Adaptive Digital Twin-Assisted 3C Management for QoE-Driven MSVS: A GAI-Based DRL Approach

Xinyu Huang, *Graduate Student Member, IEEE*, Xue Qin, *Graduate Student Member, IEEE*, Mushu Li, *Member, IEEE*, Cheng Huang, *Member, IEEE*, and Xuemin Shen, *Fellow, IEEE*

***Abstract*—Communication, computing, and buffer control (3C) management is essential to enhance quality-of-experience (QoE) in multicast short video streaming (MSVS). The existing 3C management schemes mainly rely on static data processing methods and a general QoE model, which may not efficiently improve QoE when users' swipe behaviors exhibit distinct spatiotemporal differences. In this paper, we propose an adaptive digital twin (DT)-assisted 3C management scheme to enhance QoE in MSVS. Particularly, DTs consist of user status data and data-based models, which can update multicast groups and abstract users' swipe features. An adaptive DT management mechanism is developed to adapt to users' swipe behavior dynamics. Then, a fine-grained QoE model is established by considering the impact of resource constraints and DT model accuracy, leading to accurate buffer control. Finally, a joint optimization problem of 3C management is formulated to maximize long-term QoE. To efficiently solve this problem, a diffusion-based deep reinforcement learning (DRL) algorithm is proposed, which utilizes the denoising technique to improve the action exploration capabilities of DRL. Simulation results based on a real-world dataset demonstrate that the proposed DT-assisted 3C management scheme outperforms benchmark schemes in terms of QoE, with improvements of 18.4% and 20.5% under low and high user dynamics, respectively.**

***Index Terms*—Digital twin, 3C management, multicast transmission, short video, QoE, diffusion-based DRL.**

I. Introduction

IN THE digital media landscape, short video streaming has become a dominant form of content consumption, significantly altering global information sharing and consumption. Short video platforms like TikTok, Instagram Reels, and Snapchat have revolutionized the media sector by providing platforms that are tailored for quick, engaging, and often ephemeral video content. Measuring users' satisfaction with provided short video streaming services necessitates an efficient performance metric. Quality of experience (QoE), as an essential subjective performance metric, assesses the overall satisfaction level of a service perceived by end-users, encompassing factors such as rebuffer time, video quality, and quality variation [1]. Different users usually have different sensitive degrees of QoE factors, and these sensitivities change over time. The well-constructed QoE model can guide efficient communication, computing, and buffer control (3C) management.

Efficient 3C management is crucial for short video streaming services to enhance user QoE. The tremendous surge in video service requests (global monthly active users of platforms like TikTok and Instagram have reached 2.4 billion in 2024[1]) has posed significant challenges to the limited communication and computing resources in networks, leading to frequent playback interruptions and fluctuations in video quality, especially in densely populated areas [2], [3]. To cope with these challenges, multicast transmission, as a crucial technology in wireless networks, is widely used to allow a single data stream to be simultaneously distributed to a group of users. By employing multicast transmission to deliver short videos to users who share similar preferences and locations, redundancy in video processing and transmission can be effectively reduced, thereby alleviating overload on network traffic and computation [4]. However, optimizing multicast short video streaming (MSVS) with proper bandwidth and power allocation, along with adaptive user clustering, involves complex 3C management problems.

Generative AI (GAI) can help address this by learning and making near-optimal strategies in real time. For instance, it can model network dynamics and user behaviors to assist efficient resource allocation schemes and user clustering, enhancing transmission throughput [5]. In computing resource management, GAI can intelligently distribute video processing tasks to cloud and edge servers, improving processing speed [6]. For buffer control, it can adjust pre-loaded video data based on network conditions and playback status to minimize interruptions [7]. By using generative AI (GAI) to solve complex 3C management problems, user QoE can be effectively enhanced.

To further enhance user QoE, efficient 3C management requires the real-time perception of user dynamics and the abstraction of user behavior patterns. Therefore, digital twin (DT) is incorporated into mobile communication networks.

Received 11 July 2024; revised 27 October 2024; accepted 29 November 2024. Date of publication 12 December 2024; date of current version 9 April 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62402115, and in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada. The associate editor coordinating the review of this article and approving it for publication was V. Christos. *(Corresponding author: Cheng Huang.)*

Xinyu Huang, Xue Qin, and Xuemin Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: x357huan@uwaterloo.ca; x7qin@uwaterloo.ca; sshen@uwaterloo.ca).

Mushu Li is with the Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA 18015 USA (e-mail: mul224@lehigh.edu).

Cheng Huang is with the School of Computer Science, Fudan University, Shanghai 200433, China (e-mail: chuang@fudan.edu.cn).

Digital Object Identifier 10.1109/TCCN.2024.3516046

[1]https://www.searchlogistics.com/learn/statistics/tiktok-user-statistics/

DT consists of multiple user DTs (UDTs) and each UDT corresponds to a physical user. UDT can record the user status through data collection and prediction, and analyze the user's statistical information, such as preference and mobility pattern [8]. DT can help improve 3C management performance in MSVS from three aspects, i.e., personalized QoE (PQoE) model construction, users' future state prediction, and users' behavior pattern abstraction [9]. First, PQoE models can reflect users' sensitive degrees on different QoE factors that can guide tailored 3C management [10]. Second, users' future state prediction of playback status and swipe behaviors can help proactive video segment buffering, thus mitigating potential playback lags [11]. Third, abstracting users' behavior patterns can help update recommended video lists and reserve accurate network resources, thus alleviating network resource wastage [12].

Designing an efficient DT-assisted 3C management scheme for MSVS will face the following challenges.

- Firstly, due to the stochastic nature and spatiotemporal differences in users' swipe behaviors, designing an efficient DT management mechanism to characterize and process the highly dynamic and multi-dimensional user behavior data is challenging.
- Secondly, using a general QoE model that lacks accurate buffer control and rebuffer time estimation can cause frequent playback lags. It is challenging to construct a fine-grained QoE model by considering the effect of resource constraints on buffer update and DT model accuracy on multicast transmission delay estimation.
- Thirdly, since the 3C management problem is usually complex mixed-integer nonlinear programming, traditional deep reinforcement learning (DRL) algorithms usually struggle with highly dynamic environments due to spatiotemporal variation in user behaviors, leading to suboptimal performance [13], [14]. Therefore, an improved DRL algorithm should be developed to solve the complex 3C management problem.

In this paper, we propose a novel adaptive DT-assisted 3C management scheme for MSVS, which can effectively enhance QoE. The main contributions are summarized as follows:

- Firstly, we propose an adaptive DT management mechanism to adapt to user behavior dynamics. Specifically, a precise measurement model of DT data processing (DTP) is constructed to characterize the relationship among DT model size, user dynamics, and user clustering accuracy. The proposed DT management mechanism can adaptively trigger based on the error and entropy of predicted user status and user clustering result, respectively. An improved incremental learning method is further proposed to improve DT model update (DTU) efficiency by neural network growth and pruning.
- Secondly, we construct a fine-grained QoE model to incorporate the impact of resource constraints and DT model accuracy. Specifically, a two-layer segment buffering principle based on resource constraints is designed to realize accurate buffer control. The multicast rebuffer time estimation incorporates the impact of DT model accuracy on multicast transmission. Multicast rebuffer time, video quality, and video quality variation are determined by 3C management, together constituting the QoE model.
- Thirdly, we propose a diffusion-based DRL algorithm to solve the joint 3C optimization problem of bandwidth and computing resource allocation as well as video segment version selection. Diffusion model adopts the denoising technique to improve the action exploration capabilities of DRL algorithm. The extensive simulation results on real-world short video streaming datasets show that the proposed diffusion-based DRL algorithm can effectively improve QoE compared with benchmark schemes.

The remainder of this paper is organized as follows. Related works and system model are introduced in Sections II and III, respectively. Then, the adaptive DT and fine-grained QoE model are presented in Sections IV and V, respectively. Next, the problem formulation and proposed solution are shown in Section VI, followed by simulation results in Section VII. Finally, Sections VIII and IX make a discussion and conclusion.

II. Related Work

In this section, we introduce the related work from 3C management for multicast video streaming, GAI-assisted 3C management, and DT-assisted 3C management, respectively.

A. 3C Management for Multicast Video Streaming

Effective 3C management is crucial for optimizing the performance of multicast video streaming. *In communication resource management*, various strategies, including multicast subgrouping, scalable video coding (SVC), and bandwidth allocation, have been proposed to enhance throughput, energy efficiency, and spectral efficiency [15], [16], [17]. These approaches can address the challenges of differentiated service requirements and adaptive video buffering, which can facilitate multicast video streaming achieve higher data rates, improved energy efficiency, and enhanced user experience in various network environments. *In computing resource management,* computing tasks are usually processed with different priorities. An innovative classification approach of time-tolerant and time-sensitive video processing was proposed in [18] to improve computing efficiency for different users. To guarantee delivered video quality, a joint optimization scheme of power allocation and subgrouping was proposed to maximize sum multicast rates [19]. Furthermore, a caching and communication-aware video transcoding scheme was designed in [20] to efficiently coordinate caching, communication, and computing resources to ensure robust video quality. *In buffer control management,* many advanced buffer-aware control methods were proposed to reduce playback lags. A lower-than-affordable modulation coding scheme (MCS) order was preferred to prevent buffer underflow [21]. Additionally, a multi-user buffer-aware video streaming problem was formulated in [22], [23] as a stochastic game to maximize users' video viewing utility, which can maintain consistent video playback. From the existing works, 3C management for multicast video streaming is a complex optimization problem

that requires real-time and accurate analysis of user status to enhance QoE.

B. Generative AI-Assisted 3C Management

Recent advancements in GAI have significantly impacted 3C management in mobile communication networks. In communication resource management, GAI models are used to optimize spectrum allocation, predict network congestion, and enhance signal processing by generating synthetic data for training and testing purposes [5], [24]. These models enable more efficient utilization of communication channels and improve overall network performance. In computing resource management, GAI assists in the dynamic allocation of computational resources by forecasting workload demands and optimizing task scheduling [25]. This leads to improved load balancing and resource utilization in data centers and cloud computing environments. For buffer control management, GAI techniques are applied to predict traffic patterns and adjust buffer sizes in real-time, reducing latency and packet loss in network systems [7]. By leveraging GAI, network resources can be efficiently utilized to improve network performance.

C. Digital Twin-Assisted 3C Management

As an essential virtualization technology, DT was first introduced to monitor and mitigate anomalous events for flying vehicles [26]. By introducing DT into mobile communication networks, real-time and accurate user-level information can be aware and analyzed for efficient 3C management. A DT-assisted adaptive preloading scheme was proposed in [27] to analyze the relationship between future throughput and user behavior, which can enhance bandwidth efficiency. Similarly, accurate and user-specific PQoE models were learned in UDTs to reflect users' service requirements, which can help customized 3C management [10]. DTs were integrated into edge networks to capture QoE metrics for guiding content delivery strategies in both machine-centric and user-centric communications [28]. Multicast video streaming can benefit from DTs by modeling video buffer dynamics and adaptively adjusting video quality levels, which can realize intelligent 3C management [29]. In spite of the research efforts, designing an efficient DT-assisted 3C management scheme remains an open issue when user behavior dynamics and fine-grained QoE models are taken into account.

III. System Model

As shown in Fig. 1, we consider an adaptive DT-assisted MSVS scenario, which mainly consists of multiple base stations (BSs), users with multicast groups (MGs), one edge server equipped with a transcoder for video transcoding, multiple UDTs, and one network controller. The network operates in a time-slotted manner. In each real-time scheduling slot, the network controller needs to determine 3C management. Communication and computing resources are periodically reserved for MGs, with the interval between two resource reservation instances known as a resource reservation window. This window encompasses several scheduling slots. The indexes of scheduling slots and resource reservation windows

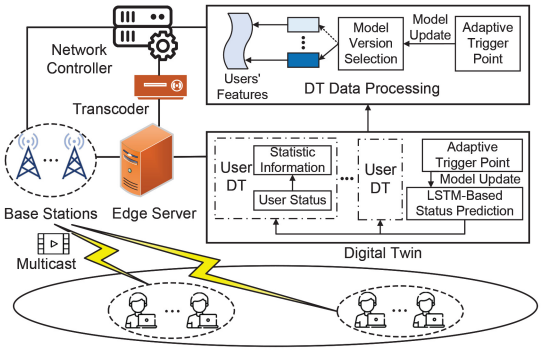


Fig. 1. Adaptive DT-assisted MSVS scenario.

are denoted by $t \in \{1, \ldots, T\}$ and $N \in \{1, \ldots, \mathbb{N}\}$, respectively.

- Multicast group: Users within one BS are clustered into multiple MGs based on the similarity of user status, including locations, channel conditions, swipe timestamps, and video preferences. The users within one MG can receive the same video data through multicast transmission. Video transmission to one MG is on the same frequency band. The user set of MG g is denoted by $\mathcal{U}_g$, and the number of MGs is G. In each resource reservation window, MG g will be recommended a video list Δ_g.
- Base station: BSs are responsible for transmitting short video sequences to users through multicast transmission, and collecting user status to update UDTs' data.
- Edge server: The edge server owns limited caching capability, which usually stores popular short video sequences in fast response to users' requests. The edge server is equipped with a transcoder, which connects to multiple BSs to transcode short video sequences into different versions, i.e., in different bitrates.
- Digital twin: A DT consists of multiple UDTs, deployed at the edge server, which stores users' status data. The data origins from two aspects, i.e., collected from physical networks and predicted through a longs short-term memory (LSTM) model. Based on the stored data, a DTP module in the network controller adopts a user clustering model to analyze user similarity for accurate MG update. The user clustering model is an autoencoder-based double deep Q-network (DDQN) model proposed in our previous work [29]. Running such a user clustering model may consume plenty of computing resources even if the user status is relatively static. Therefore, we deploy multiple user clustering models of various sizes and adaptively select an appropriate one based on user dynamics.[2] Furthermore, since the accuracy of LSTM model and user clustering model can gradually decrease with the evolving user dynamics, we utilize an improved incremental learning method to assist model update.
- Video transcoding and delivery: Video transcoding uses the SVC technology to transcode video sequences into

[2]To reduce the computational load of DTs, AI-based modules can be simplified and replaced by model-based modules.

TABLE I
NOTATIONS AND DEFINITIONS

Notation	Definition
ψ_N	The user dynamics in resource reservation window N
m_i	The model size of DT model version i
$f(m_i, \psi_N)$	The clustering accuracy with model size m_i and user dynamics ψ_N
Ψ_t^k	The accuracy of DT model k at time t
Θ_t^k	The average error of DT model k at time t
τ_k	The per batch training time for DT model k update
q_k	The gap between the current accuracy of model k and specified accuracy
φ	The time duration of overall DTU
$Q_{t,g}$	Users' average buffer length in MG g at time t
$\mathcal{C}_t^{\text{DTP}}$, $\mathcal{C}_t^{\text{DTU}}$, $\mathcal{C}_{t,g}$	The allocated computing resources for DTP, DTU and MG g at time t
$B_{t,g}$	The allocated bandwidths for MG g at time t
e_i	The selected DT model version i
$v_{t,g}^l$	The selected segment version l for MG g at time t

multiple versions to adapt to users' buffer dynamics. Then, transcoded video sequences are delivered to each MG through multicast transmission.

- Network controller: It is deployed at the edge server. On one hand, it decides the update trigger point and training batches for DT model[3] update. On the other hand, it is responsible for real-time 3C management for MSVS.

The adaptive DT-assisted MSVS operates as follows. In the network operation stage, an appropriate user clustering model is first activated to implement MG update. Then, video transcoding and multicast transmission are started based on real-time 3C management from the network controller. A cumulative error detector is used to monitor the accuracy of predicted user status in UDTs and clustering result in the DTP module. If the cumulative error exceeds a prescribed value, the network enters the DTU stage. In this stage, DTU is implemented in parallel with video transcoding and multicast transmission. The network controller determines real-time 3C management for video transcoding, multicast transmission and DTU. When the DTU is completed, the network re-enters the network operation stage.

The main notations and their definitions are listed in Table I.

IV. ADAPTIVE DIGITAL TWIN MANAGEMENT

In this section, we first introduce how to construct DT and then propose an adaptive DT management mechanism.

A. DT Construction

1) DT Data Attributes: The proposed DT is a virtual representation of physical users, containing multiple UDTs. There exist two kinds of DT data, i.e., networking-related data and behavior-related data. The former includes users' channel conditions, which can be utilized to estimate users' transmission capabilities. The latter includes users' locations, swipe timestamps, and preferences, which can be used to analyze users' mobility trajectories, swipe probability distributions, and video popularity. DT data originates from two aspects, i.e., realistic user-status data collected from the physical mobile communication network[4] and predicted user-status data from the LSTM model. DT data in resource reservation window N consists of four user-status vectors, i.e., locations, $\mathbf{Y}_N$, channel conditions, $\mathbf{H}_N$, swipe timestamps, $\mathbf{W}_N$, and preferences, $\mathbf{E}_N$. Each user-status vector is collected or predicted with different time frequencies [29] in UDTs during each resource reservation window.

2) DT Function Modules: The proposed DT includes two function modules, i.e., status prediction and feature abstraction. Specifically, based on collected data from the physical mobile communication network, DT can conduct status prediction by using the LSTM model, which can effectively reduce data collection overhead. The output of status prediction module is the predicted user status for efficient network management. Additionally, DT can abstract accurate users' behavior patterns based on the data-driven methods.

B. DT Data Processing

As shown in Fig. 2, we present a detailed DTP procedure, which uses the same neural network structure in our previous work [29]. The DTP module mainly consists of three parts, i.e., autoencoder, DDQN, and K-means++ with probability statistics. Specifically, the autoencoder is responsible for compressing time-series DT data into low-dimensional latent features by learning to encode significant data features. The DDQN analyzes the latent features to determine the appropriate clustering number in the user clustering model and output model accuracy by using two neural networks to minimize the overestimation. Here, the accuracy refers to the probability of the action output in DDQN. Based on the compressed DT data and clustering number, the K-means++ method can realize fast user clustering while the probability statistics module can analyze the swipe probability distribution and recommended video list.

In this procedure, the autoencoder and DDQN occupy the majority of computing workload due to the hidden neural network layers. Considering the fluctuation of user dynamics, the light-weight autoencoder and DDQN can be used to process DT data when the user status is relatively static, which can effectively reduce DTP consumption.

To identify the relationship between DTP model size, user dynamics, and clustering accuracy, we conduct numerical experiments to fit the mathematical relationship. The user dynamics, denoted by ψ_N, adopts data variation to represent its value, given by:

$$\psi_N = \delta_1 \operatorname{var}(\mathbf{H}_N) + \delta_2 \operatorname{var}(\mathbf{Y}_N) + \delta_3 \operatorname{var}(\mathbf{W}_N) + \delta_4 \operatorname{var}(\mathbf{E}_N), \quad (1)$$

[3]We consider three DT models, i.e., LSTM model, autoencoder model, and DDQN model, in this work.

[4]To handle DT data noise and incompleteness, the Kalman Filter and linear interpolation methods can be used.

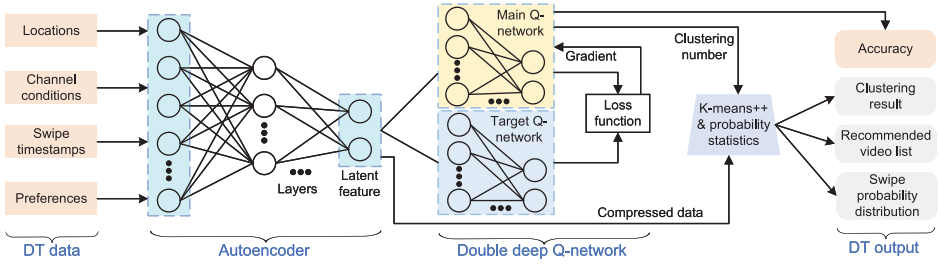


Fig. 2. DT data processing procedure.

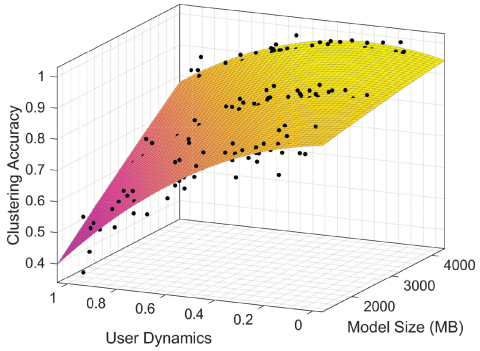


Fig. 3. Data fitting on DT data processing accuracy.

where $\delta_1, \delta_2, \delta_3, \delta_4$ represent the weighting factors of variations in different user-status vectors for normalization.

To guarantee the flexibility of DTP model, we set three kinds of model versions by adjusting the number of hidden neural network layers.[5] The model selection variable is a binary variable, denoted by e_i. If model version i is activated, $e_i = 1$; otherwise, $e_i = 0$. The model size of model version i is denoted by m_i in megabytes (MB), which is mainly determined by the weights and biases in the autoencoder and DDQN as well as the clustering number and data dimension, denoted by K and d, in the K-means++, as follows:

$$m_i = S_{\mathrm{p}} \left(\sum_{j=1}^{J} \left(w_{i,j}^{\mathrm{C}} + b_{i,j}^{\mathrm{C}} \right) + \sum_{j=1}^{J'} \left(w_{i,j}^{\mathrm{D}} + b_{i,j}^{\mathrm{D}} \right) \right) + K \times d \times S_{\mathrm{d}}, \tag{2}$$

where $w_{i,j}^{\mathrm{C}}$, $w_{i,j}^{\mathrm{D}}$, $b_{i,j}^{\mathrm{C}}$, and $b_{i,j}^{\mathrm{D}}$ are the number of weights and biases in a `Conv2D` and a `Dense` layer, respectively. Parameter J and J' are the numbers of `Conv2D` and `Dense` layers, respectively. Here, S_{p} and S_{d} represent the size of each parameter and data feature, respectively.

We conduct extensive simulation measurements of user dynamics and clustering accuracy under different DT model sizes, as shown in Fig. 3. It can be observed that with the increasing model size and decreasing user dynamics, the clustering accuracy gradually rises and approaches 1. The quadratic polynomial in the Curve Fitting Toolbox[6] is utilized to fit the mathematical relationship[7] among DT model size, user dynamics, and clustering accuracy, as follows:

$$f(m_i, \psi_N) = \mathrm{p}_1 + \mathrm{p}_2 m_i + \mathrm{p}_3 \psi_N + \mathrm{p}_4 m_i^2 + \mathrm{p}_5 m_i \psi_N + \mathrm{p}_6 \psi_N^2, \tag{3}$$

where $f(m_i, \psi_N)$ is the clustering accuracy with model size m_i and user dynamics ψ_N.

C. DT Model Update

Without adaptive parameter adjustment, the LSTM model and autoencoder-based DDQN model in DT may gradually become inaccurate due to the evolving user dynamics. Therefore, we use an improved incremental learning method to update DT models in an online manner. Specifically, the incremental learning for the LSTM model update adopts elastic weight consolidation on newly collected training batches to prevent knowledge forgetting [30]. In the autoencoder-based DDQN model, the autoencoder integrates new training data, which is fine-tuned at low learning rates to preserve learned features. The DDQN model uses experience replay to maintain a buffer containing both new and old experiences for training.

The incremental learning is utilized to update DT models by using the mini-batch stochastic gradient descent [31], as follows:

$$\theta_{t+\mathtt{b}} \leftarrow \theta_t - \frac{\eta}{\mathtt{b}} \nabla_{\theta_t} \sum_{z \sim \mathtt{b}} \mathcal{L}(F(s_{t+z}; \theta_t), y_{t+z}), \tag{4}$$

where θ_t and s_t are the neural network parameters and input at time t, respectively. Here, functions $\mathcal{L}$ and F represent the loss function and strategic function, respectively. Here, η and $\mathtt{b}$ represent the learning rate and batch size, respectively. In the DTU stage, multiple batches are sampled from experience replay to fine-tune DT model. The number of batches for LSTM, autoencoder, and DDQN model update are set to n_1, n_2, and n_3, respectively. Assuming that the relationship between model accuracy and training batches is logarithmic [32], [33], we construct a model accuracy update method for LSTM, autoencoder, and DDQN, as follows:

$$\Psi_{t+\tau_k n_k}^k = \Psi_t^k + \lambda_{k,1} \ln\left(\mathtt{b}^k n_k + \lambda_{k,2} \right), \tag{5}$$

where index k, ranging from 1 to 3, represents different DT models of LSTM, autoencoder, and DDQN, respectively. Parameters $\lambda_{k,1}$ and $\lambda_{k,2}$ represent the logarithmic parameters for DT model k, which reflect the rate of change in model accuracy as training batches increase.

[5]The number of model versions can be various for different scenarios. We set it as 3 for an example to demonstrate our idea.

[6]https://www.mathworks.com/help/curvefit/

[7]In our considered scenario, $\mathrm{p}_1 = 0.8246$, $\mathrm{p}_2 = 3.793 \times 10^{-5}$, $\mathrm{p}_3 = -0.2262$, $\mathrm{p}_4 = -2.044 \times 10^{-9}$, $\mathrm{p}_5 = 3.931 \times 10^{-5}$, $\mathrm{p}_6 = -0.3294$. The coefficient of determination and root mean square error (RMSE) are 0.9334 and 0.0326, which indicate that the fitting function has high fitting accuracy.

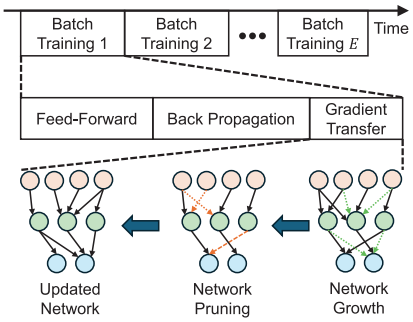


Fig. 4. The procedure of DT model update process.

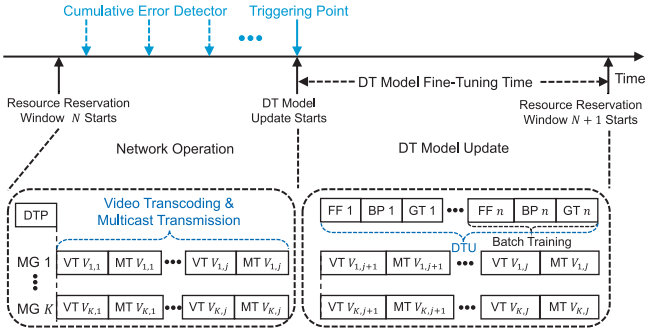


Fig. 5. The procedure of adaptive DTP and DTU.

As shown in Fig. 4, DTU consists of multiple batch training. Each batch training consists of three parts, i.e., feed-forward, back-propagation, and gradient transfer. In the gradient transfer, there exist three stages, i.e., network growth for activating connections with large gradients, network pruning for removing connections with small gradients, and network update for updating connections. Accordingly, the computation load can be expressed as

$$C_{\text{operation}} = \mathcal{F}_{\text{operation}} + \mathcal{B}_{\text{operation}} + \mathcal{G}_{\text{operation}}, \quad (6)$$

where $\mathcal{F}_{\text{operation}}$, $\mathcal{B}_{\text{operation}}$, and $\mathcal{G}_{\text{operation}}$ represent the computation load of feed-forward, back-propagation, and gradient transfer operations, respectively. The computation load of feed-forward operation, $\mathcal{F}_{\text{operation}}$, equals to that of the back-propagation operation, $\mathcal{B}_{\text{operation}}$, which is related to neural network structures, calculated as:

$$\begin{cases} \mathcal{F}^{\text{LSTM}}_{\text{operation}} = 4\mathfrak{b}^{\text{LSTM}}\wp\left(d \cdot H + H^2 + H\right), \\ \mathcal{F}^{\text{Auto}}_{\text{operation}} = \mathfrak{b}^{\text{Auto}}(W_{\text{enc}} + W_{\text{dec}}), \\ \mathcal{F}^{\text{DDQN}}_{\text{operation}} = \mathfrak{b}^{\text{DDQN}}\sum_i W_i, \end{cases} \quad (7)$$

where $\wp$ and H represent the sequence length and the number of neurons in the hidden layers of LSTM model, respectively. Here, W_{enc}, W_{dec}, and W_i represent the weight operations in the encoder, decoder, and DDQN layer i, respectively.

Next, we derive the computation load of gradient transfer operations in the proposed network growth and pruning mechanism. Specifically, *in the network growth stage*, the connections between neurons with large gradients are activated. For connection w, a mask value is used and set to be 1 when its gradient exceeds the $\mathfrak{a}$ percentile of the gradient of overall weight vector W_{grad}. We utilize an indicator function to define this operation on the mask value, i.e., $\mathbb{I}(|w_{\text{grad}}| \geq \text{Pr}_{\mathfrak{a}}(W_{\text{grad}}))$, where w_{grad} is the gradient of connection w. To guarantee that the newly activated connections can be quickly adjusted in the direction of reducing loss, the gradient needs to integrate with the current learning rate, i.e., $w_{\text{grad}} \leftarrow \eta \cdot w_{\text{grad}}$. *In the network pruning stage*, connections of neurons with small gradients are removed. For connection w, a mask value is used and set to be 0 when its gradient is below the $\mathfrak{c}$ percentile of the gradients of overall weight vector, which can be expressed as

$$w_{\text{grad}} \leftarrow w_{\text{grad}} \cdot \mathbb{I}\left(|w_{\text{grad}}| \geq \text{Pr}_{\mathfrak{c}}(W_{\text{grad}})\right). \quad (8)$$

After network growth and pruning, the network connections are updated, and a gradient transfer process is completed. During each network pruning and growth stage, assuming that each weight undergoes addition and multiplication with the same probability, the computational load, represented by $\mathcal{G}_{\text{operation}}$, equals twice the number of network parameters.

D. Adaptive DT Model Update and Data Processing Procedure

As shown in Fig. 5, we propose an adaptive DTU and DTP mechanism. In the network operation stage, an appropriate DTP model is first activated to implement MG update. Then, video transcoding (VT) and multicast transmission (MT) are started based on real-time 3C management from the network controller. A cumulative error detector is used to monitor the accuracy of predicted user status and clustering result. If the cumulative error exceeds a prescribed value, the network enters the DTU stage. In this stage, DTU is implemented in parallel with video transcoding and multicast transmission. The network controller determines real-time 3C management for video transcoding, multicast transmission and DTU. When the DTU is completed, a new network operation stage is started.

We first need to determine when to trigger DTU. Since the output of the LSTM model is predicted user status, we utilize the average error between predicted user status and collected one to trigger DTU. The average error is given by

$$\Theta^{\text{LSTM}}_t = \frac{1}{\|t - t_0\|}\Big(\left|\mathbf{H}_{t_0\sim t} - \widetilde{\mathbf{H}}_{t_0\sim t}\right| + \left|\mathbf{Y}_{t_0\sim t} - \widetilde{\mathbf{Y}}_{t_0\sim t}\right| + \left|\mathbf{W}_{t_0\sim t} - \widetilde{\mathbf{W}}_{t_0\sim t}\right| + \left|\mathbf{E}_{t_0\sim t} - \widetilde{\mathbf{E}}_{t_0\sim t}\right|\Big), \quad (9)$$

where t_0 and t represent the beginning and current time in the current resource reservation window. Vectors $\widetilde{\mathbf{H}}_{t_0\sim t}$, $\widetilde{\mathbf{Y}}_{t_0\sim t}$, $\widetilde{\mathbf{W}}_{t_0\sim t}$, and $\widetilde{\mathbf{E}}_{t_0\sim t}$ are collected users' channel conditions, locations, swipe timestamps, and preferences from time t_0 to t, respectively. If the average error exceeds the threshold Γ_1, the LSTM model is updated using incremental learning.

Similarly, the autoencoder model in the DTP module uses the average error between the input user status and the reconstructed one to trigger DTU. Finding an adaptive trigger point for DDQN model update can be challenging since it is hard to obtain a realistic and optimal user clustering result for error estimation. Since the action probability distribution of DDQN can reflect the accuracy of decision-making, we utilize

its entropy to trigger DTU. The entropy is given by:

$$\Theta_t^{\mathrm{DDQN}} = -\frac{1}{\|t-t_0\|}\sum_{t=t_0}^{t}\sum_{j=1}^{\mathcal{D}} P\big(y_{t,j}\big)\log P\big(y_{t,j}\big), \quad (10)$$

where $P(y_{t,j})$ is the output probability of action j in time t, and $\mathcal{D}$ is the action dimension. The update thresholds of autoencoder and DDQN are denoted by Γ_2 and Γ_3.

After determining the adaptive trigger point for DTU, we need to analyze how many training batches should be used for different DT models. The time duration of DTU is up to the number of training batches and allocated computing resources for batch training. The time duration of per batch training for DT model k update, denoted by τ_k, can be expressed as

$$\tau_k = \frac{\mu_1 C^k_{\mathrm{operation}} \mathbb{I}(\Theta_k \geq \Gamma_k)}{\mathcal{C}_k^{\mathrm{DTU}}}, \quad (11)$$

where Γ_k and μ_1 represent the model update threshold of model k and the computing density for model update. Here, $\mathcal{C}_k^{\mathrm{DTU}}$ is the allocated computing resources for model k update. Based on Eq. (5), we can analyze how many training batches are needed for DT model k to increase accuracy q_k, i.e.,

$$q_k = \lambda_{k,1}\ln\Big(b^k\widetilde{n}_k + \lambda_{k,2}\Big), \forall k \in \{1,2,3\}, \quad (12)$$

where q_k is the gap between the current accuracy of model k and specified accuracy of model k. Here, k ranging from $1 \sim 3$ corresponds to the LSTM model, autoencoder model, and DDQN model, respectively, in the constructed DT. By solving this equation, the number of training batches, $\widetilde{n}_k$, for DT model k to obtain q_k accuracy increase is

$$\widetilde{n}_k = \Big\lceil \big(\exp(q_k/\lambda_{k,1}) - \lambda_{k,2}\big)/b^k \Big\rceil, \quad (13)$$

where $\lceil \cdot \rceil$ is the ceiling function.

By combining the number of batches, n_k, for DT model k update, the time duration, φ, of overall DTU is given by

$$\varphi = \sum_{k=1}^{3}\tau_k\widetilde{n}_k. \quad (14)$$

To select an appropriate DT model version to process user status data, DTP activates in the initial network operation stage in Fig. 5. The DT model version selection e_i is based on future user dynamics. The range of user dynamics ψ_N is split into three parts, i.e., $[0, 0.33\kappa)$, $[0.33\kappa, 0.66\kappa)$, and $[0.66\kappa, \kappa]$, which corresponds to DT model version i from 1 to 3. Parameter κ represents the threshold of user dynamics.

V. Fine-Grained QoE Model Establishment

In this section, we establish a fine-grained QoE model, which depends on four aspects, i.e., buffer control, multicast rebuffer time, video quality, and video quality variation.

A. Buffer Control

To guarantee smooth and high-quality video playback, accurate buffer control is necessary, which includes the number and version of segments to be buffered in scheduling slot t. The buffer control should consider two perspectives, i.e., 1) buffer perspective: buffer minimum segments to avoid buffer length empty; 2) resource perspective: buffer maximum segments based on available communication and computing resources.

From the buffer perspective, the number of buffered segments, $\Lambda^1_{t,g}$, should guarantee MG g's buffer length is larger than scheduling slot length, T_{s}, to avoid the playback lags. Therefore, variable $\Lambda^1_{t,g}$ is determined by

$$\Lambda^1_{t,g} = \big[(T_s - Q_{t,g})/\chi\big]^+, \quad (15)$$

where $Q_{t,g}$ and χ represent users' average buffer length in MG g at scheduling slot t and the time duration of one segment, respectively.

From the resource perspective, the number of buffered segments, $\Lambda^2_{t,g}$, should guarantee video transcoding and multicast transmission can be completed in each scheduling slot. Variable $\Lambda^2_{t,g}$ is determined by

$$\begin{cases} \sum_{x=1}^{X_{t,g}}\sum_{l=1}^{\bar{l}} z^l_{t,g,x} \leq o\,T_s \min\limits_{u\in\mathcal{U}_g}\widehat{r}_{t,u} \leq \sum_{x=1}^{X_{t,g}+1}\sum_{l=1}^{\bar{l}} z^l_{t,g,x}, \forall g \in \mathcal{G}, \\ \sum_{x=1}^{\widetilde{X}_{t,g}}\sum_{l=2}^{\bar{l}} z^l_{t,g,x} \leq \frac{(1-o)T_s\widehat{\mathcal{C}}_{t,g}}{\mu_3} \leq \sum_{x=1}^{\widetilde{X}_{t,g}+1}\sum_{l=2}^{\bar{l}} z^l_{t,g,x}, \forall g \in \mathcal{G}, \end{cases} \quad (16)$$

where $z^l_{t,g,x}$ is the file size of segment x of version l for MG g at scheduling slot t, which adopts the SVC principle [34]. Here, $\widehat{r}_{t,u}$ and $\widehat{\mathcal{C}}_{t,g}$ are the estimated transmission and computing capabilities of user u and MG g based on the recent statistical information. Parameter μ_3 is the computing density of video transcoding. Here, l and $\bar{l}$ represent the segment version index and average segment version. Since video transcoding is related to the enhancement layers, the segment version index ranges from 2 to $\bar{l}$. Here, o is the ratio of time occupation between multicast transmission and video transcoding processes, which is set to 0.5, because these two processes usually consume almost equal time. Since Eq. (16) consists of monotonically increasing functions, the maximum segment buffering numbers, denoted by $X_{t,g}$ and $\widetilde{X}_{t,g}$, can be uniquely determined by increasing their respective values. To satisfy network resource constraints, we select the smaller one between $X_{t,g}$ and $\widetilde{X}_{t,g}$ as the number of buffered segments, i.e., $\Lambda^2_{t,g} = \min\{X_{t,g}, \widetilde{X}_{t,g}\}$.

Based on the buffer and resource perspectives, we can determine the segment buffering number, i.e., $\Lambda_{t,g} = \max\{\Lambda^1_{t,g}, \Lambda^2_{t,g}\}$. The above analysis of segment buffering number depends on the average segment version. To guarantee users' high-quality video playback, it is important to determine the segment version selection in each time scheduling slot, which will be analyzed in the following subsections.

$$R_t(\boldsymbol{\mathcal{C}_t}, \boldsymbol{B}_t, \boldsymbol{v}_t) = \frac{\mu_2\sum_i e_i m_i}{\mathcal{C}} + \frac{1}{G}\sum_{g=1}^{G}\left[\left(\frac{\mu_3\sum_l\sum_x v^l_{t,g}z^l_{t,g,x}}{\mathcal{C}_{t,g}} + \frac{\sum_l\sum_x v^l_{t,g}z^l_{t,g,x}}{\sum_i f(e_i m_i, \psi_{t,g})\min\limits_{u\in\mathcal{U}_g} B_{t,g}\log\Big(1+\frac{|h_{t,u}|^2 P_{\mathrm{D}}}{N_0+N_{\mathrm{I}}}\Big)} - Q_{t,g}\right)\right]^+, \quad (17)$$

B. Multicast Rebuffer Time

The multicast rebuffer time is related to the DTP latency, as well as real-time video transcoding and multicast transmission latency, and buffer length. The average multicast rebuffer time of all MGs is expressed in Eq. (17), shown at the bottom of the previous page.

where $\mathcal{C}_{t,g}$ and $B_{t,g}$ represent the allocated computing and bandwidth resources for MG g in scheduling slot t, respectively. To ensure that all users in MG g can receive segments, the transmission capability supported by the worst user channel condition in MG g is selected [35]. In Eq. (17), $h_{t,u}$, P_{D}, N_0, and N_{I} refer to user u' channel gain, transmission power, noise power, and interference power, respectively. Parameter μ_2 represents the computing density of DTP. Additionally, $v_{t,g}^l$ is a binary variable. If segment version l is selected for MG g in scheduling slot t, $v_{t,g}^l = 1$; otherwise, $v_{t,g}^l = 0$. It needs to satisfy

$$\sum_{l} v_{t,g}^l = 1, \forall t \in [t_N, t_{N+1}], g \in \mathcal{G}, \tag{18}$$

which can guarantee only one segment version is selected for MG g to avoid repeated video transcoding and multicast transmission. Here, t_{N+1} denotes the beginning time of resource reservation window $N+1$.

Furthermore, $\mathcal{C}_t^{\mathrm{DTP}}$, $\mathcal{C}_t^{\mathrm{DTU}}$, and $\mathcal{C}_{t,g}$ are the allocated computing resources for DTP, DTU and MG g in scheduling slot t, which have the following constraints:

$$\begin{cases} \mathcal{C}_t^{\mathrm{DTP}} = \mathcal{C}, & \forall t \in [t_N, t_N + \varsigma_N), \\ \sum_g \mathcal{C}_{t,g} = \mathcal{C}, & \forall t \in [t_N + \varsigma_N, t_{N+1} - \varphi_N), \\ \mathcal{C}_t^{\mathrm{DTU}} + \sum_g \mathcal{C}_{t,g} = \mathcal{C}, & \forall t \in [t_{N+1} - \varphi_N, t_{N+1}], \end{cases} \tag{19}$$

where ς_N is the DTP time in resource reservation window N, expressed by the first term of the right side in Eq. (17). If there exists no DTU, the time duration of resource reservation window N is set to a fixed value, $\mathcal{W}$, otherwise, we can have $t_{N+1} = t_N^{\mathrm{S}} + \varphi_N$, where t_N^{S} is the start time of DTU in resource reservation window N. Additionally, $Q_{t,g}$ is the buffer length of MG g, which is updated based on the time length of downloaded segment, as follows:

$$Q_{t+1,g} = \left[Q_{t,g} - T_{\mathrm{s}} + \min\left\{1, \frac{T_{\mathrm{s}}}{T_{t,g}^{\mathrm{V}} + T_{t,g}^{\mathrm{M}}}\right\} \Lambda_{t,g} \chi \right]^+, \tag{20}$$

where $T_{t,g}^{\mathrm{V}}$ and $T_{t,g}^{\mathrm{M}}$ represent the video transcoding and multicast transmission delay of MG g in scheduling slot t, which are expressed in the first two terms in $[\,\cdot\,]^+$ function in Eq. (17), respectively. Here, $[x]^+$ equals to $\max\{x, 0\}$. In addition, the minimization function is used to analyze the ratio of transmitted segments.

C. Video Quality

structural similarity index measure (SSIM) is a common metric to assess video quality by comparing luminance, contrast, and structural characteristics. The relationship between video bitrate, z, and SSIM, O, can be depicted as $O = 1 - \frac{1}{2z+1}$ [36], where a higher O means a higher video quality. Based on this mathematical relationship, the average video quality of new buffered segments in all MGs in scheduling slot t is expressed as

$$O_t(\boldsymbol{v}_t) = \frac{1}{G} \sum_{g=1}^{G} \sum_{x=1}^{\Lambda_{t,g}} 1 - \frac{1}{2 \sum_l v_{t,g}^l z_{t,g,x}^l + 1}. \tag{21}$$

D. Video Quality Variation

Based on the video quality in Eq. (21), we can further analyze the video quality variation between adjacent segments in all MGs. Since video quality decrease can have a negative effect on user QoE, we utilize the video quality decrease between adjacent segments to measure the video quality variation. Specifically, if the video quality at time t is lower than that at time $t-1$, video quality variation, defined by J_t, equals to $O_{t-1} - O_t$; otherwise, $J_t = 0$. Therefore, video quality variation can be expressed as

$$J_t(\boldsymbol{v}_t) = [O_{t-1}(\boldsymbol{v}_{t-1}) - O_t(\boldsymbol{v}_t)]^+. \tag{22}$$

E. Fine-Grained QoE Model

The proposed fine-grained QoE model consists of three parts, i.e., multicast rebuffer time, video quality, and video quality variation. We adopt a linear weighted function, which is widely used to integrate these factors [37], to reflect users' satisfaction with network management. The constructed QoE model is expressed as

$$\begin{aligned} QoE_t(\boldsymbol{\mathcal{C}_t}, \boldsymbol{B}_t, \boldsymbol{v}_t) = \varpi_1 O_t(\boldsymbol{v}_t) - \varpi_2 R_t(\boldsymbol{\mathcal{C}_t}, \boldsymbol{B}_t, \boldsymbol{v}_t) \\ - \varpi_3 J_t(\boldsymbol{v}_t), \end{aligned} \tag{23}$$

where ϖ_1, ϖ_2 and ϖ_3 represent MGs' sensitivity degrees of video quality, rebuffer time and quality variation, respectively.

VI. Problem Formulation and Solution

In this section, we first formulate the adaptive DT-assisted 3C management problem and then give our solution.

A. Problem Formulation

Due to MGs' dynamic video requests and swipe behaviors, bandwidth and computing resources need to be flexibly and accurately allocated to DT operation as well as video transcoding and multicast transmission, to reduce playback rebuffer time. Furthermore, the versions of transmitted segment should be as high and consistent as possible to improve video quality and reduce video quality variation. Therefore, the objective is to maximize MGs' long-term QoE by jointly optimizing bandwidth and computing resource allocation, as well as segment version selection at each scheduling slot. The formulated problem **P1** is given by

$$\mathbf{P1:} \max_{\{\boldsymbol{\mathcal{C}}_t, \boldsymbol{B}_t, \boldsymbol{v}_t\}_{t \in \mathcal{T}}} \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} QoE_t(\boldsymbol{\mathcal{C}_t}, \boldsymbol{B}_t, \boldsymbol{v}_t) \tag{24}$$

$$\text{s.t.} \quad (18), (19),$$

$$\sum_{g=1}^{G} B_{t,g} \le B, \forall t \in \mathcal{T}, \tag{24a}$$

$$\mathcal{C}_{t,g}, \mathcal{C}_t^{\mathrm{DTU}} \in [0, \mathcal{C}], \forall t \in \mathcal{T}, g \in \mathcal{G}, \tag{24b}$$

$$B_{t,g} \in [0, B], \forall t \in \mathcal{T}, g \in \mathcal{G}, \tag{24c}$$
$$v^l_{t,g} \in \{0, 1\}, \forall t \in \mathcal{T}, g \in \mathcal{G}, l \in \mathcal{L}. \tag{24d}$$

Constraint (24a) guarantees that the allocated bandwidth resources to MGs do not exceed the network bandwidth limit.

B. Problem Transformation

Since the formulated problem is mixed-integer nonlinear programming with a complex objective function, it is hard to directly use a model-based method to solve it. Considering that the transition of users' playback status satisfies the Markov properties and the objective is to maximize the long-term QoE, the optimization problem **P1** can be modeled as a Markov decision process (MDP). MDP is a discrete-time stochastic control process, including four essential elements, i.e., state, action, state transition probability, and reward. At step (scheduling slot) t, the agent, i.e., network controller, makes action a_t only based on current state s_t. Then, state s_t can be evolve into s_{t+1} according to state transition probability $\Pr(s_{t+1}|s_t, a_t)$, which can obtain reward r_t.

1) State: The state includes six elements, i.e., user dynamics ψ_t, MGs' worst channel gains $\widetilde{\boldsymbol{h}}_t$, DT model size m_t, DT model accuracy Ψ_t, DTU time duration φ_t, and MGs' buffer lengths $\boldsymbol{Q}_t$, which can be expressed as

$$s_t = \left\{\psi_t, \left\{\widetilde{h}_{t,g}\right\}_{g\in\mathcal{G}}, m_t, \Psi_t, \varphi_t, \left\{Q_{t,g}\right\}_{g\in\mathcal{G}}\right\}. \tag{25}$$

2) Action: The action consists of optimization variables, i.e., computing resource allocation variable $\boldsymbol{\mathcal{C}}_t$, bandwidth resource allocation variable $\boldsymbol{B}_t$, and segment version selection variable $\boldsymbol{v}_t$. Since discrete variable $\boldsymbol{v}_t$ hardly directly uses a gradient descend algorithm for neural network training, we relax it to a continuous vector, $\widetilde{\boldsymbol{v}}_t$, ranging from 0 to 1. When any element in vector $\widetilde{\boldsymbol{v}}_t$ belongs to [0,0.5), the element will be rounded to 0; otherwise, it will be rounded to 1. Based on the above analysis, the action can be expressed as

$$a_t = \{\boldsymbol{\mathcal{C}}_t, \boldsymbol{B}_t, \widetilde{\boldsymbol{v}}_t\}. \tag{26}$$

Furthermore, to guarantee that vectors $\boldsymbol{C}_t$ and $\boldsymbol{B}_t$ can satisfy the constraints, the output action of each element in the vectors needs to be normalized.

3) State Transition Probability: The state transition probability is given by

$$\begin{aligned}\Pr(s_{t+1}|s_t, a_t) = \Pr(\psi_{t+1}|\psi_t) \prod_{g\in\mathcal{G}} \Pr\left(\widetilde{h}_{t+1,g}|\widetilde{h}_{t,g}\right) \cdot \\ \Pr(m_{t+1}|m_t) \Pr(\Psi_{t+1}|\Psi_t, \boldsymbol{\mathcal{C}}_t) \Pr(\varphi_{t+1}|\varphi_t, \boldsymbol{\mathcal{C}}_t) \cdot \\ \prod_{g\in\mathcal{G}} \Pr\left(Q_{t+1,g}|Q_{t,g}, \boldsymbol{\mathcal{C}}_t, \boldsymbol{B}_t, \boldsymbol{v}_t\right).\end{aligned} \tag{27}$$

The equation holds due to the independence of different state components. The first two components are evolved based on user and channel dynamics. The third component is evolved based on the adaptive DT mechanism, while the fourth and fifth components are evolved based on the computing resource allocation. The last component is the evolution of MGs' buffers based on the 3C management decision.

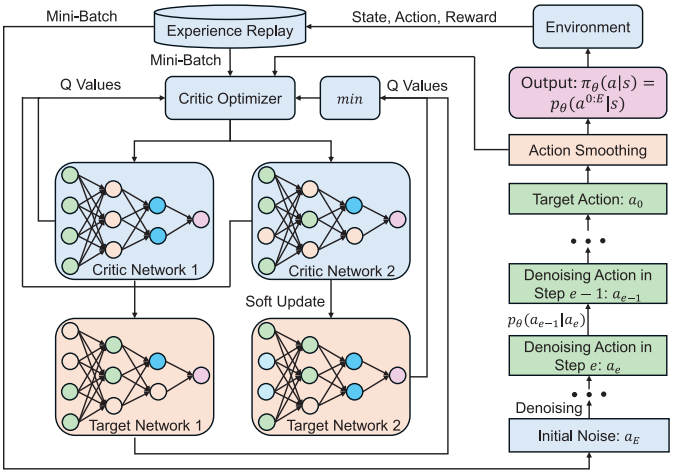


Fig. 6. The proposed diffusion-based TD3 algorithm framework.

4) Reward: The reward is designed to maximize MGs' QoE in Eq. (24) in step t, which is defined as $r_t = QoE_t(\boldsymbol{\mathcal{C}}_t, \boldsymbol{B}_t, \boldsymbol{v}_t)$.

C. Diffusion-Based TD3 Scheduling Algorithm

As shown in Fig. 6, we have proposed a diffusion-based twin delayed deep deterministic policy gradient (TD3) scheduling algorithm [13] that enhances the actor-network for 3C management. This approach utilizes a diffusion model, which is a method for gradually introducing variability into the decision-making process. Unlike traditional techniques that add a fixed amount of noise, the diffusion model adjusts noise levels dynamically during action generation. This unique feature enables our algorithm to explore a wider range of potential actions, potentially discovering superior strategies that standard methods might miss. Furthermore, the diffusion model's ability to produce a variety of actions from the same starting condition enhances the robustness of our scheme.

1) Diffusion-Based Actor: At each step, the network controller needs to make network decisions based on the current state s, i.e., $\pi_\theta(a|s)$. The reverse process of conditional diffusion models is utilized to modify the actor network. To distinguish the timestamps used in the TD3 training and the reverse process, we utilize subscripts $e \in \{1, \ldots, E\}$ to represent denoising steps.[8]

The reverse process is also known as the sampling process, which can infer target action a_0 from a noise sample $a_E \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. The transition from a_e to a_{e-1} in state s with network parameter θ is defined as $p_\theta(a_{e-1}|a_e, s)$, which follows a Gaussian distribution, as follows:

$$p_\theta(a_{e-1}|a_e, s) = \mathcal{N}\left(a_{e-1}; \mu_\theta(a_e, e, s), \sum_\theta(a_e, e, s)\right), \tag{28}$$

where the covariance matrix $\sum_\theta(a_e, e, s)$ is determined by $\beta_e \boldsymbol{I}$. We can parameterize the noise prediction model as

[8]To decrease the computational load required for diffusion-based TD3 model inference, we can reduce the number of denoising steps.

described in [38], where the mean value is expressed as:

$$\mu_\theta(a_e, s, e) = \frac{1}{\sqrt{\alpha_e}}\left(a_e - \frac{\beta_e}{\sqrt{1-\overline{\alpha}_e}}\boldsymbol{\varepsilon}_\theta(a_e, s, e)\right), \quad (29)$$

where $\boldsymbol{\varepsilon}$ is a approximator function. We have $\alpha_e = 1 - \beta_e$ and $\overline{\alpha}_e = \prod_{k=1}^{e} \alpha_k$. The forward process variance β_e can be learned by re-parameterization [39] or held as a constant. To sample a_e from $p_\theta(a_{e-1}|a_e, s)$, we can have the following relationship:

$$a_{e-1} = \frac{1}{\sqrt{\alpha_e}}\left(a_e - \frac{\beta_e}{\sqrt{1-\overline{\alpha}_e}}\boldsymbol{\varepsilon}_\theta(a_e, s, e)\right) + \sqrt{\beta_e}\mathcal{A}, \quad (30)$$

where $\mathcal{A}$ satisfies the Gaussian distribution, i.e., $\mathcal{A} \sim (\mathbf{0}, \boldsymbol{I})$. When e equals to 1, $\boldsymbol{\varepsilon}$ is set to 0 to enhance sample quality. The simplified training loss in [38] is used to train our conditional $\boldsymbol{\varepsilon}$-model, as follows:

$$\mathcal{L}_D(\theta) = \mathbb{E}_{e,\boldsymbol{\varepsilon},(s,a)}\left[||\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}_\theta(\sqrt{\overline{\alpha}_e}a_0 + \sqrt{1-\overline{\alpha}_e}\boldsymbol{\varepsilon}, s, e)||^2\right]. \quad (31)$$

2) Neural Network Training: The training process involves two key components: the actor and the critic networks. Two critic networks and their corresponding target networks are utilized to mitigate the overestimation bias and implement a delayed policy update mechanism. Specifically, the role of critic networks is to evaluate the policy by estimating action-value function $\mathcal{Q}(s, a)$. With two critic networks, $\mathcal{Q}_1$ and $\mathcal{Q}_2$, we can compute the minimum value between these two critic networks to avoid overestimation. The target networks are updated at a slower rate using a soft update rule. In contrast, the critic networks are updated by minimizing the loss calculated from the mean square error (MSE) between the predicted Q-values and the target Q-values. The target Q-value is computed by using the formula:

$$y_t = r_{t+1} + \gamma \min_{j=1,2} \mathcal{Q}\left(s_{t+1}, \widetilde{a}_{t+1}; \boldsymbol{w}_t^j\right), \quad (32)$$

where γ is the discount factor. Here, $\widetilde{a}_{t+1}$ is the smooth policy that adds the clipped noise, i.e.,

$$\widetilde{a}_{t+1} = \texttt{clip}(a_{t+1} + \widetilde{\varepsilon}, [a_{\min}, a_{\max}]). \quad (33)$$

Parameter $\widetilde{\varepsilon}$ equals to $\texttt{clip}(\mathcal{N}(0, \widetilde{\sigma}^2), [-c, c])$, which denotes $\mathcal{N}(0, \widetilde{\sigma}^2)$ is truncated between $-c$ and c. Additionally, $\boldsymbol{w}_t^j$ is the network weight of $\mathcal{Q}$ network j at step t. The weights of two critic networks are updated based on MSE, as follows:

$$\boldsymbol{w}_{t+1}^j = \arg\min_{\boldsymbol{w}_t^j} \frac{1}{|\mathtt{B}_t|}\sum_{k=1}^{|\mathtt{B}_t|}\left(y_t - \mathcal{Q}\left(s_k, a_k; w_t^j\right)\right)^2, \quad (34)$$

where $\mathtt{B}_t$ is the training batch at step t.

In the policy improvement stage for the diffusion-based actor π_θ, the function designated for actor updates integrates two main components: the policy regularization $\mathcal{L}_D(\theta)$ and the policy improvement $\mathcal{L}_q(\theta)$. The Q-value-based policy improvement $\mathcal{L}_q(\theta)$ can guide the diffusion model to prefer the action that is expected to yield a higher Q value. Therefore, our policy learning objective for actor updates is given by

$$\pi = \arg\min_{\pi_\theta} \mathcal{L}_D(\theta) - \xi \cdot \mathbb{E}_{s\sim\mathtt{B}, a_0\sim\pi_\theta}\left[\min_j \mathcal{Q}\left(s, a_0, \boldsymbol{w}^j\right)\right], \quad (35)$$

Algorithm 1: Diffusion-Based TD3 (DFFTD3)

1 **Input:** Bandwidth and computing resource capacities B and $\mathcal{C}$, video bitrate $\boldsymbol{z}$, computing density vector $\boldsymbol{\mu}$, DT model size $\boldsymbol{m}$, users' channel gains $\mathbf{H}$, and the number of training batches for DTU $\widetilde{\boldsymbol{n}}$;
2 **Output:** Network management decision, including bandwidth and computing resource allocation $\boldsymbol{B}$ and $\boldsymbol{\mathcal{C}}$, and segment version selection $\boldsymbol{v}$;
3 **Initialize:** Critic networks $\mathcal{Q}_1$, $\mathcal{Q}_2$ and policy π_θ with random parameters $\boldsymbol{w}_1$, $\boldsymbol{w}_2$, θ; Target networks $\boldsymbol{w}_1' \leftarrow \boldsymbol{w}_1$, $\boldsymbol{w}_2' \leftarrow \boldsymbol{w}_2$, $\theta' \leftarrow \theta$; Replay buffer $\mathcal{R}$;
4 **for** *each* episode **do**
5 Reset initial state s_1;
6 **for** *each step* $t \in \{1, \ldots, T\}$ **do**
7 Observe current state s and initialize a random normal distribution $a_E \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$;
8 **for** *denoising step* $e = E : 1$ **do**
9 Calculate the reverse transition distribution based on Eq. (28);
10 Calculate mean value μ_θ based on Eq. (29);
11 Calculate the action distribution based on Eq. (30);
12 **end**
13 Select action a based on Eq. (33);
14 Execute action a and observe reward r;
15 Store transition tuple (s, a, r, s') in $\mathcal{R}$;
16 **if** $t \mod \mathcal{F}_1 == 0$ *(time to update)* **then**
17 Compute target Q values based on Eq. (32);
18 Update Q-functions based on Eq. (34);
19 **end**
20 **if** $t \mod \mathcal{F}_2 == 0$ *(policy delay)* **then**
21 Update the policy based on Eq. (35);
22 Update target network parameters: $\boldsymbol{w}_j' \leftarrow \zeta \boldsymbol{w}_j + (1-\zeta)\boldsymbol{w}_j'$, for $j = 1, 2$, $\theta' \leftarrow \zeta\theta + (1-\zeta)\theta'$;
23 **end**
24 **end**
25 **end**

where a_0 is re-parameterized by Eq. (30), which allows the gradient of the Q value with respect to the action to propagate backward through the entire diffusion sequence. Parameter ξ is the weighting factor to balance the influence between the diffusion process and the actor-critic process. The detailed algorithm procedure of proposed diffusion-based TD3 (DFFTD3) for 3C management is presented in Algorithm 1.

In Algorithm 1, critical steps include initializing the actor and critic networks at lines 3-4, followed by a loop where the current state is observed and noise is introduced for diffusion at lines 6-7. The core of the algorithm involves a diffusion process detailed from lines 8-12, where the action is refined through several denoising steps, calculating the reverse transition and mean action values to generate a new action distribution. Actions are then selected based on the updated policy and executed to observe rewards at lines 13-14. Critically, the network undergoes periodic updates, where

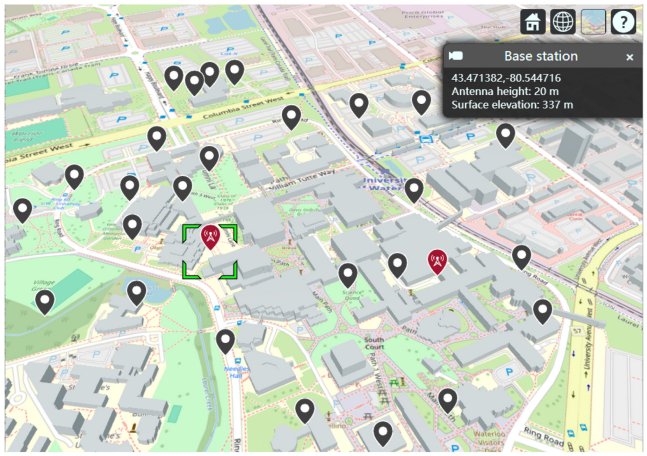


Fig. 7. The simulation scene, where BSs and users are represented by red and gray icons, respectively.

Q-functions are updated every F_1 steps and the policy network every F_2 steps as specified in lines 16-21.

3) Time Complexity Analysis: In the training phase, the training time complexity of the diffusion-based TD3 algorithm is mainly determined by the operations in both critic networks and the actor network with the diffusion process. Specifically, for a fully connected neural network, the time complexity of one forward pass is proportional to the number of layers L_c and the number of neurons per layer n_c. Since there are two critic networks and each network processes a batch of size $\mathtt{B}$, the total complexity for updating both critic networks in one iteration is $O(2 \cdot \mathtt{B} \cdot L_c \cdot n_c^2)$. The actor network involves multiple forward passes to denoise the initial action. Each forward pass through the actor network has a complexity of $O(L_a \cdot n_a^2)$, where L_a is the number of layers and n_a is the number of neurons in the actor network. Since the diffusion process involves e denoising steps, the total complexity for generating actions in the actor network is $O(e \cdot L_a \cdot n_a^2)$. Overall, the total time complexity for one training iteration is $O(2 \cdot \mathtt{B} \cdot L_c \cdot n_c^2) + O(e \cdot L_a \cdot n_a^2)$.

In the inference phase, the inference time complexity is primarily driven by the actor network with the diffusion process, as the critic networks are not used during inference. For each forward pass through the actor network, the complexity is $O(L_a \cdot n_a^2)$. The diffusion process involves e denoising steps, meaning e forward passes are required. Therefore, the total inference complexity is $O(e \cdot L_a \cdot n_a^2)$.

VII. Simulation Results

We conduct extensive simulations on the real-world dataset to evaluate the performance of the proposed adaptive DT-assisted 3C management scheme.

A. Simulation Setup

We consider a scenario where two BSs are deployed at the University of Waterloo campus to provide MSVS services for 60 users. Users' initial distribution is shown in Fig. 7, moving at speeds of 2∼5 km/h. Their channel path losses are calculated using `propagationModel` at MATLAB. Users' swipe behaviors are generated based on the short video streaming dataset.[9] We use the YouTube 8M dataset[10] to sample 1000 short videos across 8 categories, i.e., Entertainment, Games, Food, Sports, Science, Dance, Travel, and News. Each video sequence of 30 sec time duration is encoded into 15 segments. The network parameters are presented in Table II.

TABLE II
Communication Network Parameters

Parameter	Value	Parameter	Value
B	$[1.2, 2.2]$ MHz	η	0.001
C	$[7, 12]$ Gcycles/s	$\mathtt{b}$	64
T_{s}	4 sec	$\lambda_{1,1}, \lambda_{1,2}$	$\{0.015, 100\}$
δ_1	3×10^8	$\lambda_{2,1}, \lambda_{2,2}$	$\{0.013, 200\}$
δ_2	0.4	$\lambda_{3,1}, \lambda_{3,2}$	$\{0.011, 1200\}$
δ_3	0.03	ϖ	$\{1.5, 0.4, 1.4\}$
δ_4	0.01	N_{I}	-100 dBm
μ_1	2×10^7	N_0	-174 dBm
μ_2	1×10^7 Cycles/MB	P_{D}	27 dBm
μ_3	2×10^8 Cycles/MB	κ	12
$\mathtt{a}$	30	$\mathtt{c}$	5

TABLE III
DFFTD3 Algorithm Parameters

Parameter	Value	Parameter	Value
Memory size	5000	Episode length	180
Initial epsilon	1	Foward process variance	$[10^{-3}, 0.02]$
Epsilon decay	0.99	Noise clip	0.1
Final epsilon	0.05	Denoising steps	10
Number of episodes	500	Neural network layer connection	Dense
Hidden layer structure	512×512× 256 × 256	Activation function	ReLu
Batch size	128	Policy frequency	2
Actor learning rate	10^{-3}	Critic learning rate	10^{-4}

The proposed DFFTD3 neural network integrates a diffusion-based actor-network to generate actions and uses target networks to stabilize the learning process. The critic networks consist of two separate neural networks designed to estimate the Q-values in parallel. In the action selection, the trade-off between exploration and exploitation is considered, with noise added at a given timestep to simulate the reverse diffusion process. The neural network parameter setting is shown in Table III.

The constructed DT comprises two modules: 1) status prediction and 2) data feature abstraction, developed in Python 3.9. The status prediction module uses the Levy flight model to predict users' trajectories and analyzes channel fading to predict real-time channel conditions. It also samples users' swipe behaviors and preferences from a real-world dataset. A two-layer LSTM with the activation functions 'tanh' for

[9]ACM MM Grand Challenges: https://github.com/AItransCompetition/Short-Video-Streaming-Challenge/tree/main/data.

[10]YouTube 8M dataset: https://research.google.com/youtube8m/index.html.

the LSTM gates and 'sigmoid' for the recurrent step is used to make data predictions. At the beginning of each resource reservation window, the LSTM returns the full sequence of DT status. The learning rate is set at 0.001, utilizing the Adam optimizer in handling sparse gradients. The loss function is based on the MSE of DT status data. To combat overfitting, a dropout rate of 20% is applied to the inputs and the connections within the LSTM layers. The data feature abstraction module uses a two-stage probability statistics method to abstract the watching probability distribution, with further details in [40].

We compare the proposed adaptive DT-assisted 3C management scheme with the following benchmark schemes:

- Fixed DT Operation-Based (FDT) scheme: The DTU and DTP use the fixed update frequency and model version. The former is updated every 5 min, while the latter utilizes the DT model version 2. In each scheduling slot, the 3C management decision is solved by the proposed DFFTD3 algorithm.
- Myopic Optimization-based (MOP) scheme [20]: The DTU and DTP use the same strategies as the FDT scheme. Bandwidth and computing resources are divided into 10 equal units, respectively. During each scheduling slot, bandwidth units are sequentially allocated to each MG across four segment versions, and computing units are allocated to the DTU and MGs. The allocation iteration yielding the highest QoE determines the combination of bandwidth and computing resources for a specific segment version.
- Model-based Branch DQN (MBDQN) [40] scheme: The DTU and DTP use the same mechanism in the proposed adaptive DT scheme. In each scheduling slot, BDQN is responsible for determining the segment version. The bandwidth and computing resources are split into 10 equal units, which are allocated to DTU and MGs based on the branch and bound method [41]. The joint 3C management decision is fed back to the MSVS environment to update the BDQN network.
- Hierarchical Reward-based TD3 (HRTD3) scheme [23]: The DTU and DTP use the same mechanism in the proposed adaptive DT scheme. The action discretization uses the same principle in the proposed scheme. The reward function is a two-layer representation, i.e., basic reward and enhanced reward. The former depends on the violation degrees of constraints, while the latter is based on the objective function.

B. Convergence Analysis

As shown in Fig. 8, we present the convergence performance of proposed adaptive DT and 3C management scheme. Each episode consists of 180 steps, where the time duration of each step is 4 sec. Two training trials with random seeds 10 and 30 are conducted to draw the corresponding envelope area and mean curve, where the random seeds are used for action exploration and experience replay. The reward of each episode is the average QoE of all steps within its episode. It can be observed that as the number of episodes increases, the reward gradually grows. When the number of episodes approaches nearly 260, the reward converges to a stable state, indicating that the proposed adaptive DT-assisted 3C management scheme can achieve a high QoE consistently.

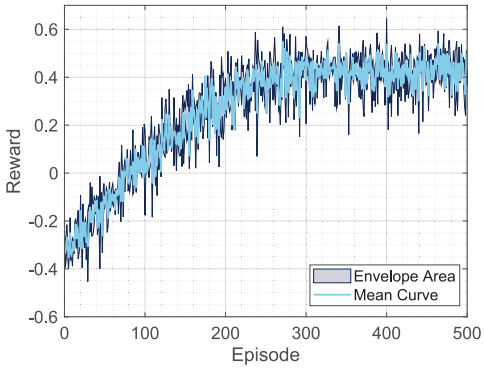


Fig. 8. Convergence performance.

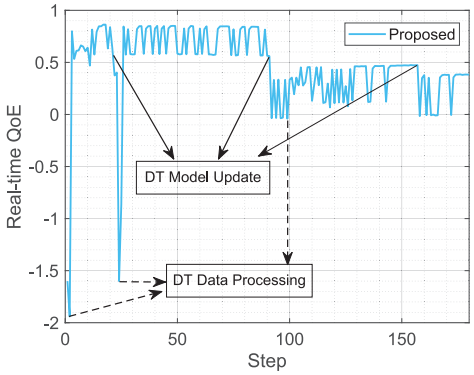


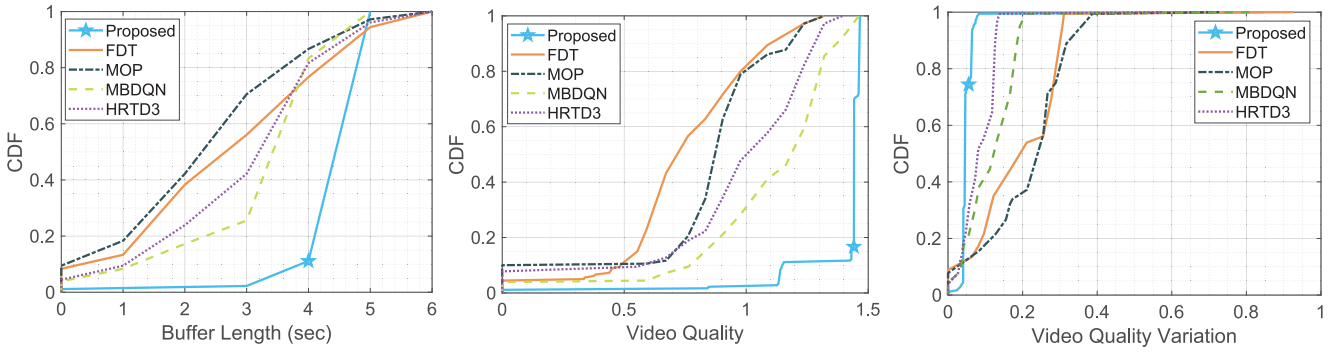
Fig. 9. Real-time QoE in each step.

C. The Influence of DT Data Processing and Model Update on QoE

To evaluate the influence of DTP and DTU on QoE performance, we present the real-time QoE of each step in one episode in Fig. 9. Initially, the QoE value remains low as the DTP conducts multicast grouping and updates the swipe probability distribution for video list recommendations, which pauses video transcoding and transmission until DTP completion. Around step 20, the QoE value decreases a lot because the current DT model cannot capture new swipe behavior patterns. Then, QoE slightly improves but is disrupted again around step 90 due to changes in swipe behaviors, necessitating another DTU for precise adjustments. However, QoE is still lower than earlier levels because increased MGs due to diverse user locations and behaviors strain the limited network resources, thus affecting video quality. Overall, the adaptive DT-assisted 3C management approach can effectively adapt to users' swipe behaviors to maintain high QoE.

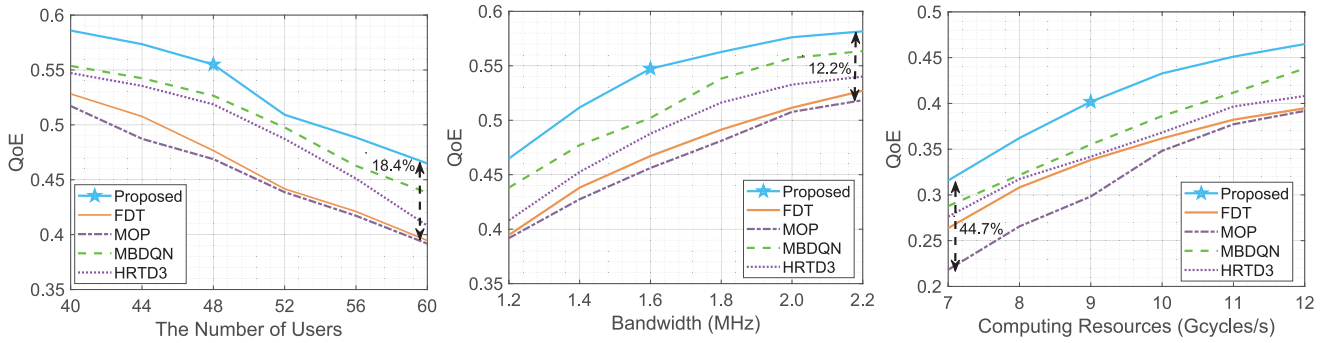
D. Performance Evaluation of QoE Components

In this subsection, we evaluate the QoE performance from buffer length, video quality, and video quality variation.



(a) Buffer length comparison (b) Video quality comparison (c) Video quality variation comparison

Fig. 10. The performance comparison of different QoE components.



(a) Fixed bandwidth 1.2 MHz and computing resources 12 Gcycles/s (b) Fixed user number 60 and computing resources 12 Gcycles/s (c) Fixed user number 60 and bandwidth 1.2 MHz

Fig. 11. The performance of QoE under different settings.

The cumulative distribution function (CDF) of buffer length for various schemes is shown in Fig. 10(a). The proposed scheme's buffer lengths mainly range from 4 to 5 sec, with probabilities of approximately from 10% to 90%, while other algorithms extend up to 6 sec but typically stay between 2 to 4 sec. The proposed algorithm adapts well to dynamic swipe behaviors and service delay, unlike other schemes which lead to more frequent buffering. The MOP scheme is least effective, primarily because it uses a fixed strategy for DTP and model updates that cannot match the dynamic swipe behaviors, resulting in poor user grouping and video recommendations. Moreover, MOP's focus on immediate performance gains causes the uneven segment buffering, resulting in the buffer length not concentrating at a higher stable level.

As shown in Fig. 10(b), we present the CDF of video quality across different schemes. The proposed algorithm consistently delivers video quality between 1.1 and 1.4, surpassing other methods. The FDT algorithm underperforms, generally yielding video quality between 0.6 to 1.1. This is due to its fixed DTP and model update strategies, which do not match user preferences, causing frequent swipes and necessitating quick buffering of lower-quality videos to decrease rebuffering times. In contrast, the proposed 3C management algorithm optimizes QoE holistically, ensuring longer buffer lengths and consistent video quality, thereby enhancing overall QoE.

As shown in Fig. 10(c), we present the CDF of video quality variation for different schemes. The proposed algorithm shows significantly lower fluctuations, mainly between 0.02 and 0.04, indicating a more consistent video quality for users. The HRTD3 and MBDQN schemes follow closely, using a hierarchical TD3 and a model-based BDQN algorithm respectively, to manage 3C strategies more effectively than the traditional MOP algorithm. The outstanding performance of our proposed scheme is due to enhanced action exploration in DRL, which adapts better to network and user dynamics.

TABLE IV
QoE Performance Under Different User Velocities

QoE	Proposed	FDT	MOP	MBDQN	HRTD3
Low velocities (2∼5 Km/h)	0.4649	0.3947	0.3918	0.4382	0.4081
High velocities (20∼40 Km/h)	0.4317	0.3631	0.3583	0.4022	0.3714

E. QoE Evaluation Under Different Network Settings

As shown in Fig. 11, we evaluate the QoE under different numbers of users, bandwidths, and computing resources. Fig. 11(a) illustrates the QoE performance with the increasing number of users and fixed network bandwidth at 1.2MHz and computing resources at 12 Gcycles/s. As the number of users increases from 40 to 60, all schemes exhibit a decreasing trend in QoE, indicating that higher user densities generally degrade service quality. However, the proposed solution shows a more gradual decline compared to the other schemes, indicating its relative robustness in handling user

density increases. Fig. 11(b) analyzes the effect of increasing bandwidth on QoE. The proposed scheme capitalizes on the additional bandwidth more effectively than other schemes. At the maximum bandwidth of 2.2 MHz, the proposed scheme exhibits a QoE that is 12.2% higher than that the MOP scheme. This is because the MOP scheme uses a myopic strategy that only allocates as many bandwidth resources to the user group that can obtain the best QoE in the current time slot. Fig. 11(c) explores how varying computing resources affect QoE. At the highest computing resource allocation of 12 Gcycles/s, the proposed scheme achieves a QoE that is 44.7% superior to that of the worst scheme. This dramatic improvement highlights the proposed solution's efficiency in leveraging additional computational power to enhance data processing and ultimately improve the overall user experience.

Table IV presents a comparative analysis of QoE under low and high user velocities (40 km/h speed limit in the campus area). In the low user dynamics scenario, the proposed scheme exhibits superior performance with a QoE of 0.4649, outperforming other schemes. Notably, the MBDQN scheme is the closest competitor with a QoE of 0.4382. In high user dynamics scenarios, the proposed scheme maintains its lead with a QoE of 0.4217, indicating its robustness and effectiveness even under more dynamic conditions. From the low to high user velocities, although the proposed scheme experiences performance degradation, its QoE performance is still higher than the baseline scheme (MOP) with 18.4% and 20.5%, because the proposed DT data processing and model update mechanism can well adapt to user dynamics.

VIII. Discussion

In the DT data collection module, we assume that only four kinds of user status data need to be collected, i.e., locations, channel conditions, swipe timestamps, and preferences, which cannot always accurately reflect users' diverse service requirements and sensitivity degrees on QoE factors. Therefore, the collected data attributes and frequency should be customized. In addition, the data collection frequency for each DT data attribute should be more flexible by comprehensively considering the DT prediction accuracy and user status dynamics.

IX. Conclusion

In this paper, we have proposed an adaptive DT-assisted 3C management scheme to maximize long-term QoE in MSVS. Specifically, we have designed an adaptive DTP and DTU mechanism to adapt to user behavior dynamics. Furthermore, we have established a fine-grained QoE model that considers the impact of resource constraints and DT model accuracy. A diffusion-based DRL algorithm has been proposed to determine the joint 3C management decision. The proposed adaptive DT-assisted 3C management scheme can address the challenges of customized resource management in scenarios with high user dynamics. For the future work, we will investigate an autonomous UDT generation and maintenance mechanism to reduce DT cost in MSVS.

References

[1] Y. Gao, X. Wei, and L. Zhou, "Personalized QoE improvement for networking video service," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 10, pp. 2311–2323, Oct. 2020.

[2] Z. Zhang, X. Wang, C. Sun, and Y. Zhang, "Cooperative transmission for multimedia video distribution: Models, algorithms and implementation," *IEEE Netw.*, early access, Jun. 10, 2024, doi: 10.1109/MNET.2024.3411645.

[3] Y. Abo Rahama, M. S. Hassan, and M. H. Ismail, "A stochastic-based rate control approach for video streaming over cognitive radio networks," *IEEE Trans. Cogn. Commun. Netw.*, vol. 5, no. 1, pp. 181–192, Mar. 2019.

[4] K. Zahoor, K. Bilal, A. Erbad, and A. Mohamed, "Service-less video multicast in 5G: Enablers and challenges," *IEEE Netw.*, vol. 34, no. 3, pp. 270–276, May/Jun. 2020.

[5] Z. Tao, W. Xu, Y. Huang, X. Wang, and X. You, "Wireless network digital twin for 6G: Generative AI as a key enabler," *IEEE Wireless Commun.*, vol. 31, no. 4, pp. 24–31, Aug. 2024.

[6] Y.-C. Wang, J. Xue, C. Wei, and C.-C. J. Kuo, "An overview on generative AI at scale with edge-cloud computing," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 2952–2971, 2023.

[7] E. Artioli, "Generative AI for HTTP adaptive streaming," in *Proc. ACM Multimedia Syst. Conf.*, Bari, Italy, 2024, pp. 516–519.

[8] X. Shen, J. Gao, W. Wu, M. Li, C. Zhou, and W. Zhuang, "Holistic network virtualization and pervasive network intelligence for 6G," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 1–30, 1st Quart., 2022.

[9] X. Huang, H. Yang, S. Hu, and X. Shen, "Digital twin-driven network architecture for video streaming," *IEEE Netw.*, vol. 38, no. 6, pp. 334–341, Nov. 2024.

[10] X. Huang, C. Zhou, W. Wu, M. Li, H. Wu, and X. Shen, "Personalized QoE enhancement for adaptive video streaming: A digital twin-assisted scheme," in *Proc. IEEE Global Commun. Conf.*, Rio de Janeiro, Brazil, 2022, pp. 4001–4006.

[11] E. Artioli, F. Tashtarian, and C. Timmerer, "DIGITWISE: Digital twin-based modeling of adaptive video streaming engagement," in *Proc. ACM Multimedia Syst. Conf.*, 2024, pp. 78–88.

[12] L. Qi, X. Xu, X. Wu, Q. Ni, Y. Yuan, and X. Zhang, "Digital-twin-enabled 6G mobile network video streaming using mobile crowdsourcing," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 10, pp. 3161–3174, Oct. 2023.

[13] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Stockholmsmässan, Stockholm, Sweden, Jul. 2018, pp. 1587–1596.

[14] G. Sun et al., "Generative AI for deep reinforcement learning: Framework, analysis, and use cases," 2024, *arXiv:2405.20568*.

[15] N. Chukhno, O. Chukhno, S. Pizzi, A. Molinaro, A. Iera, and G. Araniti, "Efficient management of multicast traffic in directional mmWave networks," *IEEE Trans. Broadcast.*, vol. 67, no. 3, pp. 593–605, Sep. 2021.

[16] A. De La Fuente, J. J. Escudero-Garzás, and A. García-Armada, "Radio resource allocation for multicast services based on multiple video layers," *IEEE Trans. Broadcast.*, vol. 64, no. 3, pp. 695–708, Sep. 2018.

[17] J. Li, Q. Pan, Z. Wu, P. Zhu, D. Wang, and X. You, "Spectral efficiency of unicast and multigroup multicast transmission in cell-free distributed massive MIMO systems," *IEEE Trans. Veh. Technol.*, vol. 71, no. 12, pp. 12826–12839, Dec. 2022.

[18] S. Zuhra, K.-L. Besser, P. Chaporkar, A. Karandikar, and H. V. Poor, "Optimal resource allocation for loss-tolerant multicast video streaming," *Entropy*, vol. 25, no. 7, p. 1045, 2023.

[19] M. N. Dani, D. K. So, J. Tang, and Z. Ding, "Resource allocation for layered multicast video streaming in NOMA systems," *IEEE Trans. Veh. Technol.*, vol. 71, no. 11, pp. 11379–11394, Nov. 2022.

[20] X. Huang, L. He, L. Wang, and F. Li, "Towards 5G: Joint optimization of video segment caching, transcoding and resource allocation for adaptive video streaming in a multi-access edge computing network," *IEEE Trans. Veh. Technol.*, vol. 70, no. 10, pp. 10909–10924, Oct. 2021.

[21] S. Chen, B. Yang, J. Yang, and L. Hanzo, "Dynamic resource allocation for scalable video multirate multicast over wireless networks," *IEEE Trans. Vehi. Technol.*, vol. 69, no. 9, pp. 10227–10241, Sep. 2020.

[22] R. Zhang et al., "Buffer-aware virtual reality video streaming with personalized and private viewport prediction," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 2, pp. 694–709, Feb. 2022.

[23] T. Zhao, F. Li, and L. He, "Secure video offloading in MEC-enabled IIoT networks: A multicell federated deep reinforcement learning approach," *IEEE Trans. Ind. Informat.*, vol. 20, no. 2, pp. 1618–1629, Feb. 2024.
[24] H. Du et al., "The age of generative AI and AI-generated everything," *IEEE Netw.*, vol. 38, no. 6, pp. 501–512, Nov. 2024.
[25] J. Wang et al., "A unified framework for guiding generative AI with wireless perception in resource constrained mobile edge networks," *IEEE Trans. Mobile Comput.*, vol. 23, no. 11, pp. 10344–10360, Nov. 2024.
[26] E. Glaessgen and D. Stargel, "The digital twin paradigm for future NASA and U.S. air force vehicles," in *Proc. Struct. Dyn. Mater. Conf.*, Honolulu, HI, USA, 2012, p. 1818.
[27] S. Liu, W. Wu, S. Li, T. H. Luan, and X. Shen, "Digital twin-assisted adaptive preloading for short video streaming," in *Proc. IEEE Int. Commun. Conf. (ICC)*, Denver, CO, USA, 2024, pp. 1–6.
[28] Y. Gao, J. Liao, X. Wei, and L. Zhou, "Quality-aware massive content delivery in digital twin-enabled edge networks," *China Commun.*, vol. 20, no. 2, pp. 1–13, Feb. 2023.
[29] X. Huang, W. Wu, S. Hu, M. Li, C. Zhou, and X. Shen, "Digital twin based user-centric resource management for multicast short video streaming," *IEEE J. Sel. Topics Signal Process.*, vol. 18, no. 1, pp. 50–65, Jan. 2024.
[30] X. Liu, M. Masana, L. Herranz, J. Van de Weijer, A. M. Lopez, and A. D. Bagdanov, "Rotate your networks: Better weight consolidation and less catastrophic forgetting," in *Proc. Int. Conf. Pattern Recognit. (ICPR)*, Beijing, China, Aug. 2018, pp. 2262–2268.
[31] F. M. Castro, M. J. Marín-Jiménez, N. Guil, C. Schmid, and K. Alahari, "End-to-end incremental learning," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Munich, Germany, Sep. 2018, pp. 233–248.
[32] J. He, R. Mao, Z. Shao, and F. Zhu, "Incremental learning in online scenario," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn. (CVPR)*, Jun. 2020, pp. 13926–13935.
[33] P. M. Radiuk, "Impact of training set batch size on the performance of convolutional neural networks for diverse datasets," *Inf. Technol. Manage. Sci.*, vol. 20, no. 1, pp. 20–24, Dec. 2017.
[34] M. Wien, H. Schwarz, and T. Oelbaum, "Performance analysis of SVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1194–1203, Sep. 2007.
[35] A. Z. Yalcin, M. Yuksel, and B. Clerckx, "Rate splitting for multi-group multicasting with a common message," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 12281–12285, Oct. 2020.
[36] G. Huang, W. Gong, B. Zhang, C. Li, and C. Li, "An online buffer-aware resource allocation algorithm for multiuser mobile video streaming," *IEEE Trans. Vehi. Technol.*, vol. 69, no. 3, pp. 3357–3369, Mar. 2020.
[37] C. Zhou, Y. Ban, Y. Zhao, L. Guo, and B. Yu, "PDAS: Probability-driven adaptive streaming for short video," in *Proc. ACM Int. Conf. Multimedia (ACM MM)*, Lisboa, Portugal, Oct. 2022, pp. 7021–7025.
[38] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 33, Dec. 2020, pp. 6840–6851.
[39] H. Du et al., "Diffusion-based reinforcement learning for edge-enabled AI-generated content services," *IEEE Trans. Mobile Comput.*, vol. 23, no. 9, pp. 8902–8918, Sep. 2024.
[40] X. Huang, S. Hu, H. Yang, X. Wang, Y. Pei, and X. Shen, "Digital twin-based network management for better QoE in multicast short video streaming," *IEEE Trans. Wireless Commun.*, vol. 23, no. 11, pp. 16187–16202, Nov. 2024.
[41] S. Boyd and J. Mattingley, *Branch and Bound Methods: Notes for EE364b*, Stanford Univ., Stanford, CA, USA, 2007.

Xinyu Huang (Graduate Student Member, IEEE) received the B.E. degree in Qian Xuesen experimental class from Xidian University, Xi'an, China, in 2018, and the M.S. degree in information and communications engineering from Xi'an Jiaotong University, Xi'an, in 2021. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Waterloo, Waterloo, ON, Canada. His research interests include digital twins, multimedia communication, and network resource management.

Xue Qin (Graduate Student Member, IEEE) received the B.Eng. degree from North University of China in 2008 and the M.A.Sc. degree from the University of Windsor, Windsor, ON, Canada, in 2022. She is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Waterloo, Waterloo, ON, Canada. Her research interests include mobile edge computing, AI, and network security.

Mushu Li (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2021. She is an Assistant Professor with the Department of Computer Science, Lehigh University, Bethlehem, PA, USA. She was a Postdoctoral Fellow with Toronto Metropolitan University, Toronto, ON, Canada, from 2022 to 2024 and the University of Waterloo from 2021 to 2022. Her research interests include mobile edge computing, the system optimization in wireless networks, and machine learning-assisted network management. She was the recipient of the Natural Science and Engineering Research Council of Canada Postdoctoral Fellowship in 2022, the NSERC Canada Graduate Scholarship in 2018, and the Ontario Graduate Scholarship in 2015 and 2016, respectively.

Cheng Huang (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Waterloo, ON, Canada, in 2020. He is currently an Associate Professor with the School of Computer Science, Fudan University. Before joining Fudan University, he was a Research Fellow with the Department of Electrical and Computer Engineering, University of Waterloo from 2020 to 2023. He has published over 70 papers in prestigious journals and conferences, including the IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, and the IEEE TRANSACTIONS ON MOBILE COMPUTING. His research interests lie in the areas of data privacy and AI security. He has received the Best Paper Awards from ICCC '15, ICC '18, GLOBECOM '22, and ICCC '23. He serves as an Associate Editor for IEEE INTERNET OF THINGS JOURNAL and *Peer-to-Peer Networking and Applications* (Springer), and the Symposium Co-Chair of IEEE GLOBECOM '24 CISS. He has served as the Publicity Chair of ICA3PP '22, PST '23, and SustainCom '23, and the TPC Member of many international conferences.

Xuemin (Sherman) Shen (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990. He is an University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research focuses on network resource management, wireless network security, Internet of Things, AI for networks, and vehicular networks. He is a Registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, a Chinese Academy of Engineering Foreign Member, an International Fellow of the Engineering Academy of Japan, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society.