QoS Guaranteed Network Slicing Orchestration for Internet of Vehicles

Yaping Cui, Xinyun Huang, Peng He, Dapeng Wu, and Ruyan Wang

***Abstract*—To support the diversified quality of service (QoS) requirements of application scenarios, network slicing has been introduced in mobile cellular network. It allows mobile cellular network operators to accomplish the creation of multiple logically isolated networks on common network infrastructure flexibly depending on specified demands. Meanwhile, in Internet of Vehicles (IoV), it is very intractable to supply a stable QoS for the vehicles, especially for the dynamic vehicular environments. Thus, we investigate the IoV slicing problem in this paper, and propose a QoS guaranteed network slicing orchestration, namely, long short-term memory based deep deterministic policy gradient algorithm (LSTM-DDPG), to ensure the stable performance for the slices. Specifically, we first decouple the resource allocation problem into two sub-problems. After that, the deep learning and reinforcement learning are used to allocate resources collaboratively to solve these two questions. We use deep learning LSTM to track the characteristic of the long-term vehicular environment changing, and the reinforcement learning algorithm DDPG is utilized for online resource tuning. Extensive simulations have proved the effectiveness of the LSTM-DDPG, which can offer stable QoS to the vehicles with a probability greater than 92%. We also demonstrated the adaptiveness of the proposed orchestration with different slicing environments, and the performance is always optimal compared to that of other algorithms.**

***Index Terms*—Internet of vehicles, network slicing, deep reinforcement learning, resource allocation.**

I. Introduction

THE Internet of Vehicles (IoV) has become a popular paradigm to provide reliable Internet services for drivers and passengers [1]. Through dense vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications, the purpose of improving road safety, transportation efficiency and infotainment experience would been achieved. Provisioning these diverse services exhibits differences in the requirements of communication performance indicators, e.g., rate, reliability, and latency. For instance, the latency of services related autonomous driving should below a few milliseconds to provide better safety. Other services, such as value-added services, mainly aim to reduce traffic jams and provide traveler entertainments, and allow longer latency and lower reliability [2], [3]. The extremely diverse communication criterions lead to heterogeneous requirements of quality of service (QoS), which poses a severe challenge to the legacy one-size-fits-all architecture [4], [5].

To cope with the aforementioned challenges, network slicing is introduced into the fifth generation and beyond (5G/B5G) systems, which based on the virtualization technologies, e.g., network function virtualization (NFV) and software defined network (SDN), and can create multiple logically isolated networks over shared physical infrastructure [6]. More concretely, SDN, as one of the enablers of network slicing, has the ability to orchestrate network resource with SDN controller dependents on its collected information. Each vehicle reports its state information to base stations (BSs), then the SDN controller connected to the BS is considered to orchestrate the resources and perform resource allocation according to the information [7], [8]. On the basis of network slicing, deviate from the present wireless network, the physical resources are allocated to each logical network slice dynamically and flexibly according to users' demands for supporting extensive use cases with various QoS requirements [9], [10].

When conducting resource allocation in network slicing, the owner of each network slice, i.e., tenant, evaluates the needed resources of its served users and applies for resource preparation from the central controller. Then, the controller dynamically allocates the virtualized resources to slices with respect to all the received reservation requests. Finally, each tenant begins to allocate resources to its subscribed users [11], this procedure is also considered as intra-slice resource allocation. In general, the change of traffic load in network slicing may lead to a degradation in resource efficiency and exasperate the provided QoS. Thus, the slice not only needs to have the customization function for the corresponding service category that required by users, but also the ability to accommodate to the uncertain traffic demands [12], [13]. In order to guarantee the performance isolation and maximize resource utilization under uncertain traffic needs, it is necessary to reconfigure network slice adaptively and intelligently.

With the increasing progress of artificial intelligence technology, machine learning (ML) has been widely applied in

This work was partially supported by Natural Science Foundation of China (NSFC) under Grant 61801065, Grant 61771082, Grant 61871062, Grant 61901070, and U20A20157, in part by the Science and Technology Research Program of Chongqing Municipal Education Commission (Grant KJQN202000603, and Grant KJQN201900611), in part by the Natural Science Foundation of Chongqing (Grant cstc2020jcyj-zdxmX0024), and in part by University Innovation Research Group of Chongqing (Grant CXQT20017).

Y. Cui, X. Huang, P. He, D. Wu, and R. Wang are with the School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China, and with Advanced Network and Intelligent Connection Technology Key Laboratory of Chongqing Education Commission of China, and with Chongqing Key Laboratory of Ubiquitous Sensing and Networking (e-mail: cuiyp@cqupt.edu.cn; s190101129@stu.cqupt.edu.cn; hepeng@cqupt.edu.cn; wudp@cqupt.edu.cn; wangry@cqupt.edu.cn). Y. Cui is the corresponding author.

Y. Cui is also with the School of Aeronautics and Astronautics, University of Electronic Science and Technology of China, Chengdu 611731, China.

Manuscript received xx xx, 2021; revised xx xx, 2021.

wireless communications [14], [15]. Compared with the numerical optimization method, it shows a stronger ability to solve complex problems, which makes ML more suitable for the dynamic feature of IoV [13]. Moreover, reinforcement learning (RL), as an important branch of ML, learns the optimal decisions by maximizing cumulative reward in the long term. By interacting with the unknown environment, RL methods are particularly appropriate for decision making in such a changing environment to tackle the uncertainty and dynamics.

Despite the potentials and advantages of network slicing, the adaptation of network slicing to IoV also remains challenging. For IoV slicing, it is easily influenced by the inherent variation of the vehicular environment. This requires the slicing system to make resource allocation decisions with partial system status information, and the validity of the obtained solution is ensured during the entire operating period. In addition, with the IoV becoming progressively dynamic, heterogeneous, and large-scale, the real-time tracing and explicit modeling of the network environment are acquiring gradually difficultly, which makes traditional mathematical methods intractable [16]. On the other hand, as the high mobility of vehicles, the general resource allocation way only according to the real-time service request is incapable to supply vehicles the guarantee of stable service performance within a specified time range.

In order to deal with the mentioned problems, this paper proposes the long short-term memory based deep deterministic policy gradient algorithm (LSTM-DDPG) to cater to the diversified QoS requirements of vehicles. The early version of this paper has been published in a conference [17], the current manuscript is an extensive extension of the previously published paper. The LSTM is used to track the long-term features of request changes, and performs long timescale resource allocation. Furthermore, to handle the problem of the IoV susceptible to the varying environment, we use the RL DDPG algorithm to perform more fine-grained resource allocation by adjusting the allocated resource for the slice, such that the vehicle's requirements can be satisfied within a time interval. Simulation results exhibits that the proposed orchestration is able to offer stable QoS to the vehicles with a stability probability greater than 92%.

The main contributions of this paper can be summarized as follows:

- We analyze the stability performance of the slice by considering the dynamic request and the vehicle's movement. After that, the resource allocation problem is formulated and decoupled into two sub-problems, i.e., the long timescale resource allocation problem and the short timescale resource scheduling problem.
- We propose a QoS guaranteed network slicing orchestration, i.e., LSTM-DDPG, of which deep learning and RL collaboratively allocate resources. Specifically, deep learning, i.e., LSTM, is employed for allocating dedicated resources, and RL DDPG performs online resource scheduling that can calibrate the offset between the real-time environment changing and historical knowledge.
- Compared with the benchmarks, we prove the effectiveness of the proposed QoS guaranteed resource allocation orchestration for the targeted IoV slicing problem.

The rest of this paper is organized as follows. Section II presents the related work. Section III describes the system model that includes the network model and slicing model. Section IV formulates the resource allocation problem and then decomposes it into two sub-problems. In Section V, we introduce the proposed QoS guaranteed network slicing orchestration, LSTM-DDPG. We use the LSTM algorithm to predict the required resources in a long timescale, and use the RL DDPG algorithm to adjust the slice resources dynamically in a short timescale. In Section VI, we show the simulation results to evaluate the performance of the proposed LSTM-DDPG, followed by a brief summary in Section VII.

II. Related Work

Network slicing is categorized into two camps, core network slicing and radio access network (RAN) slicing, the former in charge of configuring network nodes, links as well as network topologies, the latter is mainly responsible for wireless resource management [18]. Core network slicing has been investigated in a vast amount of literatures [16], [19]–[21]. In [16], the authors presented a two-stage slicing optimization model. They first predicted link traffic to decide long-term feasible slicing deployment strategy, after that the Lyapunov stability theory was used for online optimization to calibrate misloading calibration. Limited by the capacity of links and nodes in physical infrastructure, a mixed binary linear programming problem has been formulated in reference [19] to minimize total link traffic, then two heuristic algorithms have been proposed. Analogously, reference [20] elaborated on the reconfiguration problem in core network slicing, it used the branching dueling Q-network that has the ability to compress action space for reducing configuration cost. In [21], the authors introduced a multilayer complex network model to help the analysis of slice deployment, and used this model to research the traffic performance of slices. But, from the perspective of network slicing goals, it is imperfect to simply consider the status of nodes and links in the core network without considering the RAN, especially in the complex IoV. For instance, the ultra-reliable and low-latency transmission of safety-related services with different QoS requirements is still a challenge for IoV [22]. Hence, the RAN slicing is urgent to research.

In the field of RAN slicing, literatures [23]–[30] have done a detailed study. The network slicing request problem was investigated in [23], the authors modeled it as a Markov decision process (MDP) and the RL method was used for deciding whether to accept the slicing request. Reference [24] researched the resource management in RAN slicing for executing user access control as well as bandwidth allocation more efficient, whose target is to decrease resource cost while satisfying QoS constraint. Note that the access strategy proposed by [24] cannot apply to IoV, because this strategy may conflict with the safety criterion since the fact that the system tends to serve the most appropriate users and ignores some crucial information for increasing resource utilization rate. According to the QoS requirements of users, [25] proposed a

user handover strategy based on Q-learning with data sharing to minimize handover cost. The Q-learning can obtain the optimal policy when state and action spaces of the problem are small. Inversely, when there exists a large state or action space, it converges difficultly. Therefore, it is necessary to introduce deep neural networks into network slicing strategies based on RL to cope with complex environmental conditions and large action spaces.

Considering the requirements of different applications, an energy-efficient RL method was proposed in [26] to allocate wireless resources and power for RAN slicing in 5G networks. In order to achieve the aim of rapid convergence with a fine-grained spectrum allocation, reference [27] imported discrete normalized advantage functions into deep Q-networks (DQN) by obtaining the advantage and state-value terms from the Q-value function. It utilized the deterministic policy gradient descent (DPGD) algorithm to avert the needless computing. Similarly, reference [28] proposed a generative adversarial network-powered deep distributional Q network (GAN-DDQN), where the changing service requirements were treated as the state of the environment, and the allocated resource was considered as action. The main idea conceals in this algorithm is to reduce the influence of randomness and noise. Differ from [27] and [28], literatures [29], [30] use a compound learning architecture that cooperates RL with deep learning. The objective in [29] is to ensure performance isolation between slices, while [30] aims to maximize spectrum efficiency (SE) and the service level agreements satisfaction ratio (SSR).

For IoV slicing, the authors in [31] presented a new network slicing scheme, the performance of reliability and delay in the specific field of vehicular communications has improved. Reference [32] researched two common slices, enhanced mobile broadband (eMBB) slice and vehicle-to-everything (V2X) slice, and proposed an efficient network slicing scheme using off-line RL. It executes resource allocation to maximize resource utilization and meet the needs of each RAN slice. In fog RAN, reference [33] designed a Monte Carlo tree search based intelligent resource allocation algorithm, which formulated the slice scheduling scheme as a MDP. Reference [34] presented a deep RL based approach to solve the resource allocation in IoV slicing, in which the channel and power distribution, slice selection, and broadcast distance were considered jointly. In [35], the authors proposed a dynamic network slicing framework to allocate wireless resources and computing resources as well as the computation workloads. They combined deep reinforcement learning (DRL) and convex optimization to solve this coupled resource allocation problem.

III. System Model

A. Network Model

As depicted in Fig. 1, we consider the IoV consisting of base station and vehicles. The BS is capable of communicating large volumes of data with vehicles that have their own specific service requirements. To provide diverse services for vehicles, the shared common network infrastructure is divided into multiple logically isolated networks, e.g., slice 1, slice 2, and slice 3 in Fig. 1. It should be noted that one vehicle may have

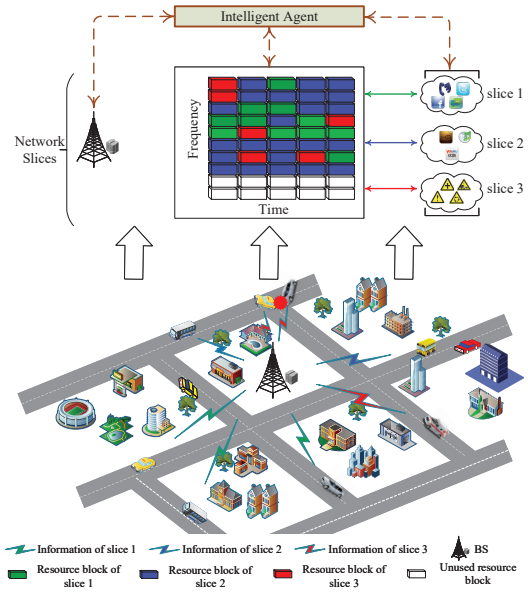


Fig. 1: IoV slicing system model.

various service requirements that map to different slices [5]. For the sake of notation, we assume that one vehicle has only one service and is permitted to access only one slice. Notice that this case can be extended to the scenario where the vehicle has multiple services.

Specifically, the protocol structure of BS is divided into three logical layers: the BS interface layer, BS virtualization layer, and BS resource layer [31]. The main function of the BS interface layer is separating the resource needs of slices off according to the type of each slice, it also provides different interfaces for slices. The BS virtualization layer allocates local RBs to each slice so as to satisfy the requirement of slices. Moreover, local resource blocks (RBs) are managed by the BS resource layer. As shown in Fig. 1, BS collects all information of vehicles' requests and provides a virtualized platform to share network resources with all slices. The slice applies for unused RBs to BS when it is insufficient to satisfy the signed service level agreement (SLA) of the requests.

B. Slicing Model

We denote the aggregate frequency bandwidth as W, which is shared by all vehicles that request service. The set of slices is denoted by $\mathcal{V}$, with $|\mathcal{V}| = V$. To facilitate analysis, the traffic of slice is expressed as the status of accessed vehicles [36]. System state $\mathcal{U}_v, v \in \mathcal{V}$ represents the set of vehicles on slice v. Specially, we use n_v to denote the cardinality of set $\mathcal{U}_v$, i.e., $|\mathcal{U}_v| = n_v$. Further, $\boldsymbol{n}^t = [n_1^t, \cdots, n_v^t, \cdots, n_V^t]$ is used to denote the traffic requirements of all slices at t timeslot. Without loss of generality, $n_v^t = 0$ when there is no traffic needs for slice v at t timeslot.

According to the SLA of the slice, each slice should make an assurance of QoS requirement for the served vehicles. Based on the characteristic of the IoV, e.g., the strict delay constraint [37], we define the QoS provided by slice as the minimum threshold of transmission rate $R_v, v \in \mathcal{V}$ for

vehicles, which means that the rate of vehicles on slice v no less than threshold R_v, can be expressed as

$$r_u \geq R_v, \forall u \in \mathcal{U}_v, v \in \mathcal{V}, \tag{1}$$

where r_u is the transmission rate of vehicle u to BS. Moreover, because of the indeterminacy of the traffic load that varying at different times, it is imperative to perform resource reservation deliberately for slices from the resource pool. Tracking the request changes over a long-term scale and reducing reconfiguration overhead are both accomplished by reservation [38]. So, the total resources are divided into the shared resource and dedicated resource, i.e., $W = W^s + \sum_{v \in \mathcal{V}} W_v^d$, where W^s is the shared resource, W_v^d is the dedicated resource allocated to slice v. The specific determination way and calculation process will be explained in Section IV.

IV. Problem Formulation

In this section, we analyze the stability performance of service provided by slice to vehicles. Following this analysis, the resource allocation problem is formulated and then we decomposed it into two sub-problems, that is, the resource allocation problem in a long timescale and the resource scheduling problem in a short timescale.

A. Resource Allocation Problem

When allocating resource to slices, it is essential to take the system state $\mathcal{U}_v^t$ into account, where we use $\mathcal{U}_v^t$ to express the system state at t timeslot. Under current $\mathcal{U}_v^t$, the allocated resource for slice v is denoted by h_v^t. In order to make the instant transmission rate satisfy formula (1), the following inequality should be held:

$$h_v^t \geq \widetilde{h}_v^t = \sum_{u \in \mathcal{U}_v^t} \frac{R_v}{\log_2\left(1 + SINR_{ub}\right)}, v \in \mathcal{V}, \tag{2}$$

where $\widetilde{h}_v^t$ indicates the minimum value of resources required by slice v to satisfy the QoS of vehicles, $SINR_{ub} = \frac{P_u G_{ub}}{I + \sigma^2}$ is the received signal-to-interference-plus-noise ratio (SINR) of the V2I link of vehicle u, in which P_u is vehicle transmit power, G_{ub} denotes the channel gain of vehicle u that considers path loss, shadowing, and fast fading, I is the received interference power, σ^2 is the noise power. In the similar way, the allocated resources for slice v at $t+1$ timeslot can be expressed as h_v^{t+1}.

The system is supposed to allocate resources flexibly according to the slice requirements enabling the adaptation of dynamic environment so as to avert the negative impact caused by the environmental changes on the performance of slices. Furthermore, the stringent performance isolation of slice has ensured that the performance for vehicles does not degrade to below the QoS the SLA provided before and after allocating resource. The system state $\mathcal{U}_v, v \in \mathcal{V}$ may vary before and after resource allocation on account of the vehicle's mobility and stochastic nature of request. We denote current state and next state as $\mathcal{U}_v^t$ and $\mathcal{U}_v^{t+1}$, respectively. The transmission rate of vehicle $u \in \mathcal{U}_v^t, v \in \mathcal{V}$ is given by

$$r_u^t = \omega_u \log_2\left(1 + SINR_{ub}\right), \tag{3}$$

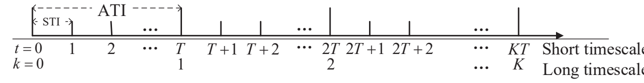


Fig. 2: The division of two-timescale.

where $\omega_u = \frac{h_v^t}{n_v^t}$ is the obtained spectrum resource of vehicle u. Similarly, we can obtain the virtual rate of vehicle $\hat{r}_{u'}^{t+1}, u' \in \mathcal{U}_v^{t+1}$ according to the load at $t+1$ timeslot, which is defined as $\hat{r}_{u'}^{t+1} = \hat{\omega}_u \log_2\left(1 + SINR_{u'b}\right)$, where $\hat{\omega}_{u'} = \frac{h_v^t}{n_v^{t+1}}$ is determined by the resources allocated at t timeslot. Notice that the resource used for calculating virtual rate $\hat{r}_{u'}^{t+1}$ is h_v^t instead of h_v^{t+1} that based on the requirement at $t+1$ timeslot, i.e., $\hat{\omega}_{u'} = \frac{h_v^t}{n_v^{t+1}}$. The main idea behind this definition is to judge whether the last allocated resources stabilize the communication rate of the vehicle, that is to say, whether the vehicle rate at $t+1$ timeslot still meets the formula (1) according to the last allocated resources. Thus, the following constraint by which slice can provide stable service between t and $t+1$ timeslot should be satisfied

$$\frac{\sum_{u' \in \mathcal{U}_v^{t+1}} \hat{r}_{u'}^{t+1}}{n_v^{t+1}} \geq R_v. \tag{4}$$

Obviously the slice can guarantee the QoS of vehicles when the system allocating resource superfluously, but this will cause serious waste. So, we also need to reduce the cost of allocating resource. Considering the depletion in long-term $t \in \mathcal{T} = \{0, 1, 2, \cdots, T, T+1, T+2, \cdots, KT\}$, the slice resource allocation can be formulated as

$$\begin{aligned} p(1): \min \ & \sum_{t \in \mathcal{T}} \sum_{v \in \mathcal{V}} h_v^t \\ \text{s.t. } C1: & \sum_{v \in \mathcal{V}} h_v^t \leq W, \forall t \in \mathcal{T} \\ C2: & \ r_u^t \geq R_v, \forall u \in \mathcal{U}_v^t, v \in \mathcal{V} \\ C3: & \ \frac{\sum_{u' \in \mathcal{U}_v^{t+1}} \hat{r}_{u'}^{t+1}}{n_v^{t+1}} \geq R_v, \forall v \in \mathcal{V}, \end{aligned} \tag{5}$$

where h_v^t is the optimization objective. Constraint $C1$ states the sum of allocated resources for slice that is constrained by total resources. The QoS requirement of slice emerges from constraint $C2$. Constraint $C3$ indicates that the allocated resources should ensure the vehicle's transmission rate stable during the interval. In $C3$, variables $\hat{r}_{u'}^{t+1}$ and n_v^{t+1} are obtained according to the information of vehicles that connected to slice v at next timeslot. The number of requests and the position of vehicles in slice v may change with time, so this constraint reflects the change of service request and vehicle mobility.

B. Problem Decomposition

To track the dynamic property of request and satisfy vehicle's requirements, motivated by [39], we accomplish the resource allocation process by two decisions, i.e., resource allocation in a long timescale and the resource scheduling in a short timescale. Specifically, we define long timescale and short timescale as the resource allocation time interval

(ATI) and the resource scheduling time interval (STI), respectively. The decision makes at the beginning of the interval, and their indexes are denoted by k and t, respectively, i.e., $\mathcal{T} = \{kT, kT+1, \cdots, (k+1)T-1\}, k \in \{0, 1, \cdots, K\}$. As shown in Fig. 2, each ATI consists of T STIs. In practical slice deployment, it is more rational for decision making to utilize historical information of slices. To overcome the changes of IoV over long timescales, at the beginning of kth ATI, the system allocates the dedicated resource $W_v^{k,d}$ for each slice according to the previous service requests. Moreover, at the beginning of each STI in ATI, based on immediate system status $\mathcal{U}_v, v \in \mathcal{V}$, the system makes a fine-tuning to satisfy $C2$ and $C3$.

Long timescale resource allocation: In realistic environment, user's request always follows particular rules, e.g., a strong daily pattern [40]. The slice frequent reconfigurations based on the instant requirements may cause service interruption and incur certain overhead, and keeping resource changes subtle is helpful for network performance improvement [38]. In addition, the selection of the duration of each ATI ensures that the statistical information of user traffic within ATI will not change [41]. So, the system will predict future demands according to history, then it performs the dedicated resource allocation in a long timescale to adapt to the long period traffic variations in the networks. For the long timescale resource allocation problem, our objective is minimizing error between predicted value $\widetilde{W}_v^{k,d}$ and real $W_v^{k,d}$, that is given by

$$p(2): \arg\min_{\widetilde{W}_v^{k,d}} \left| \widetilde{W}_v^{k,d} - W_v^{k,d} \right|, \tag{6}$$

where $W_v^{k,d}$ is calculated by the mean number of service requests and mean channel state information (CSI) of slice v within kth ATI. According to the predicted value $\widetilde{W}_v^{k,d}$, the system performs the dedicated resource allocation in a long timescale. In other words, each slice obtains the dedicated resource $\widetilde{W}_v^{k,d}$ at the beginning of kth ATI. However, these fixed resources may not satisfy the vehicle's real-time requirements dynamically during such a long timescale. Therefore, we will consider the impacts of real-time dynamic traffic requirements of each STI during an ATI.

Short timescale resource scheduling: The system status may be different in each STI, including the number of service requests and the CSI of the corresponding vehicles. Thus, the capability to adapt to the changing traffic requirements is crucial for slice. We denote the scheduled resource for slice v within kth ATI by $W_v^{k,t}, t \in \{kT, kT+1, \cdots, (k+1)T-1\}$, $W_v^{k,t} > 0$ indicates that the dedicated resource of slice v is incompetent to satisfy the requirements of vehicles, at the beginning of the present STI, so extra resources are needed to be dispatched from the shared resource pool of BS, and vice versa. The online resource scheduling problem at each STI can be expressed by

$$\begin{aligned} p(3): \min \ & \sum_{v \in \mathcal{V}} W_v^{k,t} \\ \text{s.t. } & C1 - C3 \\ & C4: h_v^t = \widetilde{W}_v^{k,d} + W_v^{k,t}, \end{aligned} \tag{7}$$

where h_v^t is the sum of the dedicated resource $\widetilde{W}_v^{k,d}$ and scheduled resource $W_v^{k,t}$. The existence of constraint $C3$ complicates the problem. In resource scheduling, we consider not only the current service demand needs, but also the service requests and channel changes at the next STI due to the vehicle's mobility. In order to solve this problem, the RL method will be used for slice resource scheduling.

V. The QoS Guaranteed Network Slicing Orchestration

As mentioned above, the resource allocation problem has formulated as a long timescale resource allocation problem and a short timescale resource scheduling problem. To cope with the long-term traffic changes, the system should predict the future needs of resource such that executes the dedicated resource allocation. In a short timescale, the system needs to perform short timescale resource scheduling in terms of the dynamic service requests of the network to achieve the purpose of online fine-tuning the allocated dedicated resource. In this section, we will propose the QoS guaranteed network slicing orchestration, i.e., LSTM-DDPG, of which the deep learning algorithm LSTM used for resource predicting and the RL algorithm DDPG performs online resource scheduling.

A. The LSTM based Dedicated Resource Allocation

To obtain the dedicated resources in a quantitative way, the critical issue is how to predict the mean resource needs in future long timescale accurately. The LSTM, as a derivative of recurrent neural networks (RNN), is capable of extracting the temporal correlation in sequences. Furthermore, LSTM avoids the gradient disappearance problem of conventional RNN and thus is widely used for classification and prediction. The internal structure of LSTM cell is shown in Fig. 3. Generally, each LSTM cell contains two states and three gates, i.e., cell state, hidden state, forget gate, input gate, and output gate [42]. The forget gate confirms what information should be forgotten in the last cell states now, the input gate is responsible for processing the current input, the output gate determines the current unit's output. We denote the output of the forget gate, cell state, and hidden state as f_t, C_t, H_t respectively, and can be formulated as

$$f_t = \sigma\left(W_f^x x_t + W_f^h H_{t-1} + b_f\right), \tag{8}$$

$$C_t = f_t \circ C_{t-1} + i_t \circ \tanh\left(W_c^x x_t + W_c^h H_{t-1} + b_c\right), \tag{9}$$

$$H_t = o_t \circ \tanh(C_t), \tag{10}$$

where

$$i_t = \sigma\left(W_i^x x_t + W_i^h H_{t-1} + b_i\right), \tag{11}$$

$$o_t = \sigma\left(W_o^x x_t + W_o^h H_{t-1} + b_o\right), \tag{12}$$

and W_i^x, W_f^x, W_c^x, W_o^x, W_i^h, W_f^h, W_c^h, W_o^h, b_f, b_c, b_i, b_o are the parameters of the neural network. Besides, in the above equations, notation $\circ$ denotes the Hadamard product and the functions $\sigma(\cdot)$ and $\tanh(\cdot)$ are sigmoid function and hyperbolic tangent function respectively. In this cell, the external input interface is x_t, and the output interface is H_t.

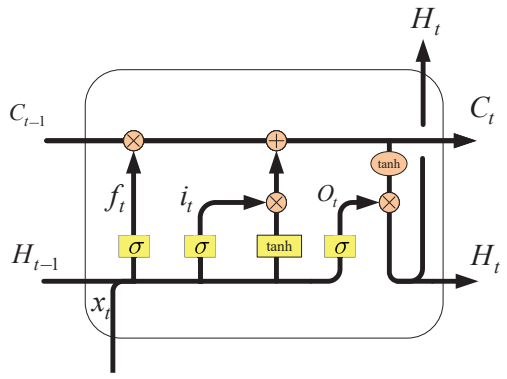


Fig. 3: The internal structure of LSTM cell.

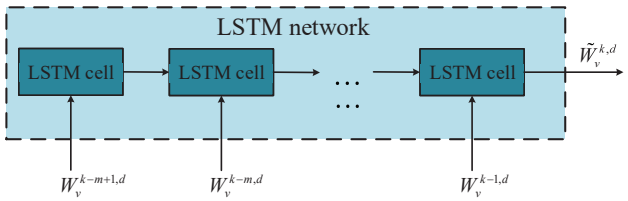


Fig. 4: The LSTM based resource prediction framework.

Moreover, the LSTM based resource prediction framework is shown in Fig. 4, which is composed of several LSTM cells in series, and the cell state delivered between adjacent cells. Inputting the historical average resource demands $\{W_v^{k-m,d}, W_v^{k-m+1,d}, \cdots, W_v^{k-1,d}\}$ to the LSTM, the predicted dedicated resources $\widetilde{W}_v^{k,d}$ will be emerged. This provides a baseline resource for the next phase, i.e., resource scheduling phase.

B. The DDPG based Online Resource Scheduling

After allocating the dedicated resource for each slice, the online resource scheduling algorithm is needed to calibrate the allocated resources, thus accommodates the slice to dynamic environments. In this stage, BS will collect vehicles channel status for resource scheduling, which may lead to additional communication overhead. There are many solutions to reduce this overhead, such as compressing local observation information before feedback it to BS through constructing a deep neural network [43], decreasing the update frequency of vehicles information [44]. The communication overhead will be acceptable by using the above solutions.

Mathematically, the decision process of online resource scheduling can be modeled as a MDP. At each time-step t, the agent observes a state $s_t \in \mathcal{S}$, then it selects an action a_t from action space. After reacting, the environment changes to the next state s_{t+1} with a probability $P(s_{t+1}|s,a)$, while the agent also receives an instant reward r_t. The accumulative discounted reward at time-step t is expressed by $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$, where $\gamma \in (0,1]$ is discount factor, the higher the value, the greater the influence of future rewards on the current action selection. The policy π is usually the mapping from state to the probability distribution of action, i.e., under a certain state s, the agent takes action based on the policy π . Furthermore, the purpose of learning is to search a best policy π^* to maximize the expectation of the cumulative return with any initial state, given by

$$\pi^* = \arg\max_{\pi} \mathbb{E}(R|\pi). \tag{13}$$

Generally, the action-value function $Q(s,a)$ and the state-value function $V(s)$ can be used to represent the cumulative return. Traditional RL methods include dynamic programming (DP), Monte Carlo algorithm (MC), Q-learning, etc., [45]. For the DP method, it requires updating the expectation of each state based on the values of all subsequent possible states and corresponding probabilities. Traversing the entire state set will make the DP method extreme computation complexity. The MC algorithm estimates the state-value by complete state sequences that are difficult to obtain in practice. Q-learning, a common RL algorithm, the basic idea is to calculate the Q-value of each state-action pair and save it in the Q-table. Then, it uses the $\epsilon - greedy$ strategy to take action depending on the records in Q-table. The updating formula of Q-value is given by

$$\begin{aligned} Q(s_t,a_t) \longleftarrow\ & Q(s_t,a_t) + \alpha\Big[r(s_t,a_t) \\ & + \gamma \max_{a_{t+1}} Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)\Big], \end{aligned} \tag{14}$$

where α is the learning rate, and it can be a constant, or is adjustable.

However, Q-learning is intractable to converge when it encounters a large state space or action space. The usual method to address this issue is approaching the actual action-value with functions. Thanks to the advances in computing power of the computer, deep neural networks have attracted extensive attention. The DQN uses neural networks to parameterize the action-value, for instance, the action-value is represented by $Q(s,a;\theta)$ [46]. Furthermore, many variants of DQN have been derived, e.g., Double DQN [47], Dueling DQN [48]. Despite these modifications have improved the performance of the networks, we remark that these algorithms do not apply to our problem. This is because the above algorithms are used to solve the problem with discrete action space. On the one hand, the fine-grained resource allocation requires discretizing resources in a tiny interval, which will lead to a huge size of the action space and ultimately result in more network parameters as well as higher computational complexity. On the other hand, if we reduce the size of action space by discretizing resources in a large interval, it will cause resource waste. Therefore, we use the DDPG algorithm that enables to handle the continuous action space to perform the fine-grained resource scheduling with a significantly reduced complexity [49].

Distinct from the value-based RL, the DDPG integrates policy gradient. The policy π can be represented as [50]

$$\pi(a,s) \approx \pi(a|s;\theta) = P(a|s;\theta). \tag{15}$$

Because the DDPG is a deterministic policy algorithm, that is, in the same state, the policy is determined by the action with the maximum probability. So, we rewrite the policy π as

$$\pi\left(s;\theta\right)=a. \tag{16}$$

In the following, we describe in detail the state, action, and reward.

1) State: For the resource scheduling problem, the real-time link situation is not suitable for the environmental state due to the variation of network load caused by the random request. The uncertain character of the number of links will cause dimension mismatching in the input if the real-time link is selected as the state. Thus, we define the state as

$$s_t=\left\{\widetilde{W}_1^{k,d},\widetilde{W}_2^{k,d},\cdots,\widetilde{W}_V^{k,d},\widetilde{h}_1^t,\widetilde{h}_2^t,\cdots,\widetilde{h}_V^t\right\}, \tag{17}$$

where $\widetilde{W}_v^{k,d}$ is the predicted resource requirements by using the LSTM, $\widetilde{h}_v^t$ is the resource requirement required by slice v.

2) Action: The action is defined as scheduling resources for each slice

$$a_t=\left\{W_1^{k,t},W_2^{k,t},\cdots,W_V^{k,t}\right\}. \tag{18}$$

3) Reward: The reward, an important part of the RL problem, would be designed elaborately to lead the agent to find the optimal policy, it reflects the optimization goal of our problem. Because the resource scheduling process occurs at each STI in ATI, for notational convenience, we have omitted the superscript k in the following formulas. As shown in problem $P(3)$, the objective is minimizing the resource cost while ensuring the transmission rate of service vehicles is stable. Considering the constraint conditions, the reward function is defined as

$$r_t=\sum_{v\in\mathcal{V}}\phi\left(h_v^t\right)+\sum_{v\in\mathcal{V}}\beta_v\cdot I\left(h_v^t\right)+\delta\left(\boldsymbol{h^t}\right), \tag{19}$$

where

$$\phi\left(h_v^t\right)=\begin{cases}\ln\left(\frac{\widetilde{h}_v^t}{h_v^t}\right), & \text{if } h_v^t\geq\widetilde{h}_v^t\\ \ln\left(\frac{h_v^t}{\widetilde{h}_v^t}\right), & \text{if } h_v^t<\widetilde{h}_v^t\end{cases}, \tag{20}$$

$$\delta\left(\boldsymbol{h^t}\right)=\begin{cases}-\infty, & \text{if } \sum_{v\in\mathcal{V}}h_v^t>W\\ 0, & \text{otherwise}\end{cases}, \tag{21}$$

where we have used the $C4$ in equation (7), and $\boldsymbol{h^t}=[h_1^t,h_2^t,\cdots,h_V^t]$. The first term at the right side in (19) is decided by the instant needed resources and allocated resources. As shown in equation (20), the agent will get a negative reward if superfluous resource is allocated for slice, i.e., $h_v^t>\widetilde{h}_v^t$. In this case, the more resources allocated, the smaller the reward acquired. This means that the agent does not allocate redundant resources largely to the slice. Inversely, when the resource of slice is insufficient, i.e., $h_v^t<\widetilde{h}_v^t$, the agent also gets a negative reward. But it will get a greater reward if the slice gets more resources. So, to maximize the reward of this term, whose maximum value is zero, the agent tends to allocate resources to meet the needs of the slice exactly.

Then, we explain the second term at the right side in (19). $I\left(h_v^t\right)$ is an indicator function, when the total resource of slice satisfies the constraint $C3$, i.e., $\frac{\sum_{u'\in\mathcal{U}_v^{t+1}}\hat{r}_{u'}^{t+1}}{n_v^{t+1}}\geq R_v$, its value equals to 1, otherwise, equals to 0. This function characterizes the stability of the slice. When the slice V can guarantee the stability of the link rate of the served vehicles in the STI, the agent will get a positive reward β_v. Meanwhile, $\boldsymbol{\beta}=[\beta_1,\beta_2,\cdots,\beta_V]$ is the weight of each slice, the requirements of stable performance of different slices can get safeguard via adjusting $\boldsymbol{\beta}$.

The last term at the right side in (19) is a Dirac impulse function. The reward will be negative infinite if the allocated resource overloading. Thus, massive impractical slicing deployment can be avoided.

Finally, the DDPG network is structured. The DDPG network consists of two networks, namely, the evaluation network and the target network. As shown in Fig. 5, each of the two networks can also be divided into two subnetworks, i.e., evaluation actor network, target actor network, evaluation critic network, target critic network, and their neural network parameters are represented by θ, θ', ω, ω', respectively. Moreover, to speed up the convergence, we also use the experience replay memory. At each time step t, the evaluation actor network chooses action according to the state s_t and present parameter θ. Then it observes the next state s_{t+1} while acquiring reward r_t. Expressly, To encourage exploration, we represent the action as a Gaussian probability distribution

$$a_t\sim\mathcal{N}\left(\pi\left(s_t;\theta\right),\sigma^2\right), \tag{22}$$

where $\pi\left(s_t;\theta\right)$ is the output of the evaluation actor network and also as the mean of probability distribution, σ is standard deviation.

The agent will save the transition tuple (s_t,a_t,r_t,s_{t+1}) to the replay memory to train networks. In training, the agent samples mini-batch samples at every iteration. The evaluation critic network is mainly responsible for calculating Q-value according to the sampled samples. The target actor network is capable of selecting action a'_{t+1} from s_{t+1}, it is worth noting that this action is only used for updating the network and will not be executed by the agent. The target actor network is used for forecasting Q-value, which is formulated by

$$Q'\left(s_t,a_t;\omega'\right)=r_t+\gamma Q\left(s_{t+1},a'_{t+1};\omega'\right). \tag{23}$$

Next, the network will be updated. Similar to DQN, the loss function of the evaluation critic network is the mean square error of Q-value, can be given by

$$J(\omega)=\frac{1}{K}\sum_{i=1}^{K}\left(Q'\left(s_i,a_i;\omega'\right)-Q\left(s_i,a_i;\omega\right)\right). \tag{24}$$

The policy gradient of the evaluation actor network can be represented as

$$\nabla_\theta J(\theta)\approx\frac{1}{K}\sum_{i=1}^{K}\left[\nabla_a Q\left(s,a;\omega\right)|_{s=s_i,a=\pi(s_i;\theta)}\nabla_\theta\pi\left(s;\theta\right)|_{s=s_i}\right]. \tag{25}$$

Now we have the update method of the evaluation network. After that, the parameter of the target network is updated

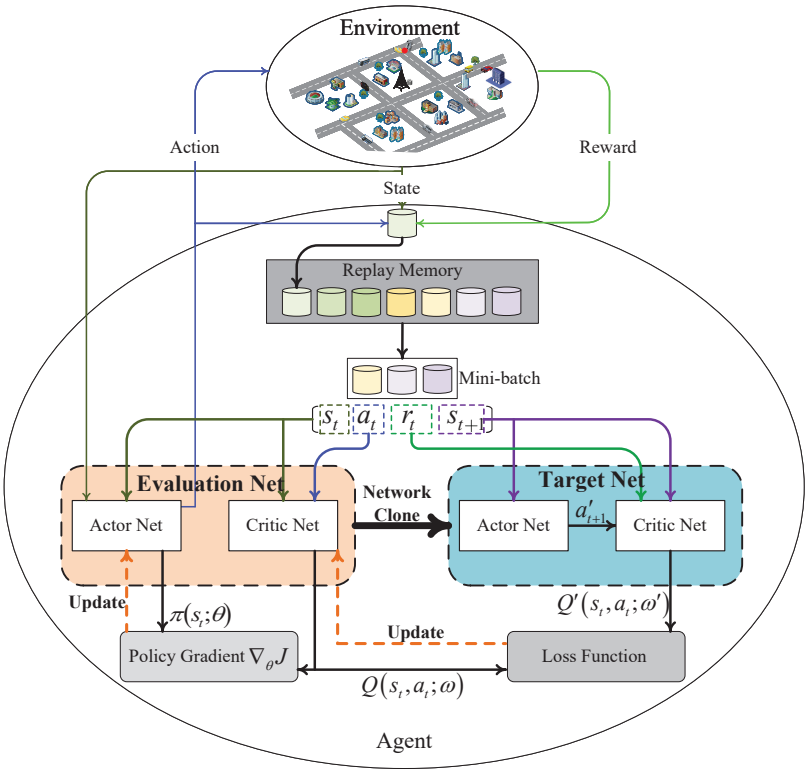


Fig. 5: The DDPG based online resource scheduling framework.

regularly by that of the evaluation network in a soft update manner, and can be presented as

$$\omega' \leftarrow \tau\omega + (1-\tau)\,\omega', \tag{26}$$

$$\theta' \leftarrow \tau\theta + (1-\tau)\,\theta', \tag{27}$$

where τ is the update coefficient, which determines the step size of updating. The online resource scheduling algorithm based on the DDPG is shown in algorithm 1.

C. Complexity Analysis

In the real deployment of the resource allocation algorithm, it divides into training stage and validating stage. We mainly discuss the computational complexity of the training stage, because most of the computation is concentrated in the training phase. For example, the target network and evaluation critic network are not applied in the validation process.

For the sake of clarity, we discuss time complexity based on the floating point operations per second (FLOPS). The addition, subtraction, multiplication, division, and exponentiation, etc., are considered as a floating point operation. Furthermore, for activation layers, the computations of Relu layers, sigmoid layers, and tanh layers are Q, $4Q$, and $6Q$ for Q inputs respectively [51].

For the LSTM networks, the time complexity is $O\left(mV^2\right)$ for each iteration, where m is the time step of the LSTM, V is the number of slices. As an ATI contains T STIs, the average time complexity of the LSTM networks in the training process is $O\left(\frac{mV^2}{T}\right)$.

The time complexity of DDPG is given by

$$\begin{aligned} O\Bigg(\Bigg(2\sum_{j=1}^{J} v_{actor,j}^{activation} u_{actor,j} + 2\sum_{l=1}^{L} v_{critic,l}^{activation} u_{critic,l} \\ + 2\sum_{j=0}^{J-1} u_{actor,j}u_{actor,j+1} + 2\sum_{l=0}^{L-1} u_{critic,l}u_{critic,l+1}\Bigg)\Bigg) \\ = O\Bigg(\sum_{j=0}^{J-1}\left(v_{actor,j+1}^{activation} + u_{actor,j}\right)u_{actor,j+1} \\ + \sum_{l=0}^{L-1}\left(v_{critic,l+1}^{activation} + u_{critic,l}\right)u_{critic,l+1}\Bigg), \end{aligned} \tag{28}$$

where J and L represent the number of layers of the actor network and the critic network, respectively, $u_{actor,j}$ ($u_{critic,l}$) is the number of units in the jth (lth) layer of the actor network (critic network), and $v_{actor,j}^{activation} u_{actor,j}$ ($v_{critic,l}^{activation} u_{critic,l}$) denotes the computations of the activation part, which determined by the type of the used activation function in each layer. Specifically, the value of $u_{actor,0}$ ($u_{critic,0}$) is equal to the input size of the actor network (critic network).

Algorithm 1 The DDPG based online resource scheduling algorithm

Phase 1: Initialize the networks
Initialize the evaluation network, target network, replay memory, the number of samples M, mini-batch K, standard deviation σ, update coefficient τ, , target network update frequency C.
Phase 2: Update the networks
for every k **do**
Allocate the dedicated resources for each slice according to the LSTM prediction results $\widetilde{W}_1^{k,d}, \widetilde{W}_2^{k,d}, \cdots, \widetilde{W}_V^{k,d}$
Observe the state $s = \left(\widetilde{W}_1^{k,d}, \widetilde{W}_2^{k,d}, \cdots, \widetilde{W}_V^{k,d}, \widetilde{h}_1^t, \widetilde{h}_2^t, \cdots, \widetilde{h}_V^t\right)$.
for $t = 1$ to T **do**
Select action with the evaluation actor network $a_t = \left(W_1^{k,t}, W_2^{k,t}, \cdots, W_V^{k,t}\right)$.
Execute action a_t and get reward r_t and observe new state s_{t+1}.
Store transition (s_t, a_t, r_t, s_{t+1}) in replay memory.
if $M \geq K$ **then**
Sample a random mini-batch of transitions (s_i, a_i, r_i, s_{i+1}) from replay memory.
Calculate $Q'(s_i, a_i; \omega')$ according to (23).
Update the evaluation network according to (24) and (25).
if $(T * C)\,\%C = 1$ **then**
Update the target network according to (26) and (27).
end if
end if
end for
end for

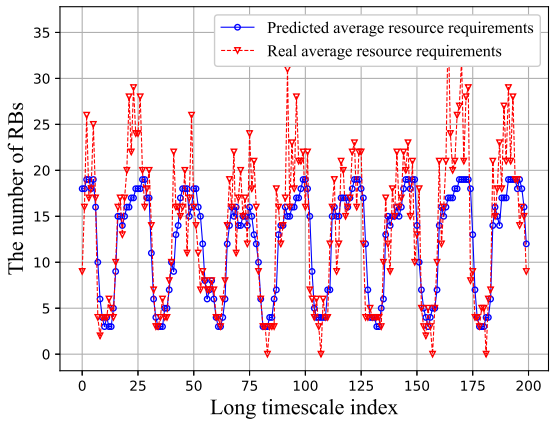
Therefore, the total time complexity of algorithm 1 is

$$O\left(\sum_{j=0}^{J-1}\left(v_{actor,j+1}^{activation}+u_{actor,j}\right)u_{actor,j+1} + \sum_{l=0}^{L-1}\left(v_{critic,l+1}^{activation}+u_{critic,l}\right)u_{critic,l+1}\right)+O\left(\frac{mV^2}{T}\right). \quad (29)$$

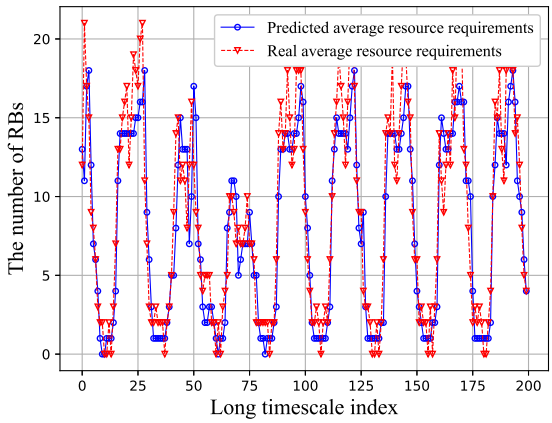
Moreover, simulation result in Fig. 7 has shown that the proposed LSTM-DDPG algorithm converges after about 20,000 iterations, which means that this algorithm can be trained in advance and then deployed for actual online resource allocation.

VI. Performance Evaluation

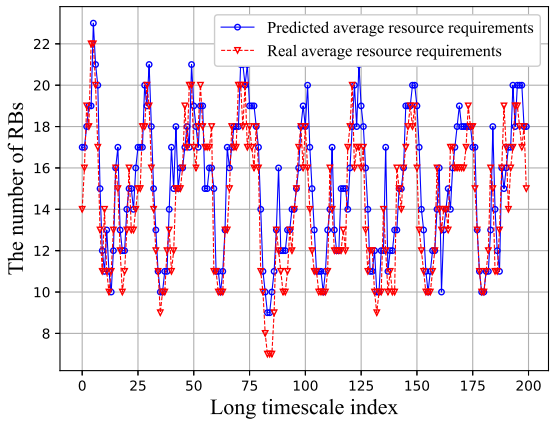
In simulations, we use the cellular traffic dataset of Milan, [52], which records the traffic changes of three services, i.e., short message service (SMS), call service, and Internet service, with a temporal interval of one hour. We treat these services as three slices and each record is regarded as the average number of accessed vehicles in slice over a long timescale. The default major simulation parameters of vehicular environment and neural network are listed in Table I [53], [54]. The learning rate of LSTM is set as 0.01, and time step is 4, i.e., the first four average resource demands are used to predict the average demands in next long timescale. In addition, we set the learning rate of the evaluation actor network and evaluation critic network in DDPG as 0.001



(a) slice 1.



(b) slice 2.



(c) slice 3.

Fig. 6: The predicted average resource requirements. (a) slice 1, (b) slice 2, (c) slice 3.

TABLE I: Simulation Parameters

Parameters	Value	Parameters	Value
Bandwidth	20 MHz	LSTM learning rate	0.01
BS antenna height	25 m	LSTM time step	4
BS antenna gain	8 dBi	LSTM batch size	32
BS receiver noise figure	5 dB	Actor hidden units	(120, 60, 3)
Vehicle antenna height	1.5 m	Actor activations	(Relu, Relu, Tanh)
Vehicle antenna gain	3 dBi	Actor learning rate	0.001
Absolute vehicle speed	36 km/h	Critic hidden units	(120, 60, 1)
Vehicle transmit power	23 dBm	Critic activations	(Relu, Relu, None)
Noise power	-114 dBm	Critic learning rate	0.002
Path loss model	128.1+37.6 $\log_{10} d$, d in km	Target update coefficient	0.01
Shadowing distribution	Log-normal	Discount factor	0.9
Shadowing standard deviation	8 dB	DDPG batch size	64

and 0.002, respectively. We compared the performance of the proposed LSTM-DDPG with that of other ML methods including the conventional DQN, advantage actor-critic (A2C) algorithm, asynchronous advantage actor-critic (A3C) algorithm, as well as the demand-based algorithm (DB) that allocates resources according to the immediate needs. Specially, we also demonstrate the performance of DDPG without the prediction model LSTM.

We first verify the LSTM performance with predicting the dedicated resource, i.e., the average number of RBs required in each long timescale. In this experiment, the requirements generated by 1000 samples in data set were used to train the network, the other 200 samples were used for testing. We compared the real average resource requirements with the predicted average resource requirements. As shown in Fig 6, the requirement of resources has a slight difference among slices due to the different services. Besides, because of the dynamic nature of the service requests, the resource requirements of the three slices change dynamically, and the daily characteristics map to the resource requirements completely. Except for the characteristics of the data itself, the curve fitting degree between the predicted values and reals is also exhibited in Fig 6. It can be seen that the LSTM enables to track the resource demand changes of slice in a long timescale, the mean absolute errors (MAEs) of resource requirements of three slices are 3.47, 2.02, 1.59 RBs, respectively. There may be gaps between the predicted resource requirements and the actual requirements, but it has a small impact on the overall network performance. Because the predicted average resource is used as part of the input of the DDPG to get the final result, rather than directly allocating resource to slice. Compared to the MAEs of slice 2 and slice 3, slice 1 shows a greater gap. The main reason to account for this phenomenon may be the variance of SMS is smaller than that of call and Internet. The variances of SMS, call, and Internet are 3.6, 11.7, 21.7 in the dataset, respectively. So, the predicted resource requirements of slice 1 are relatively stable, which is contrary to the daily characteristics of traffic, so as to cause a certain gap.

The prediction results of the LSTM will as a part of the input to train DDPG networks, so the convergence of DDPG is influenced by the LSTM. Fortunately, the LSTM network converges very fast in the simulation, converged after about 300 iterations, the impact on the convergence of DDPG was trivial. In Fig. 7, we show the convergence behavior of the proposed DDPG algorithm with increasing training iterations.

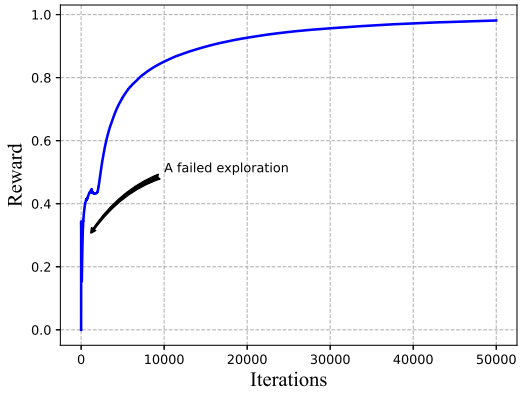
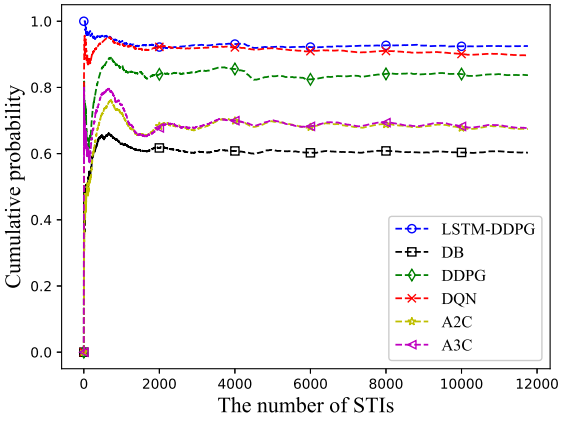


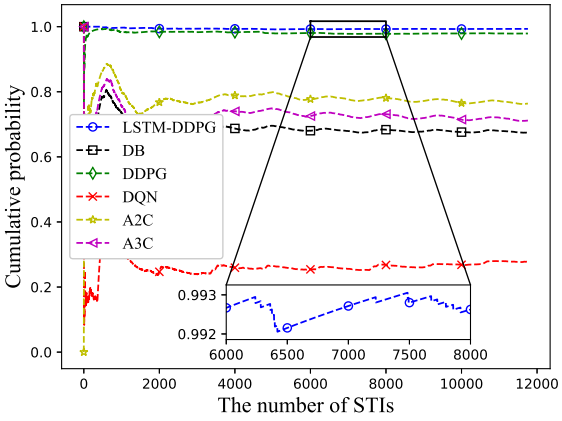
Fig. 7: Reward with increasing training iterations.

It can be seen that at the beginning of training, the agent has carried out some failed explorations due to the encouragement of exploration. As the number of iterations increases, its reward gradually converges to the maximum reward available. When the training iterations approximately reach 20, 000, the performance of the DDPG gradually converges.

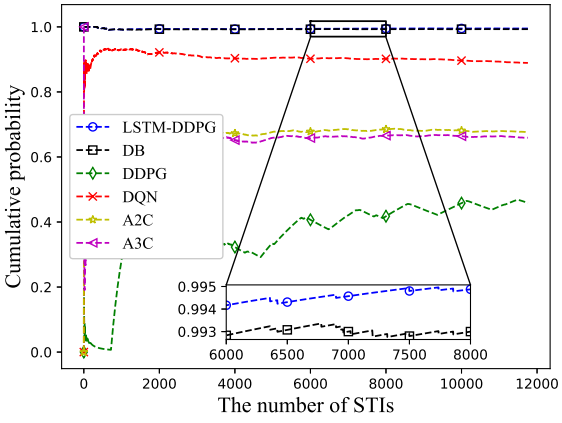
It is considered that the slice maintains stable performance in this STI when the resources allocated for the slice meet formulas (2) and (4). Furthermore, we defined the slice's cumulative stability probability as the ratio of the number of stable STIs, in which the slice can supply a stable QoS to the vehicles, and the total number of STIs. Fig. 8 shows the cumulative stability probabilities of three slices. As the restraint of QoS varies with slice, the cumulative stability probability of each algorithm under different slicing environments is also different. It can be seen that the cumulative probabilities of the DB have a great distinction under different slices, at the 8000th STI, the cumulative probabilities in three slices are 60.8%, 68.4%, 99.3%, respectively. The reason accountable for this issue is the probability that the resource allocation according to the immediate situation is able to meet the next moment is lower if the rate threshold of the slice is larger. For instance, slice 1 has a larger rate threshold than slice 2 and slice 3, so the variation of CSI caused by the vehicle's mobility makes the demands for RB more stringent such that the cumulative probability is lower in slice 1. For other algorithms, i.e., LSTM-DDPG, DDPG, A2C, A3C, and DQN,



(a) slice 1.



(b) slice 2.



(c) slice 3.

Fig. 8: The cumulative stability probabilities of three slices. (a) slice 1, (b) slice 2, (c) slice 3.

they can improve the stability performance of the slice visibly. However, the performance of the DDPG, A2C, A3C and DQN algorithm may be slightly worse than that of the DB algorithm with the slice rate threshold gradually decreasing, such as slice 2 and slice 3. Relatively, the proposed LSTM-DDPG is always

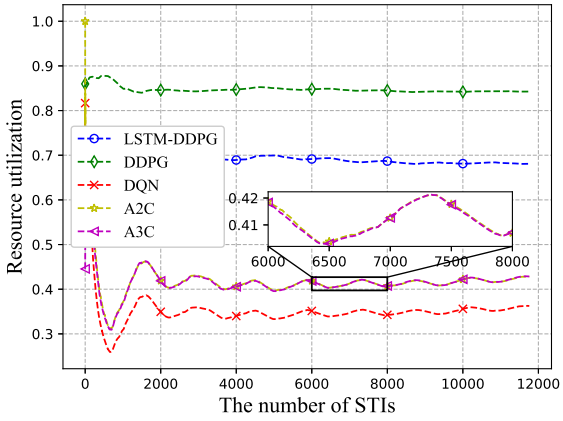


Fig. 9: Comparison of resource utilization.

better than other algorithms, because the proposed algorithm contains a forecast stage for average resource demands, which makes the agent has a better adaptation to the changes of RB demand in both long and short timescale. We can see that the cumulative stability probability of LSTM-DDPG can reach more than 92%, reached 92.7%, 99.3%, and 99.5% for three slices respectively. Compared to the traditional DQN algorithm, this indicator of the proposed LSTM-DDPG has increased by 27.8% on average.

Then, we compared the resource utilization of the LSTM-DDPG with other methods. The resource utilization (RU) is defined as $RU = \sum_t \sum_v \widetilde{h}_v^t / \sum_t \sum_v h_v^t$. To keep the performance stabilization in the slice, the system allocates resources in a redundant way, which leads to the resource utilization reduces mildly. As described in Fig. 9, we can see that the curve of A2C is very close to the curve of A3C, and the same phenomenon also appears in Fig. 8. In addition, it is clear that the RU of LSTM-DDPG can achieve 70% approximately while obtaining the maximum stability probability performance of slices, and the higher resource utilization of DDPG is achieved by the expenses of losing the stability performance. We aim to keep the performance of slices stable while maximizing the RU, which is also a tradeoff between the resource consumption and slice's performance.

As shown in Fig. 10, we evaluate the slice stability probability that the slice supplies a stable QoS to the served vehicles in STI with respect to the rate threshold of slice. It can be seen that the slice stability probability decreases with the growing of rate threshold. Apparently, the demand for resources of the vehicle would increase and fluctuate more drastically if the rate threshold becomes larger, such that the slice stability probability is reduced. Compared with the benchmarks, the proposed LSTM-DDPG shows a preferable performance adapting to different environments, and always achieves the maximal probability even in a rigorous setting. For example, when the threshold changes from 5M bit/s to 9M bit/s, the performance of DDPG drops sharply, the corresponding stability probability is changed from 96.3% to 70.9%. Conversely, the curve of LSTM-DDPG decreased gently, and the stability probability changed from 97.9% to

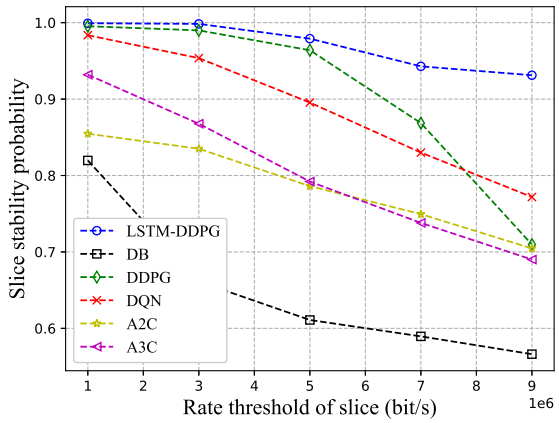


Fig. 10: Slice stability probability comparison with respect to the rate threshold of slice.

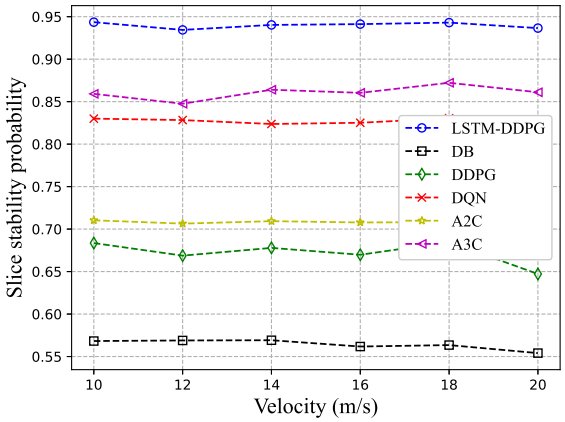


Fig. 11: Slice stability probability comparison with respect to the vehicle's velocity.

93.1%. In addition, the performances of five algorithms in the single slice scenario may be distinct from those in the multi-slice scenario, the underlying reason is that the resource allocation for all slices completed by an agent of which the states of all slices are considered, the slice's performance is easily affected by other slices subsequently.

In the last experiment, we compare the slice stability probability with respect to the vehicle's velocity. As shown in Fig 11, though the vehicle's velocity is becoming faster, the DB and LSTM-DDPG always get the worst and best results, respectively. As the velocity increases, all curves exhibit a slight downtrend, in which the curves of DDPG and A3C have obvious fluctuations. The increased vehicle speed means a more dynamic environment, which makes it more difficult for the system to predict the resources required for the slice. However, the proposed LSTM-DDPG algorithm enables tracking the traffic of the slice precisely and adapting to different vehicle speeds, so the stability probability can still reach 93.6% when the vehicle's speed is 20 m/s.

It can be seen from the above simulations that the stability performance of slice and the reduction of resource cost are guaranteed by the proposed LSTM-DDPG efficiently. In this orchestration, the DDPG is competent for fine-tuning the dedicated resources obtained by the LSTM so as to satisfy the dynamic requirements in slice. Besides, except for predicting the average resource requirements of the slice, the LSTM also contributes to adapt the parametric variation of the slice for the system, e.g., the variation of rate threshold in Fig 10, which makes the system has stronger robustness to the network environment.

VII. Conclusion

In this paper, we propose a QoS guaranteed network slicing orchestration for IoV to support vehicle different services with diversified QoS requirements. Specifically, The LSTM is used to predict the average resource demand of slice and then execute the dedicated resource allocation. Then, the DDPG is used for adjusting the allocated resource. The effectiveness and adaptiveness of the proposed orchestration have been demonstrated with simulation results. In the the future work, we will investigate the intra-slice resource allocation problem in IoV, in which the slices cost and vehicles QoS are highly coupled. In addition, as the IoV becomes more dynamic and large-scale, a distributed and low-complexity strategy is also needed to be presented.

References

[1] J. Zhang and K. B. Letaief, "Mobile Edge Intelligence and Computing for the Internet of Vehicles," *Proc. IEEE*, vol. 108, no. 2, pp. 246-261, Feb. 2020.

[2] L. Liang, H. Peng, G. Y. Li and X. Shen, "Vehicular Communications: A Physical Layer Perspective," *IEEE Trans. Veh. Technol.*, vol. 66, no. 12, pp. 10647-10659, Dec. 2017.

[3] H. Peng, Le Liang, X. Shen and G. Y. Li, "Vehicular Communications: A Network Layer Perspective," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1064-1078, Feb. 2019.

[4] K. Jiao, X. Li, M. Pan, *et al.*, "Trading Based Service-Oriented Spectrum-Aware RAN-Slicing Under Spectrum Sharing," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, 2020, pp. 1-6.

[5] J. Mei, X. Wang, and K. Zheng, "Intelligent Network Slicing for V2X Services Toward 5G," *IEEE Netw.*, vol. 33, no. 6, pp. 196-204, 2019.

[6] X. Zhou, R. Li, T. Chen and H. Zhang, "Network slicing as a service: enabling enterprises' own software-defined cellular networks," *IEEE Commun. Mag.*, vol. 54, no. 7, pp. 146-153, July 2016.

[7] X. Hou *et al.*, "Reliable Computation Offloading for Edge-Computing-Enabled Software-Defined IoV," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7097-7111, Aug. 2020.

[8] G. O. Boateng, D. Ayepah-Mensah, D. M. Doe, *et al.*, "Blockchain-Enabled Resource Trading and Deep Reinforcement Learning Based Autonomous RAN Slicing in 5G," *IEEE Trans. Netw. Service Manag.*, doi:10.1109/TNSM.2021.3124046.

[9] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, *et al.*, "Network Slicing for 5G with SDN/NFV: Concepts, Architectures, and Challenges," *IEEE Commun. Mag.*, vol. 55, no. 5, pp. 80-87, 2017.

[10] K. Samdanis, X. Costa-Perez, and V. Sciancalepore, "From network sharing to multi-tenancy: The 5G network slice broker," *IEEE Commun. Mag.*, vol. 54, no. 7, pp. 32-39, 2016.

[11] V. Sciancalepore, K. Samdanis, X. Costa-Perez, *et al.*, "Mobile traffic forecasting for maximizing 5G network slicing resource utilization," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Atlanta, GA, USA, May 2017, pp. 1-9.

[12] S. Zhang, "An Overview of Network Slicing for 5G," *IEEE Wireless Commun.*, vol. 26, no. 3, pp. 111-117, 2019.

[13] X. Shen *et al.*, "AI-Assisted Network-Slicing Based Next-Generation Wireless Networks," *IEEE Open J. Veh. Technol.*, vol. 1, pp. 45-66, 2020.

[14] J. Wang, C. Jiang, H. Zhang, Y. Ren, *et al.*, "Thirty Years of Machine Learning: The Road to Pareto-Optimal Wireless Networks," *IEEE Commun. Surv. Tut.*, vol. 22, no. 3, pp. 1472-1514, thirdquarter 2020.

[15] J. Wang, C. Jiang, K. Zhang, T. Q. S. Quek, *et al.*, "Vehicular Sensing Networks in a Smart City: Principles, Technologies and Applications," *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 122-132, February 2018.

[16] X. Cheng, Y. Wu, G. Min, *et al.*, "Safeguard Network Slicing in 5G: A Learning Augmented Optimization Approach," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 7, pp. 1600-1613, 2020.

[17] Y. Cui, X. Huang, P. He, *et al.*, "A Two-Timescale Resource Allocation Scheme in Vehicular Network Slicing," in *Proc. IEEE 93st Veh. Technol. Conf. (VTC-Spring)*, Helsinki, Finland, 2021, pp. 1-5.

[18] X. Foukas, G. Patounas, A. Elmokashfi, *et al.*, "Network Slicing in 5G: Survey and Challenges," *IEEE Commun. Mag.*, vol. 55, no. 5, pp. 94-100, 2017.

[19] N. Zhang, Y.-F. Liu, H. Farmanbar, *et al.*, "Network Slicing for Service-Oiented Networks Under Resource Constraints," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2512-2521, 2017.

[20] F. Wei, G. Feng, Y. Sun, *et al.*, "Dynamic Network Slice Reconfiguration by Exploiting Deep Reinforcement Learning," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Dublin, Ireland, 2020, pp. 1-6.

[21] W. Guan, H. Zhang and V. C. M. Leung, "Analysis of Traffic Performance on Network Slicing Using Complex Network Theory," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 15188-15199, Dec. 2020.

[22] M. Noor-A-Rahim, Z. Liu, H. Lee, *et al.*, "A survey on resource allocation in vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, early access, Sep. 4, 2020, doi:10.1109/TITS. 2020.3019322.

[23] N. V. Huynh, D. T. Hoang, D. N. Nguyen, *et al.*, "Optimal and Fast Real-Time Resource Slicing With Deep Dueling Neural Networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1455-1470, 2019.

[24] Y. Sun, G. Feng, L. Zhang, *et al.*, "User Access Control and Bandwidth Allocation for Slice-Based 5G-and-Beyond Radio Access Networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, China, May 2019, pp. 1-6.

[25] Y. Sun, G. Feng, L. Zhang, *et al.*, "Distributed Learning Based Handoff Mechanism for Radio Access Network Slicing with Data Sharing," in *Proc. IEEE Int. Conf. Commun. (ICC)*, China, May 2019, pp. 1-6.

[26] Y. Azimi, S. Yousefi, H. Kalbkhani and T. Kunz, "Energy-Efficient Deep Reinforcement Learning Assisted Resource Allocation for 5G-RAN Slicing," *IEEE Trans. Veh. Technol.*, doi:10.1109/TVT.2021.3128513.

[27] C. Qi, Y. Hua, R. Li, *et al.*, "Deep Reinforcement Learning With Discrete Normalized Advantage Functions for Resource Management in Network Slicing," *IEEE Commun. Lett.*, vol. 23, no. 8, pp. 1337-1341, 2019.

[28] Y. Hua, R. Li, Z. Zhao, *et al.*, "GAN-Powered Deep Distributional Reinforcement Learning for Resource Management in Network Slicing," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 334-349, 2020.

[29] M. Yan, G. Feng, J. Zhou, *et al.*, "Intelligent Resource Scheduling for 5G Radio Access Network Slicing," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7691-7703, 2019.

[30] R. Li, C. Wang, Z. Zhao, *et al.*, "The LSTM-Based Advantage Actor-Critic Learning for Resource Management in Network Slicing With User Mobility," *IEEE Commun. Lett.*, vol. 24, no. 9, pp. 2005-2009, 2020.

[31] X. Ge, "Ultra-Reliable Low-Latency Communications in Autonomous Vehicular Networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5005-5016, 2019.

[32] H. D. R. Albonda and J. Prez-Romero, "An Efficient RAN Slicing Strategy for a Heterogeneous Network With eMBB and V2X Services," *IEEE Access*, vol. 7, pp. 44771-44782, 2019.

[33] K. Xiong, S. Leng, J. Hu, *et al.*, "Smart Network Slicing for Vehicular Fog-RANs," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3075-3085, April 2019.

[34] Z. Mlika and S. Cherkaoui, "Network Slicing with MEC and Deep Reinforcement Learning for the Internet of Vehicles," *IEEE Netw.*, vol. 35, no. 3, pp. 132-138, May/June 2021.

[35] W. Wu *et al.*, "Dynamic RAN Slicing for Service-Oriented Vehicular Networks via Constrained Learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 2076-2089, July 2021.

[36] J. Zheng, P. Caballero, G. de Veciana, *et al.*, "Statistical Multiplexing and Traffic Shaping Games for Network Slicing," *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2528-2541, 2018.

[37] L. Liang, H. Ye, G. Yu, *et al.*, "Deep-Learning-Based Wireless Resource Allocation With Application to Vehicular Networks," *Proc. IEEE*, vol. 108, no. 2, pp. 341-356, 2020.

[38] G. Wang, G. Feng, T. Q. S. Quek, *et al.*, "Reconfiguration in Network SlicingłOptimizing the Profit and Performance," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 2, pp. 591-605, 2019.

[39] M. Yan, G. Feng, J. Zhou,*et al.*, "Intelligent Resource Scheduling for 5G Radio Access Network Slicing," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7691-7703, 2019.

[40] C. Zhang, H. Zhang, J. Qiao, *et al.*, "Deep Transfer Learning for Intelligent Cellular Traffic Prediction Based on Cross-Domain Big Data," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1389-1401, 2019.

[41] H. Zhang and V. W. S. Wong, "A Two-Timescale Approach for Network Slicing in C-RAN," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 6656-6669, 2020.

[42] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," in *Proc. 9th Int. Conf. Artificial Neural Networks*, pp. 850-855, 1999.

[43] L. Wang, H. Ye, L. Liang and G. Y. Li, "Learn to Compress CSI and Allocate Resources in Vehicular Networks," *IEEE Trans. Commun.*, vol. 68, no. 6, pp. 3640-3653, June 2020.

[44] E. Sakic, F. Sardis, J. W. Guck and W. Kellerer, "Towards adaptive state consistency in distributed SDN control plane," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, 2017, pp. 1-7.

[45] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, Cambridge, MA, USA: MIT Press, 1998.

[46] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529-533, Feb. 2015.

[47] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proc. 13th AAAI Conf. Artif. Intell.*, Feb. 2016, pp. 2094-2100.

[48] Z. Wang, N. de Freitas, and M. Lanctot, "Dueling network architectures for deep reinforcement learning," in *Proc. Int. Conf. Learning Representations*, 2016.

[49] T. P. Lillicrap, J. J. Hunt, A. Pritzel, *et al.*, "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learning Representations*, 2016.

[50] R. S. Sutton, D. A. McAllester, S. P. Singh, *et al.*, "Policy gradient methods for reinforcement learning with function approximation," *in Advances in Neural Information Processing systems*, 2000.

[51] C. Qiu, Y. Hu, Y. Chen and B. Zeng, "Deep Deterministic Policy Gradient (DDPG)-Based Energy Harvesting Wireless Communications," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8577-8588, Oct. 2019.

[52] G. Barlacchi *et al.*, "A multi-source dataset of urban life in the city of Milan and the Province of Trentino," *Scientific Data*, vol. 2, no. 1, p. 150055, 2015.

[53] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Study on LTE-based V2X Services; (Release 14), 3GPP TR 36.885 V14.0.0, Jun. 2016.

[54] L. Liang, H. Ye, and G. Y. Li, "Spectrum Sharing in Vehicular Networks Based on Multi-Agent Reinforcement Learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2282-2292, 2019.

Yaping Cui received the Ph.D. degree in traffic information engineering and control from Southwest Jiaotong University, Chengdu, China, in 2017. He joined the School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing, China, in 2017, where he is currently a Lecturer. His research interests include millimetre-wave communications, multiple antenna technologies, and smart antennas for vehicular networks.

Xinyun Huang is currently pursuing the M.S. degree with the School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing, China. His research interests include vehicular networks and network slicing.

Peng He received the Ph.D. degree from the University of Electronic Science and Technology of China, in 2018. He is currently a Lecturer with the School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications. His research interests include mobile edge computing, molecular communication, wireless body area networks. He is a member of IEEE ComSoc.

Dapeng Wu (Senior Member, IEEE) received the M.S. degree in communication and information systems from the Chongqing University of Posts and Telecommunications in June 2006, and the Ph.D. degree from the Beijing University of Posts and Telecommunications in 2009. His research interests include ubiquitous networks, IP QoS architecture, network reliability, and performance evaluation in communication systems.

Ruyan Wang received the Ph.D. degree from the University of Electronic and Science Technology of China (UESTC) in 2007, and the M.S. degree from the Chongqing University of Posts and Telecommunications (CQUPT), China, in 1997. Since December 2002, he has been a Professor with the Special Research Centre for Optical Internet and Wireless Information Networks, CQUPT. His research interests include network performance analysis and multimedia information processing.