Edge Computing Empowered Video Content Delivery Strategy in the Internet of Vehicles

Zhenyu Feng*, Jiajun Ye*, Chenyuan Feng†
* Ministry of Industry and Information Technology (MIIT) Fifth Electronics Research Institute,
Guangzhou-GWS Environmental Equipment Co., Ltd, Guangzhou, China
† Department of Communication Systems, Eurecom, France
Emails: fungzain@gmail.com, yejj2013@foxmail.com, chenyuan.feng@eurecom.fr

***Abstract*—With the advancements of mobile communication and autonomous driving technologies, user demand for mobile multimedia services in the Internet of Vehicles (IoV) has been gradually increasing, which has put a great deal of strain on communication and computing. The high mobility of vehicles and the complexity of network scenarios present significant hurdles to the improvement of the quality of in-car entertainment services, even while streaming media transmission technology helps to reduce transmission latency. This paper investigates a video content distribution strategy that combines edge leaning with edge caching, given that video data makes up the great bulk of makes up the great bulk of currently available entertainment information. Specifically, an adaptive bitrate video distribution technique is proposed for the multi-node video stream data transmission problem, which is carefully formulated as Markov decision process (MDP). Our suggested approach optimizes both the selection of video contents with varying compression bitrates and the allocation of vehicle-to-everything (V2X) communication sub-channels based on the communication environment, users' locations and buffering states, and each network node's cache information. The simulation results confirm that our suggested approach is beneficial in terms of cache hit ratio and quality of service (QoS).**

***Index Terms*—Internet of Vehicles, video delivery, Markov decision process, deep reinforcement learning, edge intelligence**

I. Introduction

The popularity of in-vehicle touch screens and the rapid advancement of intelligent driving technology have driven the demand for mobile multimedia services in the Internet of Vehicles (IoV) [1]. In 2023, global mobile data traffic is expected to reach 49 exabytes, with 82% of that bandwidth being video data, according to a report by Cisco [2]. Consequently, the most crucial aspect of data distribution will be—or already is—the transmission of video streaming data. Mobile communication networks are facing significant challenges due to the rapid expansion of video data delivery services. Therefore, supplying communication, storage, and processing power to network edge nodes aids in lowering transmission latency, backhaul link pressure, and enhancing quality of service (QoS) [3].

Though content delivery has been extensively studied [4], the user experience of watching video while driving is influenced by a variety of circumstances, and video data transmission differs significantly from file transfer [5]. The IoV presents significant hurdles for video data delivery because of the high mobility of vehicle users (VUs) and the intermittency nature of information transmission [6], [7]. To solve this problem, deep reinforcement learning (DRL) has been introduced in recent research to achieve edge intelligence empowered video delivery. Essentially, interactions with dynamic environments allow the central server and edge servers to become intelligent decision makers by learning better strategies for transmission resource allocation and video bitrate selection [8], [9]. In [8], the asynchronous advantage actor-critic algorithm is applied to wireless network resource allocation and video compression bitrate selection simultaneously. In [9], a deterministic strategy gradient learning technique is applied to optimize bandwidth allocation, content acquisition, and service node scheduling simultaneously. However, because of the highly mobile cars and time-varying content requests, it is an extremely hard task to determine the best course of action for video content delivery in the IoV.

To enhance the QoS for on-demand video streaming while reducing the computational complexity, we propose a novel adaptive bitrate video streaming delivery mechanism with V2X communication and video transcoding techniques. The main contributions in this paper are summarized as follows: Firstly, we formulate the transmission scheduling problem as Markov decision process (MDP), carefully accounting for the locations of all VUs, their video-on-demand requests, and the caching status and communication capabilities of all network nodes; Secondly, we propose an algorithm for adaptive video content distribution to enhance the viewing experience of videos for those in moving vehicles; Additionally, we design a distributed training distributed execution (DTDE) manner to further reduce the computational complexity; Finally, we use SUMO simulator to simulate real-world road configuration and traffic data, and provide comprehensive experiments to verify our suggested framework and methodology.

II. System Model

Due to the high mobility of vehicles, the IoV features a complicated and dynamic network topology structure. As such, real-time video content distribution and user quality of experience (QoE) enhancement based on ambient feedback information is highly complex. This paper considers business scenarios related to video-on-demand and urban traffic areas.

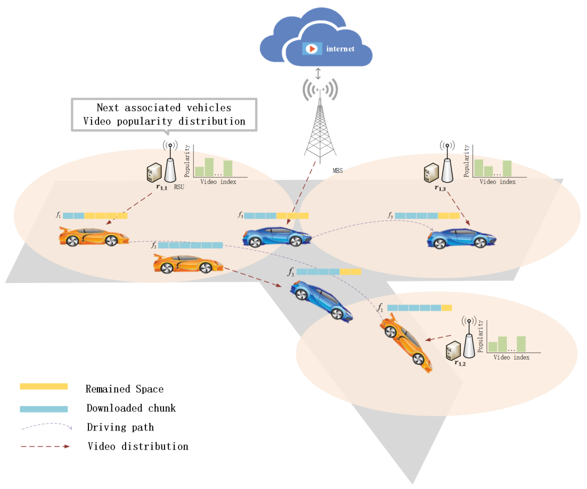


Fig. 1. System model of video delivery in the IoV.

A. Network Model

As shown in fig:system model, the video delivery network consists of one video source server, one macro base station (MBS), R roadside units (RSUs) and U mobile VUs. Let $\mathbb{N}$ denote the set of all network nodes, $\mathbb{B} = \{b_0\}$, $\mathbb{R} = \{r_1, \ldots, r_R\}$, and $\mathbb{U} = \{u_1, \ldots, u_U\}$ denote the index sets of the MBS, RSUs and VUs, respectively, and we have $\mathbb{N} = \mathbb{B} \cup \mathbb{R} \cup \mathbb{U}$. The MBS is connected to the baseband unit (BBU) pool through the backhaul links, and all RSUs are connected to the BBU pool through the fronthaul links. The RSU not only has the communication capability, but also has the capability of edge cache, computing and resource management.

For the sake of this work, we assume that VUs can request video content while in motion and share the content they receive with other users in the vehicles' coverage area. Thus, in this study, VUs who share received contents are included in the distribution nodes together with all RSUs and the MBS. For the purpose of transcoding and caching video streaming data, each distribution node is equipped with computing and caching resources. VUs can employ vehicle-to-network (V2N), vehicle-to-infrastructure (V2I), and vehicle-to-vehicle (V2V) links, respectively, to obtain video streaming data with the MBS, the closest RSU, and nearby users.

Taking into account that the duration of the cache content update is significantly longer than the duration of the customers' video-on-demand requests, this study employs a double timescale model, akin to [9], [10], to optimize the delivery decision depending on buffer circumstances. There are T brief time slots for every long time slot. Each RSU will refresh the data in its cache area at the beginning of each long time slot. The video delivery system will assign the communication resource and choose the appropriate compression bitrate for the requested video content for the network at the start of each brief time slot.

B. Video Transcoding Technology

This effort uses video transcoding technologies to increase the video content delivery efficiency. To fulfill the requirements of both the receiving terminal and the communication environment, video content can be converted using this technology to a new version with a different compression bitrate [6]. Let $\mathbb{F} = \{f_1, \ldots, f_F\}$ denote the index set of all available video files. Each video file is divided into multiple chunks with an equal playing duration τ, and the chunk index set of the f-th video file is defined as $\mathbb{X}_f = \{1, \ldots, x_f, \ldots, X_f\}$. Let $\mathbb{I} = \{1, \ldots, i, \ldots, I\}$ denote the set of I versions of video bitrate, $b(i)$ denote the video bitrate of the i-th version. The smaller the value of i, the higher the bitrate version, the higher the video resolution, and the larger the video file size.

1) Video Transcoding Delay: In this work, all of the user-required video files are stored on the video source server. For storage and transmission, every video file is split up into several chunks, and each chunk has I versions with various compression bitrates. Users within the MBS's service area could receive different bitrate versions of all video files via V2N links because the MBS has access to the remote video source server over backhaul networks. An edge server and cache storage capable of caching a restricted quantity of video chunks in the highest bitrate versions are installed in each RSU. Before sending the cached video chunks to the intended user, each RSU can employ video transcoding technologies to convert them to the proper bitrate version [11]. The quantity of brief time slots of transcoding delay can be expressed as follows:

$$t^{TC}(i^c_{u,f,x}, i^t_{u,f,x}) = \left\lceil \frac{(b(i^t_{u,f,x}) - b(i^c_{u,f,x}))\tau}{\xi\delta\epsilon} \right\rceil, \quad (1)$$

where $\lceil\cdot\rceil$ is a ceiling function, ξ denotes the amount of data that can be be handled in a CPU cycle, δ denotes the CPU frequency, $i^c_{u,f,x}(t)$ and $i^t_{u,f,x}(t)$ denote the indexes of the cached and the target bitrate versions for transmission, respectively, $b(i^c_{u,f,x})$ and $b(i^t_{u,f,x})$ denote the bitrates of the corresponding $i^c_{u,f,x}$ and $i^t_{u,f,x}$ variants, respectively, τ denotes the playing duration of each chunk, ϵ denotes the duration of each brief time slot. In this study, $\epsilon \ll \tau$, the playing duration of each video chunk, is substantially longer than the duration of each brief time slot.

2) Video Playing Duration: The user keeps receiving new video chunks while playing the video that has been cached in the buffer at a steady pace after receiving one entire video chunk. Assume that the video chunk x_f begins to be transmitted at the t-th time slot, the video lag duration experienced by user u can be expressed as:

$$T^B_{u,f,x}(t) = \sum_{k=t^s_{u,f,x}}^{t^p_{u,f,x}-1} \max\{\epsilon - \Gamma_u(k), 0\}, \quad (2)$$

where $\Gamma_u(k)$ denotes the total amount of time that all cached video chunks in user u's buffer can be played at the beginning of the k-th brief time slot, $t^s_{u,f,x}$ and $t^p_{u,f,x}$ denote the indexes of brief time slot of receiving the request of video chunk x_f

from user u and that of finishing the transmission of video chunk x_f to user u. The expression of $t^p_{u,f,x}$ is as follows:

$$t^p_{u,f,x} = t^s_{u,f,x} + \left\lceil \frac{T^p_{u,f,x}(t)}{\epsilon} \right\rceil, \tag{3}$$

where $T^p_{u,f,x}(t)$ denotes the total amount of time that user u spent receiving video chunk x_f. Considering link switching, a video chunk will be destroyed rather than being cached in the users' buffer if it is not fully sent when the user exits the distribution node's communication range. Since the video chunk is only available in the player buffer if it has been completely received by user u, the increment of video playing duration cached in user u's player buffer can be expressed as follows:

$$\Delta\Gamma_u(t) = \begin{cases} \tau, & t = t^p_{u,f,x}, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

The change of playing duration of all cached video chunks in user u's buffer can be regarded as the evolution of a queue, which can be expressed as follows:

$$\Gamma_u(t+1) = \Delta\Gamma_u(t) + \max\left\{\Gamma_u(t) - \epsilon, 0\right\}. \tag{5}$$

C. Communication Model

In this work, the V2N, V2I, and V2V communication links are assigned distinct frequency bands, and the technique of orthogonal frequency division multiplexing (OFDM) is used to prevent interference between the various sub-channels. Assume that the coverage of each RSU does not overlap, several RSUs may employ the same communication spectrum. The signal-to-noise ratio (SNR) that user u received from distribution node n can be expressed as follows:

$$SNR_{n,u}(t) = \frac{p_n g_{n,u}(t)}{\sigma^2_{n,u}(t)}, \quad n \in \mathbb{N}, \tag{6}$$

where p_n denotes the transmitting power of distribution node n, $g_{n,u}(t)$ and $\sigma^2_{n,u}(t)$ denote the channel gain and the power spectrum density of additive white Gaussian noise (AWGN) from nodes n to user u, respectively. The data rate of fetching video chunk from distribution node n to user u is given by:

$$r_{n,u}(t) = B_{n,u}\log_2\left(1 + SNR_{n,u}(t)\right), \quad n \in \mathbb{N}, \tag{7}$$

where $B_{n,u}$ denotes the bandwidth of the sub-channel between user u and node n.

D. Delivery Model

At the beginning of each brief time slot, the video delivery system determines the distribution node and bitrate version of the requested video chunks according to the target user's playing buffer status, the communication environment, and the contents of all MBS, RSU, and VUs' caches. There are three kinds of distribution nodes, namely, the MBS, the RSU nearest to the target user, and the users in the target user's communication range who has also cached the requested video chunks. Channel allocation comes in four flavors: no link, V2N link, V2I link, and V2V link. Let $\phi_{u,f,x}(t)$ denote the indicator of transmission links allocated for transmitting video chunk x_f to user u at the t-th brief time slot. Let $\varphi_{n,u}(t)$ denote the indicator of whether the distribution node n responds to user u at the t-th brief time slot. The details of different transmission decisions are as follows.

- **No Responding:** If no link is allocated for transmitting video chunk x_f to user u at the t-th brief time slot, we have $\phi_{u,f,x}(t) = 0$ or $\varphi_{n,u}(t) = 0, \forall n \in \mathbb{N}$.
- **Responding by the MBS:** If the V2N link is allocated, we have $\phi_{u,f,x}(t) = 1$ and $\varphi_{b,u}(t) = 1, b \in \mathbb{B}$. The transmission delay of user u receiving the video chunk x_f via V2N link can be expressed as follows:

$$T^{p,V2N}_{u,f,x}(t) = \frac{b(i^t_{u,f,x})\tau}{r_{b,u}(t)} + \frac{b(i^t_{u,f,x})\tau}{r_{bh}}, \tag{8}$$

  where $b(i^t_{u,f,x})$ denotes the target bitrate, r_{bh} denotes the transmission data rate of the backhual link between the MBS and the remote video source server, $r_{b,u}(t)$ denotes the transmission data rate of the V2N link between the MBS and user u.
- **Responding by the RSUs in the edge layer:** If the requested video chunk is cached in the edge layer and the V2I link is allocated, we have $\phi_{u,f,x}(t) = 2$ and $\varphi_{r,u}(t) = 1, r \in \mathbb{R}$. The nearest RSU within the V2I communication range is referred to as the local RSU for a target vehicle. Let $c_{r,f,x}(t)$ denote the indicator of whether the video chunk x_f is cached at RSU r at the t-th time slot. If it is cached, we have $c_{r,f,x}(t) = 1$, otherwise, $c_{r,f,x}(t) = 0$. The local RSU r will distribute the requested video chunk to the target user directly if it has already cached the requested video chunk and the target bitrate equals to that of cached version; if not, the local RSU must convert the cached bitrate version to the target bitrate version before transmission. Thus, if $c_{r,f,x}(t) = 1$, the transmission delay can be expressed as:

$$T^{p,V2V,DT}_{u,f,x}(t) = \frac{b(i^t_{u,f,x})\tau}{r_{r,u}(t)} + t^{TC}(i^r_{u,f,x}, i^t_{u,f,x}), \tag{9}$$

  where $r_{r,u}(t)$ denotes the transmission data rate of the V2I link between RSU r and user u, $t^{TC}(i^r_{u,f,x}, i^t_{u,f,x})$ denotes the transcoding delay needed to convert the bitrate version that RSU r caches to the desired bitrate version. The local RSU r should prefetch this chunk from the r'-th RSU in the BBU pool over the front-haul link if the requested chunk is cached by another RSU r'. This will result in a transmission latency as follows:

$$T^{p,V2V,TC}_{u,f,x}(t) = \frac{b(i^t_{u,f,x})\tau}{r_{r,u}(t)} + t^{TC}(i^{r'}_{u,f,x}, i^t_{u,f,x}) + \frac{b(i^t_{u,f,x})\tau}{r_{fh}}, \tag{10}$$

  where r_{fh} is the fronthual link transmission data rate.
- **Responding by the content sharing VUs:** Let $\mathcal{N}_u$ denotes the set of content sharing VUs that are located within the target user u's V2V communication range and that store the requested chunk at a bitrate that is equal to or greater than the target one. If the V2V link is allocated, we have $\phi_{u,f,x}(t) = 3$ and $\varphi_{v,u}(t) = 1, \forall v \in \mathcal{N}_u$. Similar to the RSU nodes, user v has the option to

transcode before transmitting or send straight. Thus, the transmission delay can be expressed as follows:

$$T_{u,f,x}^{p,V2V}(t) = \frac{b(i_{u,f,x}^t)\tau}{r_{v,u}(t)} + t^{TC}(i_{u,f,x}^v, i_{u,f,x}^t), \tag{11}$$

where $r_{v,u}(t)$ denotes the transmission data rate of the V2V link between user v and user u, $t^{TC}(i_{u,f,x}^v, i_{u,f,x}^t)$ denotes the transcoding delay of converting the bitrate version cached by user v to the target bitrate version, $i_{u,f,x}^t$ denotes the bitrate version cached by user v.

E. Problem Formulation

This study selects three essential indicators—video bitrate, video lag duration and video bitrate switching amplitude—to assess users' QoE. Among them, the amplitude difference between the bitrates of two successive video chunks is referred to as the video bitrate switching amplitude. Specifically, for the first received video chunk, the video bitrate switching amplitude value is zero. The QoE metric can be defined as:

$$\begin{aligned} QoE_{u,f,x}(t) = & \xi_1 \frac{b(i_{u,f,x}^t)}{b(1)} - \xi_2 \frac{T_{u,f,x}^B(t)}{\tau} \\ & - \xi_3 \frac{\left| b(i_{u,f,x}^t) - b(i_{u,f,x-1}^t) \right|}{b(1) - b(I)}, \end{aligned} \tag{12}$$

where ξ_1, ξ_2 and ξ_3 are weight coefficients of video bitrate, lag duration, and bitrate switching amplitude, respectively. This QoE metric is widely used in classical Model Prediction Control (MPC) algorithm [12] and adaptive bitrate Pensieve algorithm [13].

Enhancing the typical user's QoE is the aim of our video delivery strategy. Thus, the objective function is defined as:

$$\begin{aligned} \max_{i_{u,f,x}^t, \phi_{u,f,x}(t), \varphi_{n,u}(t)} & \frac{1}{U} \sum_{t=1}^{T} \sum_{u \in \mathbb{U}} \sum_{f \in \mathbb{F}} \sum_{x \in \mathbb{X}_f} QoE_{u,f,x}(t) \\ s.t. & \sum_{u \in \mathcal{U}} \varphi_{b,u}(t) B_{b,u} \le B_M, \\ & \sum_{u \in \mathcal{U}} \varphi_{r,u}(t) B_{r,u} \le B_R, \forall r, \\ & \sum_{v \in \mathcal{N}_u} \varphi_{v,u}(t) B_{v,u} \le B_U, \forall u, \\ & i_{u,f,x}(t) \in \{1, \ldots, I\}, \\ & \phi_{u,f,x}(t) \in \{0, 1, 2, 3\}, \\ & \varphi_{n,u}(t) \in \{0, 1\}. \end{aligned} \tag{13}$$

where B_M, B_R and B_U denote the entire bandwidths of V2N, V2I and V2V communications, respectively.

III. DRL-based Adaptive Delivery Algorithm

To solve (13), we formulate the optimization problem as Markov Decision Process (MDP) and propose an adaptive video transmission algorithm based on DRL. Through a series of trials, the agent can use the DRL-based algorithm to optimize the strategy for choosing the video bitrate and communication link, making it a workable solution to the video streaming transmission issue. A fundamental algorithm for reinforcement learning, the DQN algorithm chooses actions by making updates to the Q-table. Nevertheless, the DQN algorithm cannot be used to solve the multi-node video content distribution issue in (13) due to the excessively huge dimensions of the action space and state space. We therefore use distributed learning to create a unique DRL method.

A. MDP Problem Formulation

The MBS is treated as an agent, and the details of quintuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, Q)$ of this MDP problem are defined as follows.

- **State Space $\mathcal{S}$:** At time slot t, the state space is defined as $\mathcal{S}(t) = \{C(t), B(t), H(t), \Gamma(t), T(t)\}$, where $C(t)$ denotes the caching information of video chunks stored in all RSUs and VUs, $B(t)$ and $H(t)$ denote the usage status and channel gains of all sub-channels, respectively, $\Gamma(t)$ denotes the buffer status of users, $T(t)$ denotes the residence time status of users maintaining connection with the local RSU and surrounding content sharing VUs.
- **Action Space $\mathcal{A}$:** The action space is defined as $\mathcal{A}(t) = \{I(t), \Phi(t)\}$, where $I(t)$ denotes all possible video bitrate versions, $\Phi(t)$ all possible sub-channel assignments.
- **State Transition Probability $\mathcal{P}$:** After selecting the actions $\mathcal{A}(t)$, the probability of turning to status $\mathcal{S}(t+1)$ from state $\mathcal{S}(t)$ is defined as $\mathcal{P}(\mathcal{S}(t+1) \mid \mathcal{S}(t), \mathcal{A}(t))$.
- **Reward Function $\mathcal{R}$:** The agent will receive an immediate reward after performing action $\mathcal{A}(t)$, which is defined as $\mathcal{R}(\mathcal{S}(t), \mathcal{A}(t)) = \sum_{u \in \mathbb{U}} \sum_{f \in \mathbb{F}} \sum_{x \in \mathbb{X}_U} QoE_{u,f,x}(t)$. In order to maximize a long-term reward, the agent seeks to determine the best course of action. The T-step cumulative discounted reward function is defined as $\mathcal{G}(t) = \mathbb{E}\{\sum_{k=0}^{\infty} \gamma^k \mathcal{R}(\mathcal{S}(t+k), \mathcal{A}(t+k))\}$,where $\gamma \in [0, 1]$ is the discounted factor.
- **Value Function and Q table:** the state-action value function is defined as $Q_\pi(\mathcal{S}(t), \mathcal{A}(t)) = \mathbb{E}\{\sum_{k=0}^{\infty} \gamma^k \mathcal{R}(\mathcal{S}(t+k), \mathcal{A}(t+k)) | \mathcal{S}(t), \mathcal{A}(t)\}$, which is the expectation of accumulated reward of adopting the policy π from the state $\mathcal{S}(t)$. The optimal policy can be obtained by minimizing the value function, which can be expressed as follows: $\pi^* = \underset{\pi \in \Pi}{\arg\max} Q_\pi(\mathcal{S}(t), \mathcal{A}(t))$, where Π denotes the set of all policies.

B. DRL-based Algorithm Design

Using deep neural networks, the Actor-Critic (AC) model is a novel technique for estimating value functions and strategies [8]. A large nonlinear policy optimization technique based on AC architecture is called Trust Region Policy Optimization (TRPO) [14]. The TRPO method guarantees the stability of policy development by reducing the Kullback-Leiblere (KL) divergence between the past and current policies, hence preventing destructive large-scale policy modifications. While the TRPO-based method can provide successful rules, its extreme computational complexity prevents it from functioning well in time-sensitive scenarios.

To further reduce the computational complexity, we train a DRL-based algorithm in a Distributed Training Distributed

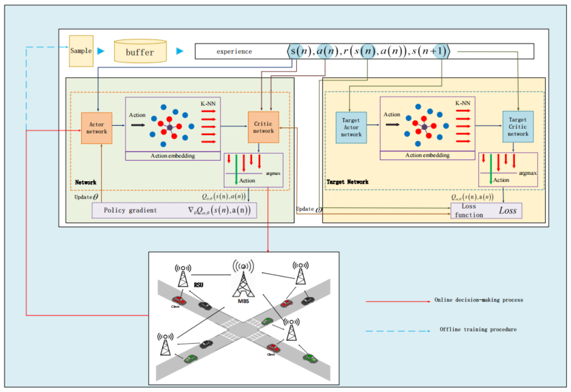


Fig. 2. Illustration of adaptive video delivery strategy in a DTDE manner.

execution (DTDE) manner. In our proposed strategy, the agent will go through the following two phases: the offline training phase and online decision-making phase. As exemplified in fig:dtde, during the offline training phase, the agent uses a number of sample data from the experience pool to calculate the value function; subsequently, it uses an optimizer, like ADAM, to update the learning strategies' parameters; finally, after a predefined period of time, the agent copies the parameters of the main network to the target network. Based on the knowledge gained from the offline training phase, the agent will choose the course of action in the online decision-making phase based on the environment's present observable status. Specifically, the main steps of our proposed DRL algorithm can be described as follows.

- **Initialization:** As shown in Fig. 2, the target network is in charge of assessing the reward function, whereas the main neural network chooses the action. A uniform distribution function that has been predefined can be used to create the initialization main neural network parameters.
- **Model Training:** During each training, the agent will select K experience quadruples from the experience pool to optimize the policy by minimizing the training loss function. Let θ and $\mathbf{w}$ denote the model parameters of the main network and the target network, respectively, The main network will calculate the policy gradient to update θ. The training loss function can be expressed as:

$$L(\mathbf{w}) = \frac{1}{2K}\sum_{n=1}^{K}\Bigg(\mathcal{R}(\mathcal{S}(t),\mathcal{A}(t)) + \gamma Q_\theta\Bigg(\mathcal{S}(t+1), \max_{\mathcal{A}(t+1)} Q_{\mathbf{w}}(\mathcal{S}(t+1),\mathcal{A}(t+1))\Bigg) - Q_{\mathbf{w}}(\mathcal{S}(t),\mathcal{A}(t))\Bigg)^2, \tag{14}$$

- **Parameter Update in Experience Pool:** The quadruple $< s(n), a(n), r(s(n), a(n), s(n+1) >$ generated during each encounter will be saved in the experience pool.

IV. EXPERIMENT AND DISCUSSION

A. Experimental Setting

In this work, we consider one 1 MBS, 7 RSUs and 20 users in an urban area. The real-world map in Shenzhen city used in this work has a size of roughly 1 km by 1 km, with (22.52N, 113.94W) as its center point. We use the SUMO simulator to generate the starting position and the moving trajectory of the vehicles over this area. We model user requests for various movies using the MovieLens 1M dataset [1]. This dataset has 3952 movies in total. And we assume each movie can be split up into 100 segments; an RSU can hold up to 6000 video chunks, whereas a user can only save up to 30 video chunks. The AWGN's power spectral density is set as -174 dBm/Hz. The transmitting powers of the MBS, RSUs, and users are set as 35dBm, 22dBm and 20dBm, respectively. The overall bandwidths and corresponding sub-channel bandwidths for V2N, V2I, and V2V communications are set as 300MHz, 200MHz, 50MHz, 30MHz, 20MHz, and 10MHz, respectively. Furthermore, the maximum communication distances for V2N, V2I and V2V communications are set as 1000m, 300m and 100m, respectively. Furthermore, the maximum communication distances for V2N, V2I and V2V communications are set as 1000m, 300m and 100m, respectively. All the experiments were conducted on an Ubuntu 20.04.1 server with an A100 GPU and the PyTorch platform.

TABLE I
IMPACT OF AVERAGE VEHICLE SPEED ON RESPONSE RATIO OF VARIOUS COMMUNICATION MODES

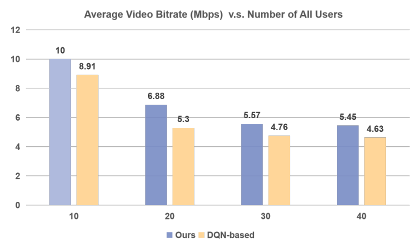
Strategy	Responder	0 km/h	30km/h	60km/h	90km/h
Ours	None	**11.02%**	**9.83%**	**10.27%**	**11.27%**
	MBS	38.45%	39.06%	42.29%	49.89%
	RSUs	42.01%	43.04%	40.08%	34.68%
	VUs	8.47%	8.07%	7.36%	4.19%
DQN	None	11.06%	10.98%	11.33%	11.60%
	MBS	45.10%	47.63%	48.77%	51.75%
	RSUs	38.62%	34.60%	33.51%	33.01%
	VUs	5.26%	6.79%	6.39%	3.64%

B. Performance Comparisons

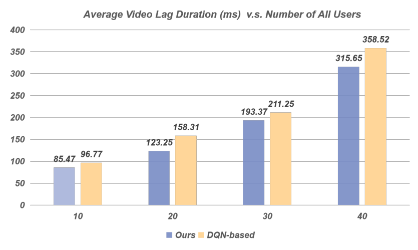
1) Impact of User Number: Fig. 3 depicts the impact of user number on different performance metrics. First of all, our proposed scheme outperforms conventional DQN-based algorithms in all Settings. Second, when the user number rises, the average video bitrate that each user receives drops. This makes sense because the entire bandwidth of V2X communications is limited. When there are only 10 users in the system, the resources are sufficient for all users to receive high-quality video chunks and experience short video lag latency and low bitrate version switching amplitude. Transmission resources become scarce as user numbers rise, leading to a sharp decline in all performance metrics.

2) Impact of User Mobility: Fig. 4 shows the impact of average vehicle speed on various performance metrics. First, it is evident that while the video bitrate switching amplitude and duration grow, the average bitrate of both techniques falls as average vehicle speed rises. This is due to the fact that the connection time of V2I and V2V links will be shortened as the vehicle speed increases. The residence duration of users

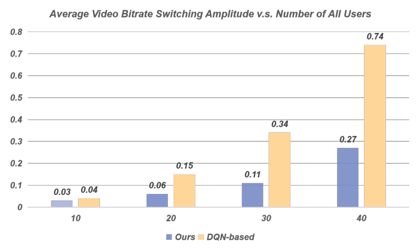
[1]https://grouplens.org/datasets/movielens/1M/



(a) Average video bitrate

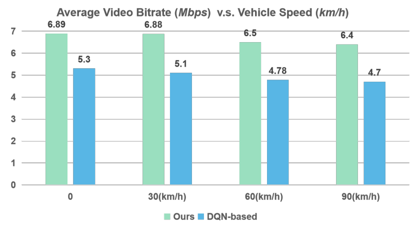


(b) Average video lag duration

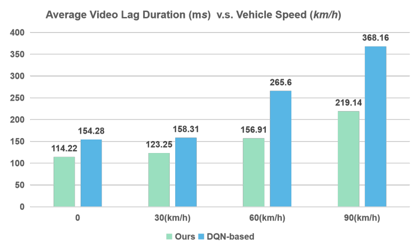


(c) Average video bitrate switching amplitude

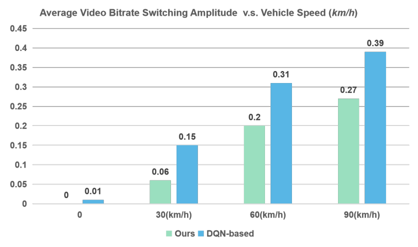
Fig. 3. Average video bitrate, video lag duration, and video bitrate switching amplitude with different user numbers and different video delivery strategies, when the speed of users is set as 30 km/h.



(a) Average video bitrate



(b) Average video lag duration



(c) Average video bitrate switching amplitude

Fig. 4. Average video bitrate, video lag duration, and video bitrate switching amplitude with different vehicle speeds and different video delivery strategies, when the user number is set as 20.

remaining within the local RSU's coverage range falls as average vehicle speed rises, link switching frequency gradually rises, and users' QoE gradually deteriorates. Additionally, Table I demonstrates that, depending on the vehicle speed, our suggested approach offers a greater response probability. Second, Fig. 3 demonstrates how our proposed strategy can provide users with a more fluid and high-quality visual experience when watching videos.

V. Conclusion

A video streaming data distribution strategy for video-on-demand services in the IoV is proposed in this study. We develop a DTDE-based algorithm to adaptively allocate transmission resources and choose video bitrate versions based on our framework. The experimental results show that compared with the existing DQN algorithm, our proposed computation-efficient algorithm performs better in terms of average video bitrate, average video lag duration, and the average video bitrate switching amplitude.

References

[1] M. H. C. Garcia and et al, "A tutorial on 5G NR V2X communications," *IEEE Commun. Surv. Tutor.*, vol. 1, no. 2, pp. 70–76, 2021.

[2] Cisco, "Cisco visual networking index: Global mobile data traffic forecast update," *2017–2022 White Paper*, 2019.

[3] C. Feng, H. H. Yang, D. Hu, Z. Zhao, T. Q. S. Quek, and G. Min, "Mobility-aware cluster federated learning in hierarchical wireless networks," *IEEE Trans. Wirel. Commun.*, vol. 21, no. 10, pp. 8441–8458, 2022.

[4] Z. Zhang, C.-H. Lung, M. St-Hilaire, and I. Lambadaris, "Smart proactive caching: Empower the video delivery for autonomous vehicles in icn-based networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7955–7965, 2020.

[5] R. Sun, Y. Wang, N. Cheng, L. Lyu, S. Zhang, H. Zhou, and X. Shen, "Qoe-driven transmission-aware cache placement and cooperative beamforming design in cloud-rans," *IEEE Trans. Veh. Technol.*, vol. 69, no. 1, pp. 636–650, 2020.

[6] T. X. Tran and D. Pompili, "Adaptive bitrate video caching and processing in mobile-edge computing networks," *IEEE Trans. Mobile Computing*, vol. 18, no. 9, pp. 1965–1978, 2019.

[7] T. Zhang and S. Mao, "Joint video caching and processing for multi-bitrate videos in ultra-dense hetnets," *IEEE Open J. Commun. Soc.*, vol. 1, pp. 1230–1243, 2020.

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

[9] G. Qiao, S. Leng, S. Maharjan, Y. Zhang, and N. Ansari, "Deep reinforcement learning for cooperative content caching in vehicular edge computing and networks," *IEEE Internet Things J.*, vol. 7, no. 1, pp. 247–257, 2020.

[10] Y. Guo, Q. Yang, F. R. Yu, and V. C. M. Leung, "Cache-enabled adaptive video streaming over vehicular networks: A dynamic approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 5445–5459, 2018.

[11] M. Wang, B. Cheng, N. U. L. Hassan, J. Wu, and C. Yuen, "Streaming delay-, expenditure- and quality-balanced video over TV white space," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4042–4057, 2020.

[12] M. S. Ito, D. Bezerra, S. Fernandes, D. Sadok, and G. Szabo, "A fine-tuned control-theoretic approach for dynamic adaptive streaming over http," in *IEEE Symp. Comput. and Commun.*, Larnaca, Cyprus, Jul. 2015, pp. 301–308.

[13] M. H, N. R, and A. M, "Neural adaptive video streaming with pensieve," in *Proc. ACM Special Interest Group Data Commun. (SIGCOMM)*, 2017, pp. 197–210.

[14] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Machine Learning*, 2015, pp. 1889–1897.