created at the network edge.

Edge transcoding systems have been proposed to leverage the computational resources at the user end [8]–[12]. However, most of the works relies on solving optimization problems in the presence of perfect knowledge of the system parameters and the performance of edge devices. However, acquiring such knowledge might not be practically feasible in the live streaming systems. Particularly, existing works do not consider the risk of assigning a transcoding task to a device with highly unstable transcoding performance.

This paper proposes a novel concept to quantify the edge devices' transcoding capabilities leveraging their contextual information. This concept considers not only the average performance of the transcoders but also the risk of performance variations, which account for the design of a joint transcoding task assignment and viewer association algorithm. To the best of authors' knowledge, this paper is the first work to consider both risk sensitivity and context awareness in an online task offloading problem. The contributions of this paper are summarised below:

- The joint transcoding task assignment and viewer association problem is formulated to maximize the long-term network utility considering the cost and average latency.
- The transcoding capability is proposed to identify proper transcoders with both high transcoding quality and low transcoding performance variation. The transcoding quality is modelled as a linear function of transcoder's context information and the performance variation is represented as the variance of transcoding outcomes.
- To estimate and learn the unknown transcoding capability, two upper confidence bounds (UCBs) of the transcoding performance variation are derived. Applying the confidence bounds and by involving a contextual UCB of the transcoding quality, a risk-aware contextual online learning scheme is designed to learn the transcoding capabilities of edge devices. The idea of considering both the context and risk awareness helps designing efficient and robust task offloading schemes and reduce the risk of assigning tasks to devices that cannot ensure stable performance.
- Based on the learnt transcoding capabilities, a novel joint transcoding task assignment and viewer association algorithm is designed. Particularly, an epoch-based strategy is designed to reduce the switching costs of task assignments.
- Numerical results based on various settings demonstrate that the proposed algorithm achieves significant network utility improvement while keeping the switching costs of transcoding task assignments competitively low.

The remainder of this paper is organised as follows. Section II discusses the related works. Section III describes the model of the edge transcoding system. In Section IV, an optimization problem is formulated to maximize the overall network utility. The designed risk-aware contextual learning for edge transcoding algorithm is described in Section V, followed by the numerical results in Section VI. The conclusions are drawn in Section VII.

II. Related work

In this section, we review the works in both cloud and edge-based live video transcoding systems and discuss the challenges in designing the edge-based transcoding schemes.

A. *Cloud Transcoding*

Due to the powerful computing ability and the 'pay as you go' feature of cloud computing, previous works tended to implement the transcoding system with the help of the cloud resources and designed various quality of experience (QoE) metrics for cloud transcoding systems [13]–[20]. For instance, [13] designed a cloud-based scheme to transcode crowdsourced video contents. The QoE is a function of the bit rate of the received live stream and the broadcaster's popularity. [14] proposed a cloud transcoding scheme considering delay constraints. In this work, the QoE was defined as a non-decreasing concave function of the received bit rate. In [16], a new live streaming framework was designed to minimize the content delivery delay with cloud transcoding. [17] proposed a cloud transcoding scheme for both delay-tolerant and delay-sensitive videos with different priorities. In [19], a scheme was designed to limit the peak power consumption while maximizing the total processing capacity in a server with heterogeneous processors using dynamic programming. [20] designed recurrent network and convolutional network based approaches to forecast the approximate transcoding resources which is reserved for transcoding and to maximize the quality of service (QoS).

B. *Edge and Crowdsourced Transcoding*

Due to the abundance of concurrent live broadcasters and the heterogeneity of source contents, a substantial amount of transcoding tasks are generated which are delay-sensitive and computationally intense. As a result, even cloud transcoding cannot meet these requirements with affordable cost [21]. Therefore, edge computing has been considered as a viable replacement because of its fast processing and quick application response time [22]. However, it is highly challenging to achieve optimal transcoding task assignment and viewer association due to the massive heterogeneous video contents and diversified QoE demands [23]. In [24], a collaborative joint caching and transcoding scheme was proposed to reduce the backhaul link usage and the viewer perceived delay. In [9], a reinforcement learning (RL)-based scheme was designed to solve the edge transcoding decision-making problems. To better schedule edge transcoding under large state space, deep RL was used to explicitly accommodate personalized QoE optimization for CLSP services [20], [23], [25]–[27]. In addition, [21] combined both the cloud and the edge resources to collaboratively transcode live videos from multiple broadcasters.

Table I: A summary of recent works on edge-assisted task assignment and live video transcoding

Key feature	Ref	Objectives	Technique summary
Context-aware	[27]	Task deadline, energy consumption	Deep reinforcement learning
	[36]	System cost, latency, viewer fairness	Combinatorial MAB
	[34]	Platform utility	MAB
	[31]	Transcoder income, overall value of task	Genetic algorithm
	[37]	QoE, energy consumption	Deep reinforcement learning
	[20]	QoS, operation cost	Deep learning-based greedy assignment
	[26]	Network utility	Deep reinforcement learning
Risk-aware	[7]	Transcoder serving time, switching costs	Optimization
	[11]	Platform utility	Complementary geometric programming
	[38]	Cumulative mean-variance	Risk-aware MAB
	[39]	Operation cost	Branch and check

In [6], a case study was presented for Twitch demonstrating that with the advance of personal computing devices, a significant fraction of CLSP viewers' devices potentially have appropriate computing resources for real-time transcoding. In addition, the viewers have already expressed the willingness to support the broadcasters and the CLSPs in terms of donation and subscription [21]. Thus, the cost by involving them into transcoding can be much lower as compared to general edge computing. These studies demonstrate the potential of incentivizing the viewer devices to do transcoding. However, since the viewer devices are not professional and can be highly heterogeneous, their performances may be unknown and unstable. In [7], the transcoder selection relies on the prior data collection and analysis, which may become inaccurate over time. Therefore, optimal online decision-making strategies which can learn devices' performance and select devices which are more capable for transcoding are highly desirable.

Such edge-assisted crowdsourced transcoding systems are similar with the crowdsourcing systems which exploit the collective intelligence of crowd, provide an effective paradigm for large-scale data acquisition and distributed computing [28]–[30]. In the edge-assisted transcoding system, the viewer devices assigned with transcoding tasks can be treated as the crowd workers for computing. The crowdsourcing system has been introduced into many areas such as text translation, consumer research, and hiring workers for software development.

There have been extensive works on task assignment problems in the crowdsourcing systems [31]. As a classical decision-making model, multi-armed bandit (MAB) has been used to model the task assignment problems. For instance [32] proposed a UCB-based task assignment algorithm with a limited budget for crowdsensing. [33] modelled the crowdsourcing system as an MAB and proposed a bounded ε-first algorithm to maximize the overall utility of completing a number of tasks. [34] proposed a budget-limited UCB-based greedy approach to learn the worker performance of a crowdsourcing system and to select workers with high performance to maximize the long-term utility. In [35], a hierarchical context-aware learning algorithm is proposed to learn and estimate the worker's context-specific performance in mobile crowdsourcing.

We have reviewed some recent papers in Table I on live video transcoding offloading with edge computing. Most works either consider the context information of the system and edge devices when making decisions [31], [36], or directly formulate the feedback of the decision as a function of context information. In [38], the risk of performance fluctuation is considered in the MAB problem. However, this work only studied a standard MAB that at each time only one arm will be played and the arm selection is not aware of the context information of each arm. In addition, there are limited works studying to consider the risk of decisions which can lead to high performance fluctuation. However, these works only consider known problem-specific risk such as the edge devices' online stability and the probability of failure [7], [11], [39]. None of these works directly models feedback of decision as a function of risk to make the task offloading risk-aware.

Overall, although the task assignment problem in edge-assisted computing systems has been studied in recent years, which can be used in edge-assisted crowdsourced transcoding systems, several technical problems have not yet been addressed. First, the existing task assignment decision-making models are not practical enough since the tasks were assumed to arrive sequentially and the number of tasks need to be assigned per time slot is fixed. In addition, most of existing works only focus on identifying the device with the highest average performance for task offloading. Although some works considered risk in the edge computing framework, the risk is not integrated into the feedback of the decision. Moreover, the switching costs of assigning a task to different edge devices over time have not been considered yet. In particular, to the best of our knowledge, there is no work studying to combine risk and context awareness in the edge-assisted task assignment problem.

III. System Model

We consider an edge-assisted CLSP as illustrated in Figure 1. The raw live videos from the broadcasters are first uploaded to the regional data centers which are responsible for video transmission and transcoding task assignment. The pre-processed videos (e.g. segmented video chunks) with the original bit rates are then forwarded to edge transcoders and the transcoded videos will be transmitted to the end viewers. Table II summarizes the symbols used in this article.

Define a set of broadcasters, i.e., $\mathcal{I} = \{1, 2, \cdots, I\}$. The bit rate of the original live video of broadcaster $i \in \mathcal{I}$ in time slot t is defined as B_i^t. Moreover, consider there are J^t viewers in time slot t and $V_{i,j}^t$ is a binary parameter indicating

Table II: Symbols and Notations

Symbol	Meaning
i or $\mathcal{I}$	index (or set) of broadcaster(s)
j or $\mathcal{J}^t$	index (or set) of viewer(s) in time slot t
y or $\mathcal{Y}$	index (or set) of video representation(s)
f or $\mathcal{F}$	index (or set) of edge device(s)
B_i^t	Bit rate of live video from broadcaster i
b_y	Bit rate of representation y
$I_{i,y,f,j}^t$	Indicator variable of transcoding task assignment and the viewer association
ω_f^t	Transcoding capability of device f
$c_{i,j,y}^t$	Cost of transcoding a video from broadcaster i to representation y at edge device f
$q_{i,j,y}^t$	QoE of a viewer j watching a transcoded video from broadcaster i
ξ_j^t	Transmission delay
δ_j^t	Transcoding delay
D_j^t	Overall latency
U_i^t	Network utility
r_f^t or $\mathbf{R}^t$	Transcoding outcome (set)
$\tilde{\gamma}_f^t$ or γ_f^t	(Estimated) Transcoding quality, $\gamma_f^t = \mathbb{E}(r_f^t)$
σ_f	Variance of the transcoding outcome
ρ	Risk-tolerance factor
$\mathbf{x}_f$	Contextual information of edge transcoder f
$\boldsymbol{\theta}^*$ or $\tilde{\boldsymbol{\theta}}^t$	Unknown (Estimated) context weight vector
$(s_f^t)^2$	Empirical variance of the transcoding outcome
ζ	Threshold factor

whether viewer $j \in \mathcal{J}^t = \{1, 2, \cdots, J^t\}$ chooses to watch broadcaster i's live stream in time slot t or not. In addition, denote $y \in \mathcal{Y} = \{1, 2, \cdots, Y\}$ as a video representation which is one of Y possible standard quality levels of a transcoded video and the bit rate of representation y is defined as b_y.

When an original live video is uploaded by a broadcaster i to the regional data centers, the scheduler of CLSP will decide which representation should be transcoded by which edge device based on viewer requirements, the performance of the edge devices, and system constraints such as the experienced delay by users as well as the cost for transcoding. In the transcoding process, each edge device $f \in \mathcal{F} = \{1, 2, \cdots, F\}$ is able to transcode the broadcasters' live videos into standard video versions where $\mathcal{F}$ represents the set of all available edge devices. After the transcoding process, all of the standard transcoded live videos will be transmitted from the edge devices to the associated viewers. In addition, it is assumed that a viewer can only watch one transcoded video from a broadcaster in the same time slot.

To describe the transcoding task assignment and the viewer association, a binary variables is defined as $I_{i,y,f,j}^t$ which takes 1 when edge device f is selected to transcode the original video from broadcaster i requested by viewer j into representation y in time slot t, and 0 otherwise.

A. Cost Model

To incentivize an edge device to participate in transcoding, we define the cost of transcoding one live video from broadcaster i to representation y at edge device f, which is paid by

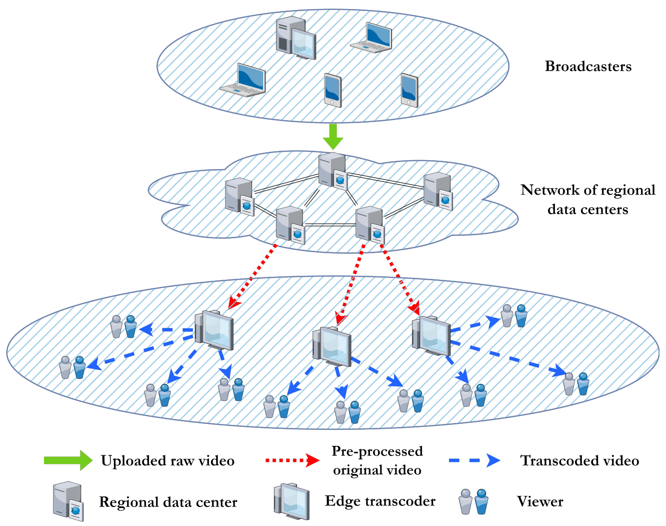


Figure 1: System model of the edge-assisted CLSP

the CLSP for the edge device, as $c_{i,y,f}^t$, which is written as

$$c_{i,y,f}^t = \sum_{j \in \mathcal{J}} I_{i,y,f,j}^t \cdot \Phi_y \cdot \omega_f^t, \tag{1}$$

where Φ_y is a non-decreasing concave function of representation y and ω_f^t is defined as the transcoding capability of device f in time slot t.

A higher value of ω_f^t means the edge device is more reliable and it can transcode a live stream with higher quality and less delay. To encourage edge devices with higher transcoding capability to join the transcoding candidate pool, $c_{i,y,f}^t$ is assumed to be linearly increasing with ω_f^t. The transcoding capability plays an important role which is not only related to the transcoding quality but also the transcoding performance uncertainty. In a nut shell, an edge device with relatively higher average performance and lower performance fluctuation is more capable for transcoding. The exact definition of transcoding capability will be presented in Section V.

Based on the definition of $c_{i,y,f}^t$, the total cost related to broadcaster i (denoted by c_i^t) can be defined as

$$c_i^t = \sum_{y \in \mathcal{Y}} \sum_{f \in \mathcal{F}} c_{i,y,f}^t. \tag{2}$$

B. QoE Model

From the perspective of a viewer, the quality of the received video, namely, the received bit rate can greatly determine the viewer's experience [14]. Therefore, we use the term QoE to denote how good the received video is. The QoE is determined by two factors. First, the acceptable quality levels of the received live videos of different broadcasters vary in terms of their genres (e.g., card game, pixel art game, first shooter game, etc). By categorizing the live videos into a set of genres denoted by $\mathcal{G} = \{1, 2, \ldots, K\}$ and defining $g_i^t \in \mathcal{G}$ as the genre of video from broadcaster i in time slot t, we can define $s_{g_i^t}$ as the suggested basic bit rate, according to the genre of broadcaster i, for viewer to watch the live video in time slot t. Second, it is vital to consider the network capacity of each viewer. Let u_j^t be the highest bit rate that viewer j can receive, which varies due to the viewer network condition, the QoE model can be expressed as

$$q_{i,j,y}^t = \log\left(\frac{b_y}{u_j^t} + \frac{b_y}{s_{g_i^t}}\right), \tag{3}$$

where b_y represents the bit rate of representation y. In (3), the QoE model is a non-decreasing concave function of two ratios. The first ratio quantifies the effect of the network condition of viewer j. The higher this ratio is, the better QoE can be achieved. However, this ratio should not exceed one, and a constraint is added to the optimization problem; Otherwise, the viewer capacity is smaller than the bit rate of the transcoded representation and this transcoded representation cannot be smoothly played at the viewer end. The second ratio quantifies how better the received video quality is compared with the basic genre rate of broadcaster i considering the fact that same representation from different genres of broadcasters can lead to different QoE levels.

The QoE of a viewer j watching a transcoded video from broadcaster i can be calculated as

$$Q_{i,j}^t = \sum_{y\in\mathcal{Y}}\sum_{f\in\mathcal{F}} I_{i,y,f,j}^t \omega_f^t q_{i,j,y}^t. \tag{4}$$

C. Delay Model

Delay is another performance measure that should be taken into account for the optimal transcoding task assignment and viewer association for real-time processing and delay-sensitive applications in the edge-assisted CLSPs. The latency experienced by the viewers can be categorized into three types, i.e., transmission delay, transcoding delay, and playout delay.

The transmission delay is referred to as the round trip time, including the broadcaster-transcoder delay and the transcoder-viewer delay. In the traditional cloud transcoding system, both the broadcasters and the viewers can be far from the cloud data center, which brings non-negligible latency. The transcoding delay is the processing delay of transcoding a live video to a different quality version. Normally, the transcoding delay can be calculated as the time difference between the input of original video and the output of the transcoded representation. The playout delay is determined by the viewer devices and their decoding time. Thus, it would not affect the the transcoding task assignment and the viewer association and hence not involved in this paper.

Let define ξ_j^t and δ_j^t as the transmission delay and the transcoding delay experienced by viewer j in time slot t, respectively. The transmission delay can be expressed as

$$\xi_j^t = \sum_{i\in\mathcal{I}}\sum_{y\in\mathcal{Y}}\sum_{f\in\mathcal{F}} \tilde{\tau}_{i,f,j}^t I_{i,y,f,j}^t V_{i,j}^t, \tag{5}$$

where $\tilde{\tau}_{i,f,j}^t$ denotes the network delay from broadcaster i to viewer j via edge device f. Next, the transcoding delay can be represented as

$$\delta_j^t = \sum_{i\in\mathcal{I}}\sum_{y\in\mathcal{Y}}\sum_{f\in\mathcal{F}} \tilde{\delta}_{i,y,f} I_{i,y,f,j}^t V_{i,j}^t, \tag{6}$$

where $\tilde{\delta}_{i,y,f}$ represents the transcoding delay for edge device f to transcode an original live video with bit rate B_i^t to a representation with bit rate b_y. Therefore, the overall latency in the proposed transcoding system in time slot t experienced by the viewer j can be represented as

$$D_j^t = \xi_j^t + \delta_j^t. \tag{7}$$

IV. Optimization Problem: Edge Transcoding and Viewer Association

According to the system model formulated in the previous section, there is a tradeoff between QoE maximization and cost minimization imposed on CLSP. On one hand, CLSP prefers to incentivize more edge devices to participate in transcoding and provide ABR service to more viewers. The more $I_{i,y,f,j}^t$ is set to one (i.e., a larger number of edge devices is selected for transcoding), the higher QoE can be gained. On the other hand, this will lead to higher cost based on (1). Therefore, the binary indicators must be optimized carefully to balance the tradeoff between the QoS and the cost as two components of the network utility. To formalize such a tradeoff, we define the weighted-difference between the QoE and cost (which is referred to as the network utility) related to a broadcaster as

$$U_i^t = \sum_{j\in\mathcal{J}^t} V_{i,j}^t Q_{i,j}^t - \lambda \cdot c_i^t, \tag{8}$$

where the parameter λ is used to tune the tradeoff between the two components.

Aiming to jointly optimize the transcoding task assignment and viewer association by maximizing the total network utility in each time slot over the whole transcoding system. We therefore, formulate an optimization problem as

$$(\boldsymbol{\mathcal{P}}) \max_{I_{i,y,f,j}^t} \sum_{t=1}^{T}\sum_{i\in\mathcal{I}} U_i^t, \tag{9a}$$

$$\textbf{s.t. C1}: V_{i,j}^t I_{i,y,f,j}^t b_y \le \min\{u_j^t, B_i^t\}, \tag{9b}$$

$$\textbf{C2}: \sum_{y\in\mathcal{Y}}\sum_{f\in\mathcal{F}} I_{i,y,f,j}^t \le 1, \forall i\in\mathcal{I}, \forall j\in\mathcal{J}^t, \tag{9c}$$

$$\textbf{C3}: \sum_{y\in\mathcal{Y}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}^t} I_{i,y,f,j}^t \le M, \forall f\in\mathcal{F}, \tag{9d}$$

$$\textbf{C4}: I_{i,y,f,j}^t \in \{0,1\}, \forall y\in\mathcal{Y}, \forall i\in\mathcal{I}, \forall f\in\mathcal{F}, \forall j\in\mathcal{J}^t, \tag{9e}$$

$$\textbf{C5}: D_j^t \le D_{\text{th}}, \forall j\in\mathcal{J}, \tag{9f}$$

where **C1** makes sure the received bit rate is lower than both the original video bit rate (B_i^t) and the viewer capacity. **C2** ensures that a viewer can only play one representation from one broadcaster in each time slot. **C3** guarantees that each transcoder can only serve M viewers at most due to the limited bandwidth resource. **C4** guarantees that variable $I_{i,y,f,j}^t$ is binary. **C5** ensures that for every viewer, the experienced delay of each viewer is lower than a predefined threshold D_{th}.

The formulated problem is a linear integer programming problem which can be efficiently solved by an optimization toolbox called Mosek [40]. However, the transcoding capabilities of transcoders are required to solve $\boldsymbol{\mathcal{P}}$, that are unknown in real live streaming systems. Therefore, an online learning scheme is highly demanded.

V. Risk-Aware Contextual Learning for Edge Transcoding

In light of the proposed system model in Section III, in this section, a novel risk-aware learning algorithm is designed to learn the transcoding capabilities (i.e., ω_f^t) online leveraging contextual information. Then, a novel transcoding task assignment and viewer association algorithm is designed to maximize the network utility of the transcoding system.

A. Risk-aware Contextual MAB

Since the aim of the edge-assisted transcoding system is to select a number of edge transcoders per time slot to maximize the cumulative network utility, this problem can be modelled using a bandit framework, where the arms are the edge devices and the rewards are the transcoding outcomes of selected edge devices.

We model the transcoding outcome of a task assigned to edge device f in time slot t as a random variable, denoted by r_f^t, for which the statistical properties are unknown. Suppose $\gamma_f^t = \mathbb{E}[r_f^t]$ represents the expected performance of transcoding for edge device f, where $\mathbb{E}[.]$ is the mathematical expectation. Define γ_f^t as 'transcoding quality' of device f. Moreover, $\sigma_f^2 = \text{Var}[r_f^t]$ is the variance of the transcoding outcome. When the variation of transcoding outcome of a transcoder (σ_f^2) is large, the transcoder can still perform poorly even with high transcoding quality. This is unaffordable and risky.

The risk is defined related to the performance fluctuations of the transcoder devices. In particular, the high risk represents the case when the transcoder has a large performance variation (i.e., transcoding outcomes with high variance). For instance, choosing a transcoder with high uncertainty can lead to unacceptable transcoding delay and severely deteriorate the viewer experience. Besides, choosing a more risky transcoder can lead to frequent transcoding task switches, which means the same transcoding task will be assigned to different edge transcoders and results in high communication overhead and playback latency. These problems reflect the importance of considering the performance uncertainty of transcoders.

Such a risk can occur due to the unexpected unavailability of transcoders. This happens in edge-assisted crowdsourced live streaming since transcoder devices considered in our work are assumed to be edge viewers' devices, which are not specifically employed for live video transcoding. The risk can also originate from the unstable computational and transmission resources of the edge transcoders. Particularly, a sudden high transmission error or a low transmission rate can aggravate the riskiness of edge transcoders as well.

Therefore, we model the transcoder selection problem as a risk-aware MAB for which the objective is to balance the tradeoff between maximizing the expected value of returned transcoding outcomes and minimizing the variance of the transcoding outcomes. In particular, we define the transcoding capability as the mean-variance measure of each transcoder, which can be written as

$$\omega_f^t = \rho\gamma_f^t - \sigma_f^2. \tag{10}$$

where $\rho > 0$ is the risk-tolerance factor introduced to balance the tradeoff between a high reward and a low risk. This linear combination of the transcoding quality and the variance of the transcoding outcome in fact defines the transcoding capability of edge device f.

To learn the transcoding capabilities of each edge transcoder online, in the following, we propose an index-based MAB algorithm by analytically driving the UCB of γ_f^t and σ_f^2.

1) Contextual UCB for transcoding quality: The transcoding quality (γ_f^t) of an edge device is dependant on various factors such as the device computational power, network conditions, online stability etc. Such factors will form the contextual information of a device as a transcoder. For example, since the viewer devices are not specifically implemented for video transcoding and can switch offline during transcoding [6], online stability of a device would affect the transcoding quality. Besides, the computational power and network condition of an edge device can also affect the transcoding outcome by incurring varying latency which can be experienced at the viewer end.

Therefore, we model the transcoding quality of a transcoder as the linear combination of its contextual information and a vector of unknown coefficients $\boldsymbol{\theta}^*$. Consequently, the transcoding quality can be represented as

$$\gamma_f^t = \mathbb{E}\left[r_f^t|\mathbf{x}_f^t\right] = (\mathbf{x}_f^t)^\top\boldsymbol{\theta}^*, \tag{11}$$

where $\mathbf{x}_f^t$ represents the z-dimensional contextual information of edge transcoder f, and $\boldsymbol{\theta}^*$ denotes the z-dimensional unknown coefficients which can be treated as the weight of each contextual information. In addition, the number of the contextual information types is defined as z.

Collecting samples of the transcoding outcomes and the contextual information through task assignments over time, the unknown coefficients $\boldsymbol{\theta}^*$ can be learned. Learning the coefficients belong to a linear regression problem and it can be solved by ridge regression [41], which adds L2 regularization to the lost function. Ridge regression is a suitable technique to solve the linear regression problem when the number of samples is highly limited, which fits the situation of the crowdtranscoding system, since the samples are collected in an online form and there are only limited samples in the early stage.

Define $\mathbf{R}^t$ as the set of transcoding outcomes till time slot t, with the number of transcoding outcomes as m^t. Let $\mathbf{W}^t$ be a design matrix of dimension $m^t \times z$ whose rows correspond to the observations of contextual information of m^t transcoding outcomes till time slot t and columns correspond to the z types of the contextual information. According to [42], we can acquire the estimated coefficients $\tilde{\boldsymbol{\theta}}^t$ by ridge regression as

$$\tilde{\boldsymbol{\theta}}^t = ((\mathbf{W}^t)^\top\mathbf{W}^t + \mathbf{I}_z)^{-1}(\mathbf{W}^t)^\top\mathbf{R}^t, \tag{12}$$

where $\mathbf{I}_z$ is the z-dimensional identity matrix.

Let define the estimated transcoding quality as $\tilde{\gamma}_f^t$. Based on the learned knowledge of the unknown coefficients, we can update the estimated transcoding quality using the contextual information as

$$\tilde{\gamma}_f^t = (\mathbf{x}_f^t)^\top \tilde{\boldsymbol{\theta}}^t. \tag{13}$$

For the UCB of the transcoding quality (γ_f^t), according to [43], for any $\kappa > 0$, with the probability of at least $1 - \kappa/T$, the deviation between the estimated transcoding quality and the real transcoding quality can be upper bounded by

$$|(\mathbf{x}_f^t)^\top \tilde{\boldsymbol{\theta}}^t - (\mathbf{x}_f^t)^\top \boldsymbol{\theta}^*| \le (\phi + 1)\sqrt{(\mathbf{x}_f^t)^\top {\mathbf{A}^t}^{-1} \mathbf{x}_f^t}, \tag{14}$$

where $\mathbf{A}^t = \mathbf{I}_z + (\mathbf{W}^t)^\top \mathbf{W}^t$ and $\phi = \sqrt{\frac{1}{2}\ln\frac{2TF}{\kappa}}$. This UCB can help to estimate the real transcoding quality of each transcoder, which holds with a high probability.

2) UCB for variance: In order to estimate the UCB of the transcoding outcome's variance (σ_f^2), we first define the empirical variance $(s_f^t)^2$ of the transcoding outcome of edge device f until time slot t as

$$(s_f^t)^2 = \frac{1}{\tau_f^t - 1}\sum_{d=1}^{\tau_f^t}\left(r_f^{t_f(d)} - \bar{r}_f^t\right)^2, \tag{15}$$

where $\bar{r}_f^t$ represents the empirical mean of transcoding outcome until t, $t_f(d)$ represents the time slot when the d^{th} transcoding outcome of device f is observed, and τ_f^t denotes the number of times that transcoder f has been chosen till t.

Fact 1: Let X be a Gaussian random variable with variance σ^2. Define the empirical variance over n samples as s_n^2, based on [44] we have

$$\Pr\left\{\sigma^2 \le v_n\right\} = 1 - \alpha, \tag{16}$$

where $v_n = \frac{(n-1)s_n^2}{\chi^2_{1-\alpha,n-1}}$ and $\chi^2_{1-\alpha,n-1}$ is the upper 100α percentage points of the chi-square distribution with $(n-1)$ degrees of freedom.

Fact 1 gives a definition of the confidence interval of the variance of a random variable when the variable follows the Gaussian distribution. It implies that there is a probability of $100(1-\alpha)\%$ that the constructed confidence interval based on the sample variance will contain the true value of σ^2.

According to Fact 1, a UCB of the variance of a random variable is proposed when the variable follows the Gaussian distribution. Therefore, based on Fact 1, we can derive the UCB of σ_f^2 under assumption that r_f^t is normally distributed.

Lemma 1: Given the UCBs of both the mean and the variance of the transcoding outcomes based on (14) and (16), the contextual Gaussian risk-aware UCB (CGRA-UCB) of transcoding capability of the transcoder f can be written as

$$\tilde{\omega}_f^t = \rho\left(\tilde{\gamma}_f^t + (\phi+1)\sqrt{(\mathbf{x}_f^t)^\top {\mathbf{A}^t}^{-1}\mathbf{x}_f^t}\right) - v_{\tau_f^t}, \tag{17}$$

where $v_{\tau_f^t} = \frac{(\tau_f^t - 1)(s_f^t)^2}{\chi^2_{1-\alpha,\tau_f^t-1}}$. The first term in the RHS of (17) represents the UCB of the transcoding quality and the second term reflects the variance of the transcoding outcome.

In (17), τ_f^t represents the number of transcoding tasks which is assigned to transcoder f till time slot t and $(s_f^t)^2$ denotes the empirical variance of the transcoding outcome of transcoder f at time slot t.

The bound in (17) is designed under the assumption that r_f^t follows an independent Gaussian distribution. However, when the reward distribution is unknown, the confidence interval presented in (1) is not pertinent. To overcome this limitation, we utilize the asymptotic distribution of the empirical variance to drive a confidence interval without any prior assumption of the reward distribution.

Fact 2: Let X be a continuous random variable with mean μ, variance σ^2, and $\mu_4 = \mathbb{E}\left[(X-\mu)^4\right]$. According to [38], [45], the asymptotic distribution of the empirical variance is

$$\sqrt{n}\left(s_n^2 - \sigma^2\right) \to \mathcal{N}\left(0, \mu_4 - \sigma^4\right). \tag{18}$$

Based on Fact 2, in the following lemma, we develop an asymptotic UCB on the variance.

Lemma 2: Applying Fact 2, define the UCB of the reward variance as v_n^{upper}, for a sufficiently large n, an asymptotic confidence interval of the variance can be derived as

$$\Pr\left\{\sigma^2 \ge v_n^{\text{upper}}\right\} \le \alpha, \tag{19}$$

where $v_n^{\text{upper}} = \frac{ns_n^2 + \sqrt{n\chi^2_{\alpha,1}(\mu_4 - s_n^4) + \left(\chi^2_{\alpha,1}\right)^2\mu_4}}{n + \chi^2_{\alpha,1}}$ and s_n^4 is the empirical estimate of σ^4.

Proof: See Appendix A.

Since μ_4 is unknown, an estimate of μ_4 is required. We approximate μ_4 as $\tilde{\mu}_4^n = \frac{1}{n}\sum_{d=1}^n (r_d - \bar{\mu})^4$, where r_d represents the random reward.

When the distribution of r_f^t is unknown, by setting $n = \tau_f^t$, we can estimate the UCB of σ_f^2 according to Lemma 2, which can be calculated as

$$v_{\tau_f^t}^{\text{upper}} = \frac{\tau_f^t s_{\tau_f^t}^2 + \sqrt{\tau_f^t \chi^2_{\alpha,1}\left(\tilde{\mu}_4^f - s_{\tau_f^t}^4\right) + \left(\chi^2_{\alpha,1}\right)^2 \tilde{\mu}_4^f}}{\tau_f^t + \chi^2_{\alpha,1}}, \tag{20}$$

where $\tilde{\mu}_4^f = \frac{1}{\tau_f^t}\sum_{d=1}^{\tau_f^t}(r_f^{t_f(d)} - \bar{r}_f^t)^4$.

Lemma 3: Given the UCBs of both the mean and the variance of the transcoding outcomes based on (14) and Lemma 2, we can build a new contextual asymptotic risk-aware UCB (CARA-UCB) of transcoding capability, which can be written as

$$\tilde{\omega}_f^t = \rho\left(\tilde{\gamma}_f^t + (\phi+1)\sqrt{(\mathbf{x}_f^t)^\top {\mathbf{A}^t}^{-1}\mathbf{x}_f^t}\right) - v_{\tau_f^t}^{\text{upper}}. \tag{21}$$

B. Transcoder Selection Algorithm

With the learnt transcoding capability and based on either (17) or (21), to assign the transcoding tasks to the edge devices which are expected to return relatively high reward and are less risky, we need to solve an instantaneous version of the optimization problem $\boldsymbol{\mathcal{P}}$ in (9). The instantaneous optimization problem at time-slot t can be formulated as

$$(\hat{\boldsymbol{\mathcal{P}}}) \ \max_{I_{i,y,f,j}^t} \sum_{i\in\mathcal{I}} U_i^t, \tag{22a}$$

$$\textbf{s.t.} \ \ \textbf{C1} - \textbf{C5}. \tag{22b}$$

The instantaneous optimization problem in (22) will be solved whenever new bounds of transcoding capabilities of edge devices are available. After the task assignment, the UCB estimations can be updated based on the observed transcoding outcomes.

However, in order to learn the transcoding capability, simply assigning one transcoding task to different edge transcoders in different time slots is not efficient, because assigning a transcoding task to different transcoders frequently can lead to unaffordable task-switching costs and further increase the communication overheads.

To deal with this problem, according to (12), we noticed that whichever transcoder is selected can contribute in collecting information about the coefficients vector $\boldsymbol{\theta}^*$, thus can further guide the learning process of the transcoding capability. Therefore, instead of determining the task assignment and viewer association in each time slot, we investigate an epoch-based sampling strategy which means a transcoding is consistently assigned to the same edge device for a finite number of time slots (which is referred as an epoch) and the task reassignments are only proceeded once at the beginning of each epoch. With this strategy, we can greatly reduce the switching costs while keep learning the transcoding capability.

The effectiveness of the epoch-based sampling depends on a well-designed epoch length [46], which is supposed to increase as time continues. Given τ_f^t as the task assignment counter of a edge device f, define F^t as the set of edge devices whose assigned task numbers are more than a certain threshold till time slot t, which can be written as

$$F^t = \left\{ f : \tau_f^t \geq t^{\zeta} \log t \right\}, \tag{23}$$

with $\zeta > 0$ is the threshold factor. Define the smallest counter as

$$\tau_{\text{min}}^t = \min_{f \in F^t} \tau_f^t, \tag{24}$$

the length of an epoch till time slot t can be calculated as

$$E^t = \lceil (1+\epsilon)^{\tau_{\text{min}}^t} \rceil, \tag{25}$$

where $\epsilon > 0$.

The detailed risk-aware contextual transcoding task assignment and viewer association algorithm is described in Algorithm 1. Since the time slots have been divided into epochs, we only need to solve the optimization problem per epoch. Thus, the computational cost can be greatly reduced.

In addition, according to the optimization problem $\hat{\boldsymbol{\mathcal{P}}}$, multiple transcoding tasks can be assigned to the same transcoder. However, the computational resources of the transcoders are limited and performing excessive transcoding tasks on one transcoder concurrently can lead to soaring transcoding delay and exhaust the bandwidth resources. Therefore, to avoid overwhelming the transcoders, every time the optimization variable $I_{i,y,f,j}^t$ is calculated, a self-inspection process is executed at every selected transcoder and any transcoder assigned with excessive tasks will offload these tasks to the cloud data center for transcoding.

C. Computational Complexity Analysis

The computational complexity of the proposed algorithm in each epoch consists of two parts. The first part of complexity originates from the ridge regression where matrix inversion and multiplication are introduced. The computational complexity of this method scales as $\mathcal{O}(z^2 m^t)$, where z is the dimension of context space and m^t is the number of transcoding outcomes till time slot t. Since the dimensionality of the context information is assumed to be fixed, m^t will dominate the computational complexity and the complexity only grows linearly in terms of the number of transcoding outcomes.

The second part of complexity comes from solving the instantaneous optimization problem $\hat{\boldsymbol{\mathcal{P}}}$ (22) which is a 0-1 linear integer programming problem. According to [47], the computational complexity of such a problem is $\mathcal{O}(2^L kL)$ where L is the number of optimization variables and k is the number of constraints. In our problem, we have $L^t = IFYJ^t$ and $k = J^t + F + 2IJ^t$ where J^t, Y, F, and I represent the numbers of viewers, representations, edge devices, and broadcasters, respectively. Combing both parts, the computational complexity at the time slot t is $\mathcal{O}(z^2 m^t + 2^{L^t} k L^t)$.

Algorithm 1 Risk-aware contextual transcoding task assignment and viewer association algorithm

```
Require: $\zeta$, $\rho$, $\mathbf{x}_f^t$, $\phi$
  for $t \leftarrow 1$ to $T$ do
    if Current epoch ends then
      Update $\tau_f^t$ and $\tau_{\text{min}}^t$
      Calculate the epoch length $E^t$ based on (23), (24), and (25)
      for $f \leftarrow 1$ to $F$ do
        Calculate the estimated transcoding capability $\tilde{\omega}_f^t$ based on (17) or (21)
      end for
      Solve optimization problem $\hat{\boldsymbol{\mathcal{P}}}$ to get $I_{i,y,f,j}^t$
      Execute self-inspection and modify $I_{i,y,f,j}^t$ accordingly
    else
      Maintain the same assignment, $I_{i,y,f,j}^t \leftarrow I_{i,y,f,j}^{t-1}$
    end if
    Observe the transcoding outcomes
    Update $\tilde{\boldsymbol{\theta}}^t$ based on (12)
  end for
```

VI. Simulation Results

A. Simulation Setup

We test the proposed algorithm with a synthetic data set, which is based on the real-world settings. We assume a live stream transcoding system with 4 broadcasters, 50 viewers, 15 edge devices and 4 representations. The viewer count of each broadcaster live stream is decided by its popularity. The popularity is modelled by Zipf distribution which is normally used for video content popularity modelling (e.g., [48]).

We set the original live video rate and the representation rates according to the twitch broadcaster settings [49]. The original rates for four broadcasters are set as 4000kbps, 2500kbps, 1500kbps and 500kbps. The specific bit rates of the four representations are set to be 400kbps (240P), 1200kbps (480P), 2000kbps (720P), and 3500kbps (1080P).

Table III: Comparison with benchmarks on risk sensitivity and context awareness

	Risk sensitivity	Context awareness
Proposed scheme	Yes	Yes
LinUCB	No	Yes
MV-UCB	Yes	No

Moreover, we randomly set the viewer capacity in the range of $[500, 4000]$ kbps.

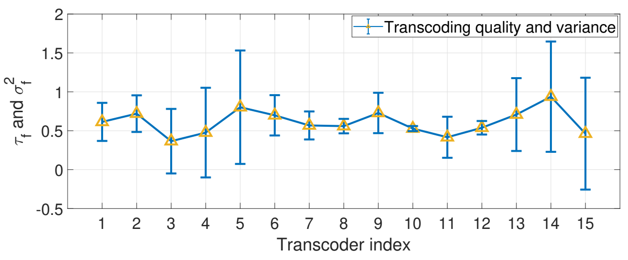
Based on the system model, a transcoding task can be assigned to multiple edge devices to serve different viewers. Since the transcoders are at network edge which is close to the viewers, the edge devices and the viewers are assumed to be distributed in a 1000 meters $\times$ 1000 meters region, and their locations are randomly determined following a uniform distribution.

According to [6], the transcoding capability can be affected by transcoder's computational power. Besides, since the candidate viewers can also undertake transcoding tasks and the viewers with low stability can be offline during transcoding, the online stability of the transcoders should be considered as a factor of transcoding capability. Based on [7], the online stability is generated by sampling the Pareto distribution. As a result, we choose the CPU mark, the average CPU usage, the average RAM usage and the online stability as the contextual information of edge transcoders.

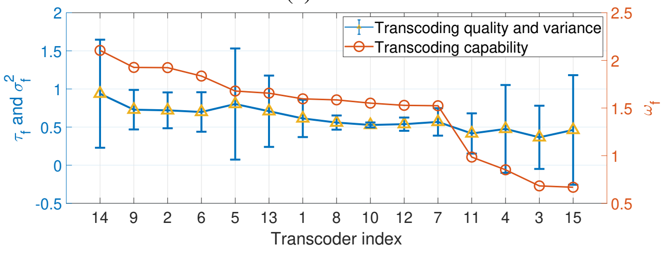
As discussed in [9], the transcoding delay is calculated based on the required computational resources of a task and the available CPU cycles (determined by the CPU mark and usage) of the transcoder. For the transmission delay, it can be divided into two parts as discussed in Section III-C. The broadcaster-transcoder delay is randomly set in the range of $[200, 300]$ ms according to [14], and the transcoder-viewer delay is set in the range of $[0, 100]$ ms depending on the distance between a viewer and a edge device. To be more specific, this delay can be calculated as the distance between transcoder and viewer times 100 ms. In other words, this setting implicitly considers the average channel gain and transmission rate which are functions of the distance between a viewer and an edge device. To demonstrate the risk-awareness of the designed algorithm, the variance of an edge transcoder follows a uniform distribution within the range of $[0, 1]$. Finally, to test the performance of the designed algorithm, both Gaussian and Gamma distributions are simulated to generate the transcoding outcomes. The parameters are uniformly selected.

The LinUCB algorithm [50] which utilizes the contextual information to estimate the reward is simulated as the benchmark. In addition, the MV-UCB algorithm [51] which is a risk-aware MAB algorithm, is also simulated for comparison. Table III describes the features of the proposed algorithm and the benchmarks.

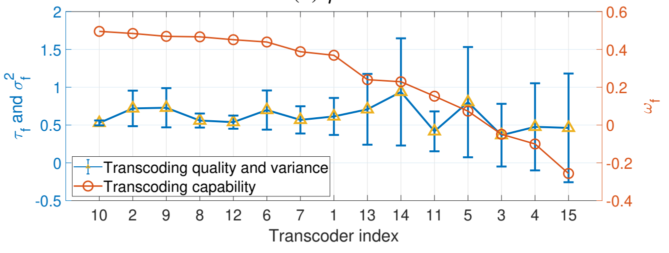
Figure 2a presents the initial transcoding quality and the variance of 15 transcoders. By sorting the transcoders in the descending order in terms of the transcoding capability with varying risk-tolerance factor ρ, Figures 2b-2d are generated. The risk-tolerance factor is set to decrease from 3 to 0.1. In Figure 2b, a larger risk-tolerance factor makes the transcoding



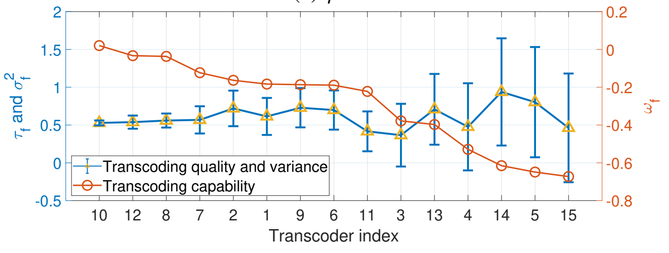
(a) Default



(b) $\rho = 3$



(c) $\rho = 1$



(d) $\rho = 0.1$

Figure 2: Transcoding quality and variation given varying risk-tolerance factor ρ

quality dominate the transcoding capability, which represents a risk-neutral setting of transcoders. In Figure 2d, the risk-tolerance factor is set to be close to 0, which leads to a pure-risk setting. We can observe that the transcoding capability of each transcoder can be quite different. In Figure 2c, we set $\rho = 1$, which leads to another order of transcoders. In this figure, both transcoding quality and the variance can contribute to the capability, and the transcoders which are relatively more capable are quite different from both previous settings ($\rho = 0.1$ or 3). This setting highlights the importance of considering impacts from both the transcoding quality and the variance. We set $\rho = 1$ in the following simulations, and the results are collected from 30 Monte Carlo (MC) runs.

B. Numerical Results

We first evaluate the proposed algorithm under the Gaussian distribution scenario as compared to the benchmarks. In Figure 3, the network utilities per time slot achieved by all three

Table IV: Simulation parameters

Parameter	Value
Original rates (kbps)	4000,2500,1500,500
Representation rates (kbps)	3500,2000,1200,400
Broadcaster genrge rate (kbps)	200,3000,1200,400
Viewer capacity (kbps)	$[500, 4000]$
broadcaster-transcoder delay (ms)	$[200, 300]$
transcoder-viewer delay (ms)	$[0, 100]$
risk-tolerance factor ρ	1
Threshold factor ζ	0.75
Context weight vector $\boldsymbol{\theta}^*$	(0.2,0.1,0.05,0.65)
QoE-cost coefficient λ	0.05

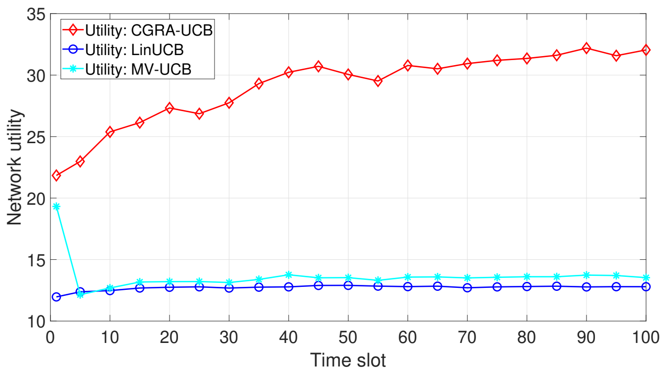


Figure 3: Network utility versus time: Gaussian reward case

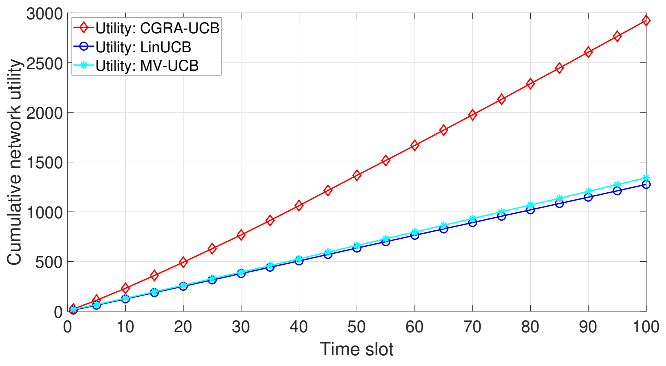


Figure 4: Cumulative utility versus time: Gaussian reward case

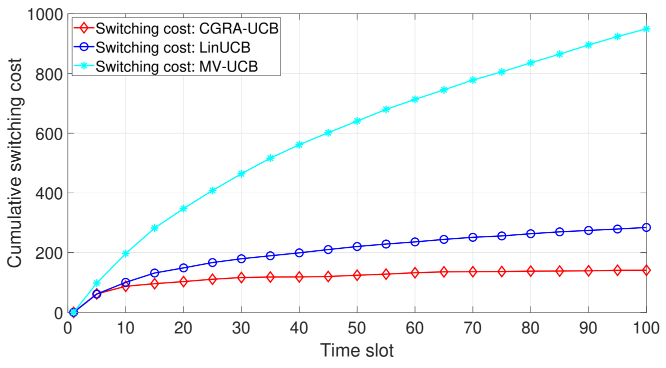


Figure 5: Switching costs versus time: Gaussian reward case

algorithms are presented. It is shown that the proposed algorithm using CGRA-UCB outperforms the LinUCB and MV-UCB because it not only utilizes the contextual information to

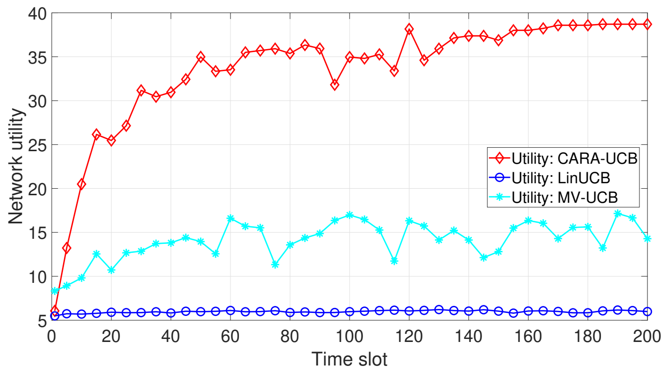


Figure 6: Network utility versus time: Gamma reward case

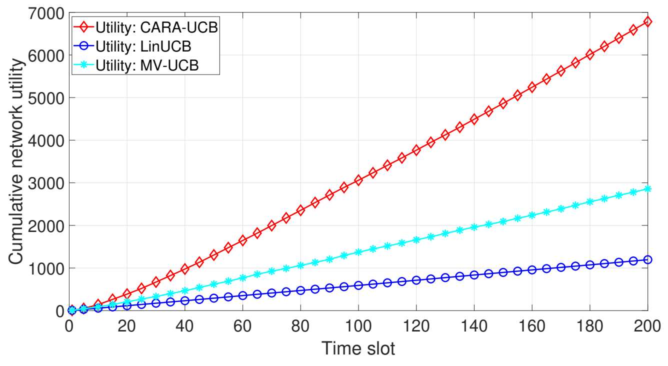


Figure 7: Cumulative utility versus time: Gamma reward case

learn the transcoding quality but also considers the uncertainty of the transcoder's transcoding outcome. In addition, the cumulative network utilities are depicted in Figure 4. This also confirms the superiority of the proposed algorithm in comparison with the benchmarks. Moreover, in Figure 5, the cumulative switching costs are presented and the proposed algorithm achieves up to 85.1% cumulative switching costs as compared to the benchmarks. This is because the proposed algorithm does not solve the optimization problem $\hat{\mathcal{P}}$ per time slot so that the task assignment and viewer association will not change frequently.

In Figures 6, 7, and 8, Gamma distribution is used to generate the transcoding outcomes. In this case, we simulate for 200 time slots since the used bound (derived in Lemma 3) is an asymptotic bound and its accuracy increases as more transcoding outcomes are collected, which takes longer time to converge. From the results, we can find that after 150 time slots, the proposed algorithm with the CARA-UCB tends to converge and shows good performance. The results demonstrate that the proposed algorithm using CARA-UCB achieves a higher network utility while reducing up to 86.8% switching costs as compared to the benchmarks. The experienced average latency per viewer of each algorithm is presented in Table V. According to the results, we can find that the average delays of the proposed algorithm are slightly higher in both scenarios, although still within the delay threshold. The delay thresholds in both simulations are set to 1.3 seconds, which demonstrates that the proposed algorithm can improve the network utility of the transcoding system

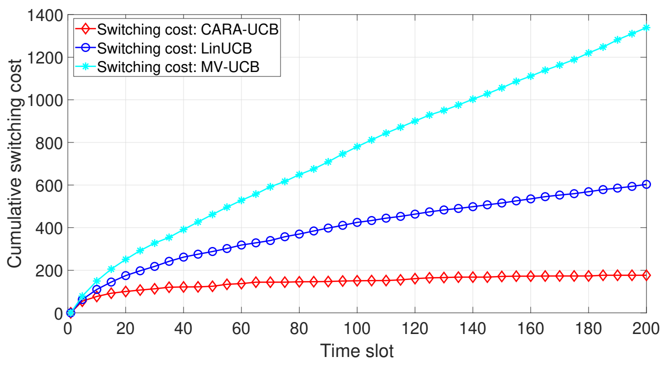
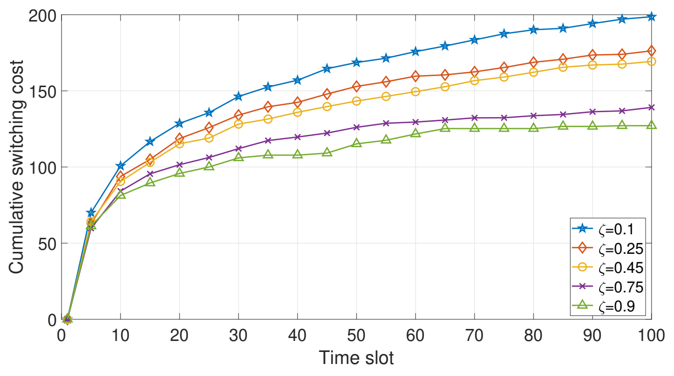


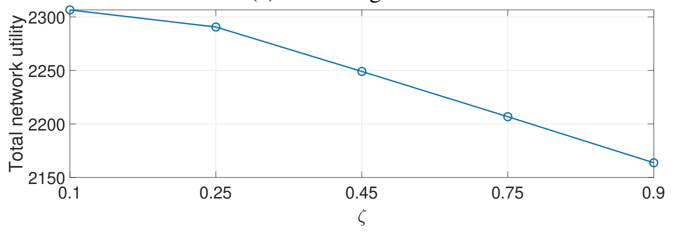
Figure 8: Switching costs versus time: Gamma reward case

Table V: Averaged delay in second

Algorithm	Gaussian case	Gamma case
CGRA/CARA-UCB	0.78	0.88
LinUCB	0.44	0.42
MV-UCB	0.46	0.58



(a) Switching costs



(b) Network utility

Figure 9: Transcoding performance versus ζ

while satisfying the delay constraint.

The running time of the proposed algorithm with and without the designed epoch-based strategy is presented in Table VI. Here without the epoch-based strategy means the optimization is solved in every time slot. The simulation is based on Gaussian reward setting with 100 time slots and the results are averaged over 30 MC runs. The results demonstrate that the epoch-based strategy can greatly reduce the running time by 92.3% since the optimization problem is solved much less frequently and the number of reassignment needed is much smaller.

In order to study the impact of the epoch-based sampling strategy, ζ is changed to generate different thresholds based on (23), which will help to determine the length of epoch according to (24) and (25). In Figure 9, both the cumulative switching costs and the total network utilities versus ζ are presented. We can observe that by decreasing ζ, the network utility can be increased at the expense of higher switching costs, since when ζ is decreased, the epoch length will increase more slowly and the optimization problem $\mathcal{P}$ in (9) will be solved more frequently. This figure demonstrates the importance of selecting a proper ζ to balance the tradeoff between the switching costs and the network utility.

In Figures 10, 11, and 12, three different epoch length determination strategies are evaluated in the Gaussian reward setting. The proposed transcoder selection algorithm calculates the epoch length based on the smallest task assignment counter of the edge transcoder via (23)-(25). As benchmarks, two more cases are simulated using the average of counters and the largest counter to calculate the epoch length, respectively. The results reveal that all three strategies perform well and show fast convergence thanks to the proposed refined UCBs of the transcoding capability. Particularly, the proposed strategy based on the minimum counter achieves the highest network utility. This confirms that the designed strategy can efficiently identify suitable transcoders to maximize the network utility, since it can offer more chances for exploring the transcoding capability of each transcoder and help to refine the bounds more frequently, which helps to identify the most suitable transcoders efficiently. In addition, using the maximum counter to calculate the epoch length achieves competitively low switching costs since this strategy tends to increase the epoch length faster than others, which can reduce the number of task reassignments.

To further demonstrate the performance of the proposed algorithm, we present the transcoding performances with and without the knowledge of the transcoding capabilities of edge transcoders under the Gaussian distribution scenario. In Figure 13, the cumulative network utilities and switching costs are presented. This result shows that the proposed scheme can learn the transcoding capability quickly and achieve a highly competitively network utility as compared to the case when the transcoding capability is known. In addition, with known transcoding capabilities, a lower switching costs can be achieved since the suitable transcoders can be quickly identified and the transcoding task assignment will not be

Table VI: Running time of the proposed algorithm

Proposed algorithm	Average running per run (seconds)	Average time running time per time slot (seconds)	Average number of reassignments
With epoch	520.8	5.2	7.7
Without epoch	6748.2	67.5	100

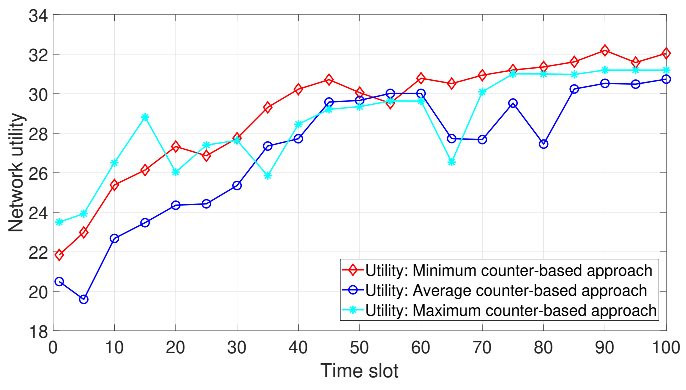


Figure 10: Network utility versus time

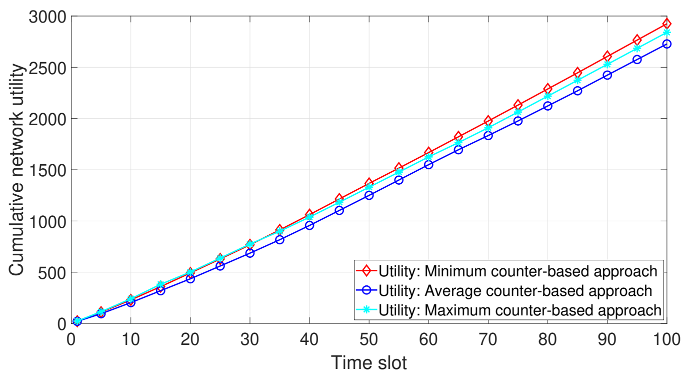


Figure 11: Cumulative network utility versus time

changed frequently.

Finally, we have compared the proposed edge-assisted transcoding algorithm with the Top-N scheme which is a currently-running cloud transcoding scheme in Twitch.TV. Top-N offers N premium broadcasters with the ABR service but only the basic representation rate is available for the rest of the broadcasters. Normally N is determined based on the broadcaster's popularity. The network utility and cumulative network utility of both the proposed algorithm and the Top-N scheme are presented in Figures 14 and 15.

Since Top-N is based on the cloud transcoding, we assume it can ensure highly stable viewer QoE and we set the transcoding capability to be 1 which is higher than the most capable edge-transcoder (whose transcoding capability is 0.496). In

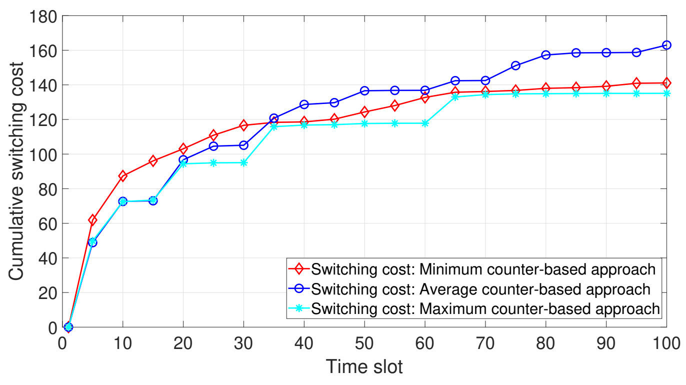
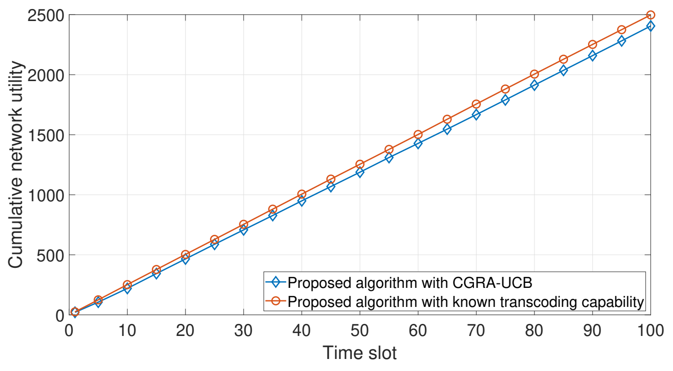
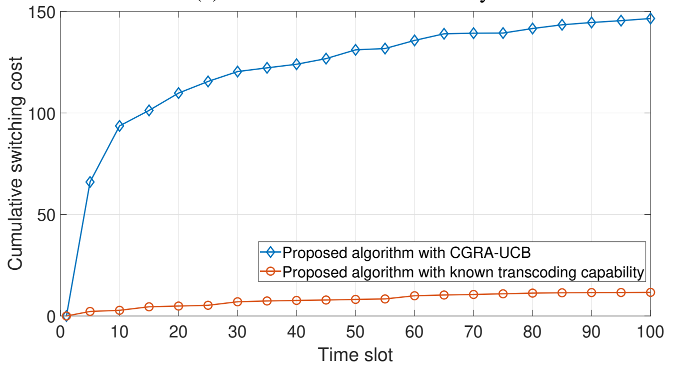


Figure 12: Switching costs versus time



(a) Cumulative network utility



(b) Switching costs

Figure 13: Transcoding performance versus ζ

addition, we set the unit cost of transcoding to be 10 times the cost of edge transcoding. Based on the results, we can find that as N increases, the utility of cloud transcoding increases. In particular, the proposed edge-assisted transcoding algorithm can utilize edge computing resources efficiently and achieve highly competitive network utility.

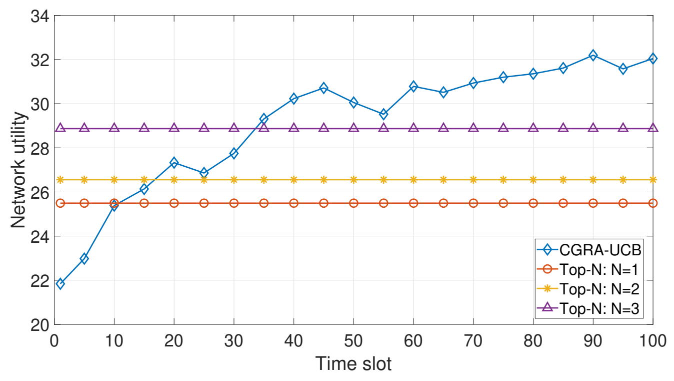


Figure 14: Network utility versus time

VII. Conclusions

In this paper, we proposed an edge-assisted transcoding task offloading algorithm for CLSP, considering the contextual information and the risk of performance variations of the edge devices. First, an optimization problem was formulated to solve the transcoding task assignment and viewer association problem under the assumption of known transcoding

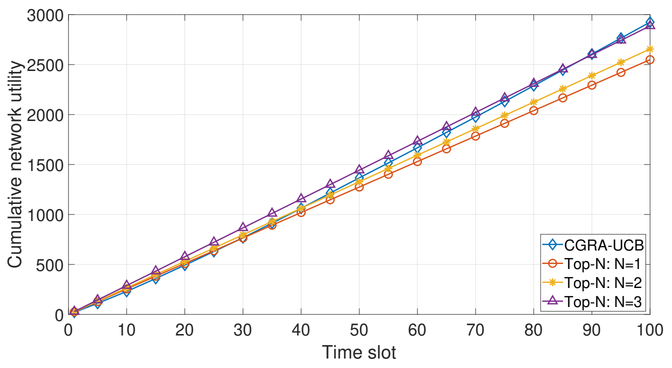


Figure 15: Cumulative network utility versus time

capabilities of edge devices. Then, two risk-sensitive bandit algorithms are developed to deal with the exploration-exploitation dilemma and to learn the transcoding capabilities. An epoch-based assignment strategy was introduced to reduce the switching costs of transcoding task assignment. Numerical results based on various settings confirm that the proposed risk-aware contextual algorithms can achieve superior performances as compared to different benchmark schemes that are either contextual or risk-sensitive. In a nutshell, leveraging both contextual awareness and risk sensitivity can improve resilience and robustness of an online task offloading scheme. In future, we will extend the proposed algorithm by considering a larger scale problem with extremely high live video quality (such as 4K and 8K) and with different behaviors of edge devices.

Appendix

The distribution presented in Fact 2 can be transformed into a standard Gaussian distribution as

$$\frac{\sqrt{n}\left(s_n^2-\sigma^2\right)}{\sqrt{\mu_4-\sigma^4}} \rightarrow \mathcal{N}\left(0,1\right). \tag{26}$$

Therefore we have $\frac{n\left(s_n^2-\sigma^2\right)^2}{\mu_4-\sigma^4} \rightarrow \chi_1^2$. Consequently, the one-sided confidence interval is defined as

$$\Pr\left\{\frac{n\left(s_n^2-\sigma^2\right)^2}{\mu_4-\sigma^4} \le \chi_{\alpha,1}^2\right\} = 1-\alpha. \tag{27}$$

As a result, the $(1-\alpha)$ asymptotic confidence interval of the variance σ^2 is established as

$$\Pr\left\{v_n^{\text{lower}} \le \sigma^2 \le v_n^{\text{upper}}\right\} = 1-\alpha, \tag{28}$$

where $v_n^{\text{lower}} = \frac{ns_n^2-\sqrt{n\chi_{\alpha,1}^2(\mu_4-s_n^4)+\left(\chi_{\alpha,1}^2\right)^2\mu_4}}{n+\chi_{\alpha,1}^2}$. From (28), obviously we have $\Pr\left\{\sigma^2 \ge v_n^{\text{upper}}\right\} \le \alpha$, which completes the proof.

References

[1] C. Zhang and J. Liu, "On crowdsourced interactive live streaming: A twitch.tv-based measurement study," in *Proceedings of the 25th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, Portland, 2015, p. 55–60.

[2] K. Bilal and A. Erbad, "Impact of multiple video representations in live streaming: A cost, bandwidth, and qoe analysis," in *Proceedings of the IEEE International Conference on Cloud Engineering*, 2017, pp. 88–94.

[3] N.-N. Dao, A.-T. Tran, N. H. Tu, T. T. Thanh, V. N. Q. Bao, and S. Cho, "A contemporary survey on live video streaming from a computation-driven perspective," *ACM Computing Surveys*, Feb 2022.

[4] J. Wise, "Twitch statistics 2022: How many people use twitch?" https://earthweb.com/twitch-statistics/.

[5] K. Bilal, O. Khalid, A. Erbad, and U. Samee, "Potentials, trends, and prospects in edge technologies: Fog, cloudlet, mobile edge, and micro data centers," *Computer Networks*, vol. 130, pp. 94 – 120, 2018.

[6] Q. He, C. Zhang, X. Ma, and J. Liu, "Fog-based transcoding for crowdsourced video livecast," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 28–33, 2017.

[7] Q. He, C. Zhang, and J. Liu, "Crowdtranscoding: Online video transcoding with massive viewers," *IEEE Transactions on Multimedia*, vol. 19, no. 6, pp. 1365–1375, 2017.

[8] C. Ge, N. Wang, W. K. Chai, and H. Hellwagner, "Qoe-assured 4k http live streaming via transient segment holding at mobile edge," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 8, pp. 1816–1830, 2018.

[9] Z. Zhang, R. Wang, F. R. Yu, F. Fu, and Q. Yan, "Qos aware transcoding for live streaming in edge-clouds aided hetnets: An enhanced actor-critic approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11 295–11 308, 2019.

[10] B. Yan, S. Shi, Y. Liu, W. Yuan, H. He, R. Jana, Y. Xu, and H. J. Chao, "Livejack: Integrating cdns and edge clouds for live content broadcasting," in *Proceedings of the 25th ACM International Conference on Multimedia*, 2017, p. 73–81.

[11] X. Liu, M. Derakhshani, and S. Lambotharan, "Joint transcoding task assignment and association control for fog-assisted crowdsourced live streaming," *IEEE Communications Letters*, vol. 23, no. 11, pp. 2036–2040, 2019.

[12] A. Erfanian, F. Tashtarian, A. Zabrovskiy, C. Timmerer, and H. Hellwagner, "Oscar: On optimizing resource utilization in live video streaming," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 552–569, 2021.

[13] Q. He, J. Liu, C. Wang, and B. Li, "Coping with heterogeneous video contributors and viewers in crowdsourced live streaming: A cloud-based approach," *IEEE Transactions on Multimedia*, vol. 18, no. 5, pp. 916–928, 2016.

[14] Y. Zheng, D. Wu, Y. Ke, C. Yang, M. Chen, and G. Zhang, "Online cloud transcoding and distribution for crowdsourced live game video streaming," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 8, pp. 1777–1789, 2017.

[15] K. Bilal, A. Erbad, and M. Hefeeda, "Crowdsourced multi-view live video streaming using cloud computing," *IEEE Access*, vol. 5, pp. 12 635–12 647, 2017.

[16] C. Zhang, J. Liu, and H. Wang, "Cloud-assisted crowdsourced livecast," *ACM Transactions on Multimedia Comput. Commun. Appl.*, vol. 13, no. 3s, pp. 46:1–46:22, 2017.

[17] G. Gao, Y. Wen, and C. Westphal, "Dynamic priority-based resource provisioning for video transcoding with heterogeneous qos," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 5, pp. 1515–1529, 2019.

[18] C. Dong, Y. Jia, H. Peng, X. Yang, and W. Wen, "A novel distribution service policy for crowdsourced live streaming in cloud platform," *IEEE Transactions on Network and Service Management*, vol. 15, no. 2, pp. 679–692, 2018.

[19] D. Lee and M. Song, "Quality-aware transcoding task allocation under limited power in live-streaming systems," *IEEE Systems Journal*, pp. 1–12, 2021.

[20] E. Baccour, F. Haouari, A. Erbad, A. Mohamed, K. Bilal, M. Guizani, and M. Hamdi, "An intelligent resource reservation for crowdsourced live video streaming applications in geo-distributed cloud environment," *IEEE Systems Journal*, vol. 16, no. 1, pp. 240–251, 2022.

[21] Y. Zhu, Q. He, J. Liu, B. Li, and Y. Hu, "When crowd meets big video data: Cloud-edge collaborative transcoding for personal livecast," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 1, pp. 42–53, 2020.

[22] W. Khan, E. Ahmed, S. Hakak, I. Yaqoob, and A. Ahmed, "Edge computing: A survey," *Future Generation Computer Systems*, vol. 97, pp. 219–235, 2019.

[23] F. Wang, C. Zhang, F. wang, J. Liu, Y. Zhu, H. Pang, and L. Sun, "Intelligent edge-assisted crowdcast with deep reinforcement learning for personalized qoe," in *Proceedings of the IEEE Conference on Computer Communications*, Paris, 2019, pp. 910–918.

[24] K. Bilal, E. Baccour, A. Erbad, A. Mohamed, and M. Guizani, "Collaborative joint caching and transcoding in mobile edge networks," *Journal of Network and Computer Applications*, vol. 136, pp. 86–99, 2019.

[25] F. Wang, J. Liu, C. Zhang, L. Sun, and K. Hwang, “Intelligent edge learning for personalized crowdsourced livecast: Challenges, opportunities, and solutions,” *IEEE Network*, vol. 35, no. 1, pp. 170–176, 2021.

[26] X. Chen, C. Xu, M. Wang, Z. Wu, S. Yang, L. Zhong, and G.-M. Muntean, “A universal transcoding and transmission method for livecast with networked multi-agent reinforcement learning,” in *Proceedings of the IEEE Conference on Computer Communications*, 2021, pp. 1–10.

[27] H. Huang, Q. Ye, and Y. Zhou, “Deadline-aware task offloading with partially-observable deep reinforcement learning for multi-access edge computing,” *IEEE Transactions on Network Science and Engineering*, 2021.

[28] M. Li, J. Wu, W. Wang, and J. Zhang, “Toward privacy-preserving task assignment for fully distributed spatial crowdsourcing,” *IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 13 991–14 002, 2021.

[29] A. I. Chittilappilly, L. Chen, and S. Amer-Yahia, “A survey of general-purpose crowdsourcing techniques,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 9, pp. 2246–2266, 2016.

[30] J. Ren, Y. Zhang, K. Zhang, and X. Shen, “Exploiting mobile crowdsourcing for pervasive cloud services: challenges and solutions,” *IEEE Communications Magazine*, vol. 53, no. 3, pp. 98–105, 2015.

[31] X. Tao and W. Song, “Location-dependent task allocation for mobile crowdsensing with clustering effect,” *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 1029–1045, 2019.

[32] K. Han, C. Zhang, and J. Luo, “Taming the uncertainty: Budget limited robust crowdsensing through online learning,” *IEEE/ACM Transactions on Networking*, vol. 24, no. 3, pp. 1462–1475, 2016.

[33] L. Tran-Thanh, S. Stein, A. Rogers, and N. R. Jennings, “Efficient crowdsourcing of unknown experts using bounded multi-armed bandits,” *Artificial Intelligence*, vol. 214, pp. 89–111, 2014.

[34] P. Yang, N. Zhang, S. Zhang, K. Yang, L. Yu, and X. Shen, “Identifying the most valuable workers in fog-assisted spatial crowdsourcing,” *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1193–1203, 2017.

[35] S. Müller, O. Atan, M. van der Schaar, and A. Klein, “Context-aware proactive content caching with service differentiation in wireless networks,” *IEEE Transactions on Wireless Communications*, vol. 16, no. 2, pp. 1024–1036, 2017.

[36] Y. Ma, C. Xu, X. Chen, H. Xiao, L. Zhong, and G.-M. Muntean, “Fairness-guaranteed transcoding task assignment for viewer-assisted crowdsourced livecast services,” in *Proceedings of the IEEE International Conference on Communications*, 2021, pp. 1–6.

[37] S. Wang, S. Bi, and Y.-J. A. Zhang, “Deep reinforcement learning with communication transformer for adaptive live streaming in wireless edge networks,” *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 1, pp. 308–322, 2022.

[38] X. Liu, M. Derakhshani, S. Lambotharan, and M. van der Schaar, “Risk-aware multi-armed bandits with refined upper confidence bounds,” *IEEE Signal Processing Letters*, vol. 28, pp. 269–273, 2021.

[39] Y. Zhang and H.-Y. Wei, “Risk-aware cloud-edge computing framework for delay-sensitive industrial iots,” *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 2659–2671, 2021.

[40] M. ApS, *The MOSEK optimization toolbox for MATLAB manual. Version 9.0.*, 2019. [Online]. Available: http://docs.mosek.com/9.0/toolbox/index.html

[41] G. C. McDonald, “Ridge regression,” *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 1, no. 1, pp. 93–100, 2009.

[42] A. E. Hoerl, R. W. Kannard, and K. F. Baldwin, “Ridge regression:some simulations,” *Communications in Statistics*, vol. 4, no. 2, pp. 105–123, 1975.

[43] W. Chu, L. Li, L. Reyzin, and S. Robert, “Contextual bandits with linear payoff functions,” in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, vol. 15, 2011, pp. 208–214.

[44] D. C. Montgomery and G. C. Runger, *Applied statistics and probability for engineers*. John Wiley and Sons, 2014.

[45] A. DasGupta, *Asymptotic theory of statistics and probability*. Springer Science & Business Media, 2008.

[46] T. Le, C. Szepesvári, and R. Zheng, “Sequential learning for multi-channel wireless network monitoring with channel switching costs,” *IEEE Transactions on Signal Processing*, vol. 62, no. 22, pp. 5919–5929, 2014.

[47] K. Genova and V. Guliashki, “Linear integer programming methods and approaches–a survey,” *Journal of Cybernetics and Information Technologies*, vol. 11, no. 1, 2011.

[48] Z. Chen, J. Lee, T. Q. S. Quek, and M. Kountouris, “Cooperative caching and transmission design in cluster-centric small cell networks,” *IEEE Transactions on Wireless Communications*, vol. 16, no. 5, pp. 3401–3415, 2017.

[49] “Broadcasting guidelines,” https://stream.twitch.tv/encoding/.

[50] L. Li, W. Chu, J. Langford, and R. Schapire, “A contextual-bandit approach to personalized news article recommendation,” in *Proceedings of the 19th International Conference on World Wide Web*, Raleigh, 2010, pp. 661–670.

[51] S. Vakili and Q. Zhao, “Risk-averse multi-armed bandit problems under mean-variance measure,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 6, pp. 1093–1111, 2016.