IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 2, 2020, accepted September 8, 2020, date of publication September 16, 2020,
date of current version October 1, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3023939

Task Offloading Strategy Based on Reinforcement Learning Computing in Edge Computing Architecture of Internet of Vehicles

KUN WANG[1], XIAOFENG WANG[1], XUAN LIU[2], AND ALIREZA JOLFAEI[3], (Senior Member, IEEE)

[1]College of Physics and Electronic Engineering, Shanxi University, Taiyuan 030006, China
[2]State Grid Shanxi Electric Power Company Maintenance Branch, Taiyuan 030000, China
[3]Department of Computing, Macquarie University, Sydney, NSW 2109, Australia

Corresponding author: Kun Wang (eekunwang@126.com)

This work was supported by the Scientific and Technological Innovation Programs of Higher Education Institutions in Shanxi (STIP) under Grant 201802004.

ABSTRACT With the rapid increase of vehicles, the explosive growth of data flow and the increasing shortage of spectrum resources, the performance of existing task offloading scheme is poor, and the on-board terminal can't achieve efficient computing. Therefore, this article proposes a task offload strategy based on reinforcement learning computing in edge computing architecture of Internet of vehicles. Firstly, the system architecture of Internet of vehicles is designed. The Road Side Unit receives the vehicle data in community and transmits it to Mobile Edge Computing server for data analysis, while the control center collects all vehicle information. Then, the calculation model, communication model, interference model and privacy issues are constructed to ensure the rationality of task offloading in Internet of vehicles. Finally, the user cost function is minimized as objective function, and double-layer deep Q-network in deep reinforcement learning algorithm is used to solve the problem for real-time change of network state caused by user movement. The results show that the proposed offloading strategy can achieve fast convergence. Besides, the impact of user number, vehicle speed and MEC computing power on user cost is the least compared with other offloading schemes. The task offloading rate of our proposed strategy is the highest with better performance, which is more suitable for the scenario of Internet of vehicles.

INDEX TERMS Internet of Vehicles, mobile edge computing, task offloading, reinforcement learning, privacy security.

I. INTRODUCTION

With the development of automobile industry and the improvement of economic level, the number of automobiles is increasing. This has caused serious traffic jams and frequent traffic accidents. Thus, people's demands for car safety and driving comfort are becoming more and more urgent [1]. The Internet of Vehicles (IoV) came into being. It has become a research hotspot that the government, research institutions and vehicle manufacturing companies pay attention to together [2], [3]. One of the most important meanings of IoV is that messages exchanged between vehicles can help improve road safety.

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Tariq.

IoV technology is a combination of information technology and transportation. The vehicle is equipped with an on-board equipment unit with wireless communication, perception and computing capabilities. Vehicles can communicate through wireless transmission [4]. In addition, the vehicle can also communicate with an RSU. RSU has the function of providing routing and access network for vehicles. At present, there are mainly two communication protocols used in IoV: dedicated short-range communication and cellular network-based wireless communication protocols for vehicles. Cellular networks can provide vehicles with a wide range of communications. But the current real-time information exchange efficiency is lower than that of dedicated short-range communication [5].

With the rapid development of in-vehicle equipment and the increasing amount of data, in-vehicle applications have

also generated computing needs, such as applications such as real-time road conditions and automatic identification. These in-vehicle applications require a lot of computing and storage resources. And many in-vehicle applications are sensitive to delay, such as application services such as autonomous driving and driving safety enhancement. These applications require a lot of computing resources and have very strict requirements on time delay. The existing in-vehicle equipment cannot meet these conditions. This brings huge challenges to IoV in terms of computing and communication capabilities [6].

Cloud computing can make up for the lack of computing resources of on-board equipment. However, due to the huge amount of data transmitted, there will be a large transmission delay from vehicle-mounted equipment to core networks. This cannot meet the delay requirements of some in-vehicle services [7]. To this end, MEC technology is introduced into IoV. It migrates computing and storage capabilities to the scope of access network, and can provide low-latency, high-bandwidth and real-time network services [8]. Specifically, the computing tasks of vehicles can be offloaded to MEC server on RSU, saving a lot of transmission delay and energy consumption of vehicles. Moreover, the MEC server has rich computing and storage resources, which can greatly accelerate the execution speed of computing tasks and solve the problem of insufficient computing resources of vehicle itself [9]. However, the complex network scenario of IoV also brings many problems to the application of MEC technology. The high-speed mobility of vehicles and the diversity of offloading methods in IoV make the computing task offloading strategy more complicated [10]. The movement of vehicles will cause the communication parameters to change during offloading process, and the impact of cell handover on offloading needs to be considered. The coexistence of multiple communication modes in IoV makes there are multiple options for offloading in IoV. Therefore, how to use a reasonable offloading scheme to efficiently complete the computing task is a problem worthy of in-depth study.

II. RELATED WORK

In the MEC-based of IoV, due to the limited computing resources of vehicle itself, in most cases it is insufficient to support the needs of various computing tasks. Therefore, offloading computing tasks by vehicles is an effective solution to this problem. Due to the abundant storage and computing resources on MEC server, the vehicle can offload its own computing tasks to nearby RSUs where MEC server is deployed through Vehicle to Infrastructure (V2I) wireless communication. Then the computing task can be calculated on MEC server. The total delay of V2I computing task offloading consists of several parts. The uploading delay of computing tasks from vehicles to MEC server will occur. After receiving computing tasks, the MEC server performs corresponding calculation processing on it, which will cause calculation execution time delay. When the calculation is completed, calculation results need to be returned to original vehicles, and there will be a delay. The traditional V2I offloading is to transfer the entire computing task to MEC server for calculation. The continuous connection of vehicles with the remote will consume a lot of energy, and the upload of computing tasks and the transmission of calculation results will also bring a certain delay. The goal of reference [11] was to minimize the weighted sum of energy consumption and delay. Each user had multiple tasks, which was more comprehensive. Reference [12] used game theory to solve optimization problem and proves the existence of Nash equilibrium. Reference [13] calculated the theoretical upper limit of server task processing, and proved that its algorithm can be very close to theoretical value. It transformed the non-convex quadratic function under quadratic constraint condition into a separable semi-definite programming problem by relaxation techniques. Reference [14] proposed a compromise solution. A part of tasks can be processed locally and then offloaded to cloud to execute the remaining part. Reference [15] proposed a collaboration method based on MEC and cloud computing. This method offloaded services in vehicle network to the car. By joint optimization of computing offloading decision-making and computing resource allocation, the problem of cloud-MEC collaborative computing offloading was proposed. The simulation results showed that the algorithm can effectively improve the practicability and calculation time of system, especially for the case where MEC server cannot meet the demand due to insufficient computing resources.

In order to minimize the energy consumption of smart devices, it was necessary to jointly optimize offloading options for wireless resource allocation and computing resource allocation [16]. Reference [17] proposed a random mixed integer nonlinear programming problem. This problem was based on joint optimization of task distribution decision-making, flexible computing resource scheduling and radio resource allocation. To solve this problem, Lyapunov optimization theory was introduced to decompose the original problem into four separate sub-problems. These sub-problems were solved by convex decomposition method and matching game, theoretically analyzed the trade-off between energy efficiency and service delay. The simulation results also verify the superiority of proposed task sharing and resource allocation scheme in C-RAN. Reference [13] proposed a new architecture. This architecture can dynamically orchestrate edge computing and caching resources by making full use of AI-based algorithms, thereby improving system practicability. And it developed a joint edge computing and caching scheme to maximize the practicability of system. They developed a novel resource management scheme by using deep reinforcement learning. Numerical results proved the effectiveness of this scheme. Reference [18] transformed the calculation of shunt formula into an optimization problem to minimize the cost of shunting. At the same time it provided performance guarantee. On the basis of random optimization, a Dynamic Computation Offloading Algorithm (DCOA) was proposed, which decomposed the optimization problem into a series of sub-problems, and solved these sub-problems in

parallel in an online and distributed manner. Experimental evaluation showed that DCOA can be a trade-off between offloading cost and performance.

The vehicle itself has certain computing resources. Thus, Vehicle to Vehicle (V2V) communication method can be used to offload computing tasks to other idle vehicles. Performing task calculations on idle vehicles was equivalent to treating those vehicles with idle resources as edge devices similar to MEC servers [19]. The computing resources of vehicles are very limited compared to MEC server. Therefore, the V2V offloading method is suitable for small computing tasks, or sharing a small part of large computing tasks. The V2V offloading method can improve the utilization of computing resources, make reasonable use of idle computing resources, and also reduce the pressure on RSU. Reference [20] proposed an event-triggered dynamic task allocation framework based on linear programming optimization and binary particle swarm optimization. In order to evaluate the effectiveness of Folo, the mobility of fog nodes at different times of the day was simulated based on realistic taxi trajectories. Two representative tasks were performed, including video streaming and real-time object recognition.

The V2V computing task offloading process includes three processes: the transmission of computing tasks, the execution of calculation and the return of calculation result. The transmission of computing tasks and the return of calculation results both use V2V communication mode [21]. V2V communication is to establish a mobile network between moving vehicles, turning each participating vehicle into a wireless router or node. The general V2V communication range is about 100 meters to 300 meters. Vehicles within the V2V communication range can be connected to each other and create a larger VANET network. Reference [22] proposed a context-aware communication method to use edge computing technology to effectively integrate different licensed and unlicensed spectrum. At the same time, it effectively combined route aggregation, data caching, and decentralized calculation methods to compensate for limited wireless resources. In order to improve the routing performance of multi-hop broadcast and multi-hop unicast communication, asynchronous multi-hop broadcast and asynchronous multi-hop unicast schemes are introduced.

With the popularization of 5G networks, the data in IoV has increased sharply, and computing resources have become increasingly tight. The above offloading strategies and communication methods can no longer fully adapt to the development of modern IoV. To this end, a task offloading strategy based on reinforcement learning computing in the edge computing architecture of IoV is proposed. The innovations are summarized as follows:

1) In order to improve offloading efficiency, the IoV system architecture is designed, and the IoV is divided into various communities. The RSU of cells receives internal vehicle information and transmits it to MEC server for data analysis. The control center gathers all vehicle information and makes full use of local and MEC computing resources of vehicles.

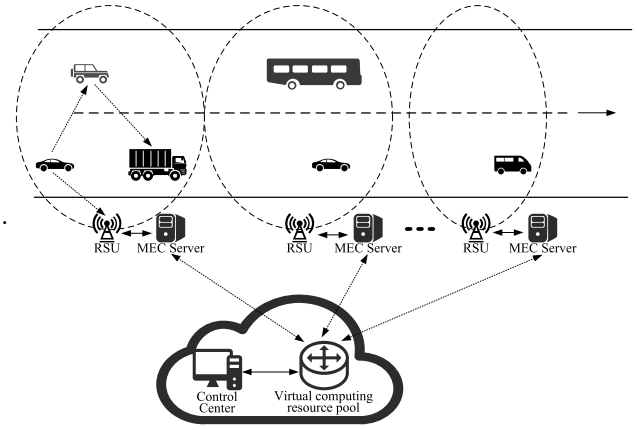

FIGURE 1. Overall framework of IoV system.

2) Due to the complex structure of IoV, the proposed task offloading strategy constructs a calculation model, a communication model, an interference mode and a privacy protection model. On this basis, the user cost function is minimized as objective function. And we use deep reinforcement learning algorithm to solve it in order to achieve efficient task offloading.

2) Considering that the mobility of IoV users may lead to changes in cells where the user is located. And with the increase of users, action space will also increase exponentially. Thus, our proposed strategy adopts Double Deep Q-Network (Double DQN) to solve task offloading problem to overcome the real-time change of network status caused by user movement, which improves the convergence of this strategy.

III. SYSTEM MODEL AND PROBLEM MODELING

A. SYSTEM MODEL

The mentioned IoV system scenario is a one-way straight road with RSUs deployed along one side of roads, as shown in Fig.1. It is assumed that the roadside base station is a micro cell with a coverage radius of about 200 meters. The radius of micro cells is represented by r, and the vertical distance from the center of corresponding cell to the road is represented by d. Each RSU is equipped with a MEC server. Moving vehicles may transfer their computing tasks to the RSU closest to them. MEC server provides complex computing services for computing tasks transmitted to RSU [23]. There is a control center in the access network to obtain information about base stations and MEC servers in each cell. Vehicles can obtain information about the cell in front of it through this control center.

There may be many vehicles in a cell within its communication coverage area. For convenience, use v to identify vehicle v. The vehicle that has a computing task to be offloaded is designated as the source vehicle, which is represented by symbol v. During V2V offloading process, the computing task may go through multiple hops from v and finally complete the computing service on a certain vehicle. The vehicle that completes computing tasks is set as the target vehicle, which is represented by symbol v.

Through the known RSU coverage radius v and vertical distance v from the center of cells to the road, total distance v traveled by vehicles in the coverage of a single cell can be calculated:

$$s = 2\sqrt{r^2 - d^2} \tag{1}$$

The dwell time v of vehicles within the coverage of a single RSU can be expressed as:

$$t_b = s/v_s \tag{2}$$

In the formula, parameter v represents the speed of source vehicle v. In the set scene, the vehicles each travel at a constant speed.

B. CALCULATION MODEL

1) LOCAL CALCULATION MODE

If IoV user v chooses to perform task v locally, define v as the user's local computing power. Therefore, local calculation time consumes v:

$$t_v^l = \sum_{m=1}^{M} \frac{\left(1 - d_{v,m}\right) b_v}{c_v^l}, \quad \forall v \in V \tag{3}$$

In the formula, v is the path loss calculated locally by vehicles, and v is the amount of data transmitted.

For local calculations, in order to calculate the energy consumption of connected car users, the energy consumption v per revolution of computer CPU is used. v represents the energy consumption coefficient, and v represents the frequency of CPU. Therefore, the locally calculated energy consumption v:

$$e_v^l = \sum_{m=1}^{M} \left(1 - d_{v,m}\right) \kappa \left(f_v^l\right)^2 b_v \tag{4}$$

Thus, the cost function v of vehicle network user v on the local computer can be expressed as:

$$C_v^l = t_v^l + e_v^l \tag{5}$$

2) TASK OFFLOADING MODE

The task offloading mode mainly includes three stages: task transmission stage, task calculation stage and task result return stage [24]. In the task transmission stage, according to the system model, IoV users multiplex cellular channel to transmit tasks to MEC server. Co-channel interference in a small area needs to be considered. Since multi-cell scenarios are considered, users in adjacent cells will also cause inter-cell interference. Define the transmission power allocation strategy of IoV users as v. According to Shannon's theorem, the expression of task transmission rate v of vehicle network user v can be derived:

$$v_t = B \log\left(1 + \frac{p_v \gamma_v}{\sum_{i=1, i \neq v}^{V} \varphi_i p_i \gamma_i + p_u \gamma_u + \rho^2}\right) \tag{6}$$

In the formula, v represents the transmission power of vehicle network user v, and v represents the channel gain of uplink transmission channel. v represents whether user v and user v share a channel, if they share v, otherwise v. v represents the transmission power of other IoV users, and v represents the channel gain of other users' uplink transmission channels. v represents the transmit power of cellular user, and v represents the channel gain of uplink channel of the cellular user. v represents the power spectral density of white noise.

The IoV channel model uses Rayleigh fading model. Since the mobility of vehicles will greatly affect the distance between senders and receivers, the channel gain v can be derived by estimating the change between senders and receivers by speed.

$$\left|\gamma_{i,j}\right|^2 = G \cdot \left|d_{i,j} + v_{i,j} \cdot t_w\right|^{-a} \cdot \left|\gamma_0\right|^2 \tag{7}$$

In the formula, v is the antenna gain, v is the distance between senders and receivers, and v is the relative speed between senders and receivers. v is the time interval between data preparation and actual antenna transmission, and v represents Rayleigh fading.

Then the task transmission process time consumption v:

$$t_v^t = \sum_{m=1}^{M} \frac{d_{v,m} b_v}{v_t}, \quad \forall v \in V \tag{8}$$

For task calculation phase, MEC server on each base station side can provide computing offloading services to multiple users at the same time [25]. After receiving the offloaded task, the server will perform computing tasks and transmit calculation results to users after the calculation is completed. The computing resource allocation strategy is defined as v, where v represents the computing power allocated after user v is offloaded to MEC server m. Therefore, a feasible computing resource allocation strategy must satisfy the computing resource constraints:

$$\sum_{v=1}^{V} c_{m,v} \leq c_m, \forall m \in M \tag{9}$$

In the formula, v represents the remaining computing power of MEC server. Using the computing resources allocated by MEC server, the time consumption v for MEC to execute the machine loss task can be calculated:

$$t_v^e = \sum_{m=1}^{M} \frac{d_{v,m} b_v}{c_{m,v}}, \quad \forall v \in V \tag{10}$$

For the task calculation result return stage, the downlink transmission rate is generally higher. Secondly, the calculation result is generally much smaller than uploaded task data. Therefore, the delay at this stage is negligible. Therefore, the time consumption v and energy consumption v of the task offloading phase of task v of connected car user can be expressed as:

$$\begin{aligned} T_v^u &= t_v^t + t_v^e \\ E_v^u &= p_v \left(t_v^u + t_v^e\right) \end{aligned} \tag{11}$$

At the same time, the cost function C_v^u of IoV user v in task offloading stage can be derived:

$$C_v^u = T_v^u + E_v^u \tag{12}$$

Through the derivation of previous task model, the total cost function υ for local computing mode car users and task offloading computing mode users is expressed as follows:

$$C_s = \sum_{v=1}^{V} \left(C_v^l + C_v^u \right) \tag{13}$$

C. COMMUNICATION MODEL

The communication between vehicles and RSU is carried out through LTE-Advanced direct link. The upload link from vehicles to RSUs is set as a frequency-flattened block fading Rayleigh channel [26]. The path loss between vehicles and RSUs can be represented by the model υ. The parameter υ here is the distance between vehicle υ and the center of the coverage area of RSUs, and the parameter υ is the path loss factor. In addition, the channel fading factor of upload link is represented by the symbol d_1, and the Gaussian white noise power is represented by symbol d_1. According to Shannon's formula, the data transmission rate of upload link can be calculated as:

$$v_{VR} = B_{VR} \log_2 \left(1 + \frac{P_t d_1^{-v} \tau^2}{N_0} \right) \tag{14}$$

In the formula, d_1 represents the bandwidth of upload channel, and the parameter d_1 represents the transmission power of vehicle-mounted device.

In the set scene, all vehicles travel at a constant speed. The speed of vehicle d_1 is represented by symbol d_1. The mobility of vehicles will cause the distance d_1 between vehicles and the center of the coverage area of RSUs to change with time, and the change rule is expressed as:

$$d_1(t) = \sqrt{l^2 + \left(\frac{s}{2} - v_a^s t \right)^2} \tag{15}$$

In the formula, v_a^s is the driving speed of source vehicle V_s.

Therefore, the data rate v_{VR} of upload link changes over time. This change is caused by the mobility of vehicle and can be expressed as $v_{VR}(t)$. The vehicle stays within the coverage of connected RSU. The average upload rate of t_b during this period can be expressed as:

$$\overline{v_{VR}} = \frac{\int_0^{t_b} v_{VR}(t)\, dt}{t_b} \tag{16}$$

In order to simplify the problem, the average upload rate represents data transmission rate at which source vehicle offloads computing tasks to MEC server in the cell during offloading process.

In addition, in V2V communication network, the communication between V2V uses IEEE802.11p protocol in the DSRC communication method. The maximum communication range is indicated by symbol d_r. The communication channel between vehicles uses a simple independent and identically distributed channel, and the path loss can be defined as:

$$\begin{aligned} l_{V2V}^{dB} &= 63.3 + 17.7 \log_{10}(d_{i,j}) \\ 0 &\le d_{i,j} \le d_r \end{aligned} \tag{17}$$

In the formula, parameter $d_{i,j}$ represents the communication distance between any vehicle V_i and V_j.

Since the vehicles in the research scene travel at a constant speed at their respective speeds, different speed values between vehicles will produce a relative speed between vehicles. The relative speed between vehicles V_i and V_j is represented by symbol $v_{re}^{i,j}$. In order to simplify the research problem, it is assumed that the relative vehicle speed VM has a very small effect on the path loss value, which can be ignored [27].

In the scene, V2V communication uses orthogonal frequencies to reduce the mutual influence between vehicles. The communication bandwidth of V2V is denoted by B_{V2V}. Therefore, the data transmission rate between any two communicable vehicles V_i and V_j can be expressed as:

$$v_{V2V}^{i,j} = B_{V2V} \log_2 \left(1 + \frac{P_t l_{V2V} \tau^2}{N_0} \right) \tag{18}$$

D. INTERFERENCE MODEL

Communication resources refer to the time-frequency blocks required to transmit signals. In order to reduce the complexity of communication resource allocation, the communication resources used in V2V communication are isolated from common user equipment. We use *pair* to represent a V2V communication pair. Assume that different V2V resources are independent of each other. In order to utilize communication resources as much as possible, V2V communication allows spatial multiplexing. That is, different *pair* use the same V2V communication resources to communicate. At the same time, it is assumed that the unit communication resource can meet V2V data transmission demand in the edge computing of vehicles. Thus, a V2V communication resource needs to be allocated to each *pair*. But communication resources can be allocated repeatedly [28]. If two *pair* use the same V2V communication resource, interference will occur between them. In such scenarios, due to the mutual interference of multiple V2V communications, the Signal to Interference plus Noise Ratio (SINR) is a decisive factor affecting the quality of V2V communications. It is equal to the effective received signal power divided by interference signal power plus the noise power, expressed as follows:

$$\vartheta = \frac{S}{I + N_0} \tag{19}$$

In the formula, S represents the received signal power, I represents the interference signal power, and N_0 represents the noise power.

For a V2V communication pair $pair_s$, use d_{ss} to represent the distance between its sender and receiver. d_{ss} represents the V2V communication channel, P_t represents the transmission power, and α represents the path loss index. Then the received signal power S_s of $pair_s$ is:

$$\begin{aligned} S_s &= P_t d_{ss}^{-\alpha} H_{ss} \\ H_{ss} &= |h_{ss}|^2 \end{aligned} \tag{20}$$

For $pair_s$ and $pair_d$, d_{sd} is used to indicate the distance between the sender of $pair_d$ and the receiver of $pair_s$. Assuming that the set of $pair_s$ that use the same communication resources as $pair$ is Θ_s, the interference I_s at $I_s = \sum_{d\in\Theta_s} P_t d_{sd}^{-\alpha} H_{sd}$ is:

$$I_s = \sum_{d\in\Theta_s} P_t d_{sd}^{-\alpha} H_{sd} \tag{21}$$

Then the SINR of $pair_s$ is:

$$\vartheta_s = \frac{P_t d_{ss}^{-\alpha} H_{ss}}{\sum_{d\in\Theta_s} P_t d_{sd}^{-\alpha} H_{sd} + N_0} \tag{22}$$

In order to ensure the success of V2V communication, the SINR of received signal must be greater than or equal to a threshold V_i.

E. CONDITIONAL PRIVACY PROTECTION OF VEHICLE IDENTITY

In the process of communicating with other entities of IoV, vehicle V_i does not use its real identity, but its pseudonym $PID_i = \left\{PID_i^1, PID_i^2\right\}$. Among them $PID_i^1 = r_i \cdot P$, $PID_i^2 = RID \oplus h\left(r_i \cdot P_{pud}\right)$. According to the discrete logarithm problem, other vehicles cannot obtain the private key r_i of vehicle V_i when PID_i^1 and P are known. At the same time, because r_i is stored in the vehicle security manager, even vehicle V_i itself cannot leak r_i. Therefore, for vehicles other than pseudonymous owner, the true identity corresponding to pseudonym cannot be obtained based on the pseudonym received and known public information [29]. In order to be able to hold a malicious vehicle accountable, it is necessary to have the ability to trace the true identity of vehicles. If you need to trace the true identity of vehicles, first find the corresponding pseudonym $PID_i = \left\{PID_i^1, PID_i^2\right\}$ in the message sent by V_i. Get r_i true identity according to the equation:

$$\begin{aligned} &PID_i^2 \oplus h\left(s \cdot PID_i^1\right) \\ &= RID \oplus h\left(r_i \cdot P_{pub}\right) \oplus h\left(s \cdot r_i \cdot P\right) \\ &= RID \oplus h\left(r_i \cdot P_{pub}\right) \oplus h\left(r_i \cdot P_{pub}\right) = RID \end{aligned}$$

In order to ensure the timeliness of messages, after receiving a message, the message receiver first checks whether the message has expired [30]. Let t_r denote the time when the message was received, and t denote timestamp contained in the message. Δt_1 represents the time difference between the message sending vehicle clock and system clock, and Δt_2 represents the estimated network delay. If $|t_r - t| < \Delta t_1 + \Delta t_2$ is satisfied, then the receiver receives this message. Otherwise, the message is expired and receiver rejects the message.

IV. RESOURCE ALLOCATION BASED ON DEEP REINFORCEMENT LEARING

A. DEEP REINFORCEMENT LEARING THEORY

Reinforcement learning is a kind of machine learning. Agents use reinforcement learning to find an effective strategy when solving sequential decision problems. This strategy determines how the Agent should make the best choice in each state it may encounter. Unlike supervised learning, the agent cannot determine whether the behavior is correct or not. Instead, a reward signal represented by a value is used. The current behavior of the agent not only affects the instantaneous reward it receives, but also affects the next state or future reward [31]. Therefore, the strategy sought by a reinforcement learning agent is not to maximize instant rewards, but to maximize the sum of rewards over a period of time. Reinforcement learning is an important learning tool. In many cases, acting on the Agent to make it more adaptive (such as robot control, system optimization, etc.). Generally speaking, if the ultimate goal of reinforcement learning is provided, the agent can automatically learn how to change itself to the greatest extent to achieve the ultimate goal [32].

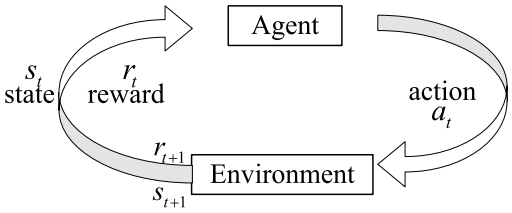

FIGURE 2. The framework of reinforcement learning.

The standard reinforcement learning framework is that the agent continuously interacts with its environment in discrete time. It is mainly composed of four elements: reward and punishment feedback function, value function, strategy selection and interactive environment, as shown in Fig.2.

At each time t, the current environment state $s_t \in S$ perceived by the Agent. Where S is the sum of all possible states, and then take some action $a_t \in A$, where A is the set of all possible actions. The environment will feed back a reward signal $r_{t+1} \in R$ and a new arrival state s_{t+1}. It is usually assumed that the state space is decomposable, namely $s_i = (x_1, x_2, \ldots, x_n) \in R^n$. At the same time, it is also assumed that the environment satisfies the Markov property, that is, a given state. The reward is only determined by the previous actions and status, which is:

$$\begin{aligned} &Pr\left\{s_{t+1}=s', r_{t+1}=r \,|s_t, a_t, r_t, s_{t-1}, a_{t-1}, \ldots, r_1, s_0, a_0\right\} \\ &= Pr\left\{s_{t+1} = s', r_{t+1}=r \,|s_t, a_t\right\} \end{aligned} \tag{23}$$

Markov Decision Processes (MDP) refers to the reinforcement learning process that satisfies the Markov property. It can be described as a four-tuple (S, A, P, R), $P : S \times A \times S \to [0, 1]$, representing the probability of arrival between states:

$$P\left(s, a, s'\right) = Pr = \left\{s_{t+1} = s' \,|s_t = s, a_t = a\right\} \tag{24}$$

$R : S \times A \times S \to R$ is the designated expected instant reward:

$$R\left(s, a, s'\right) = E\left\{r_{t+1} \,|s_t = s, a_t = a, s_{t+1} = s'\right\} \tag{25}$$

Agent's goal is to maximize the sum of future discount rewards over a period of time. Time t is $\sum_{k=0}^{\infty} \nu^k r_{t+k+1}$, where $\nu \in [0, 1]$ is the discount parameter. In order to maximize this formula, a strategy $\pi : S \mapsto A$ must be learned, and $\pi(s)$ specifies the action selected in state s [33].

Each strategy has an associated state value function $V^{\pi}: S \mapsto R$. Each strategy also has an action value function $Q^{\pi}: S \times A \mapsto R$, which specifies the expected long-term discount reward obtained after the a action and the strategy π are selected in the state s:

$$Q^{\pi}(s,a) = E\left\{\sum_{k=0}^{\infty} v^{k} r_{t+k+1} \,|\pi, s_t = s, a_t = a\right\} \quad (26)$$

There is an optimal action value function $Q^{*}(s,a) = \max_{\pi} Q^{\pi}(s,a)$, and there is an optimal strategy π^{*} such that:

$$\begin{aligned}\pi^{*}(s) &= \arg\max_{a} Q^{*}(s,a) \\ &= \text{argmax}_{a} \sum_{s'} P\left(s,a,s'\right)\left[R\left(s,a,s'\right) + vV^{*}\left(s'\right)\right]\end{aligned} \quad (27)$$

The goal of reinforcement learning is to find the optimal π^{*} or the optimal π^{*} after all. As a commonly used reinforcement learning method, Q-learning algorithm is characterized by time series learning of offline strategies. The Q-learning algorithm uses the state-action reward value and $Q^{*}(s,a)$ as the estimation function during the iterative update, instead of the $V(s)$ and state rewards in the temporal difference algorithm [34]. To ensure convergence, each action must be traversed in each iteration of the calculation. The theoretical calculation is:

$$\begin{aligned}&Q^{*}(s,a) \\ &= v\sum_{s\in S'} P\left(s,a,s'\right)\left(r\left(s,a,s'\right) + \max_{a'} Q^{*}\left(s',a'\right)\right) \quad (28)\\ &Q\left(s_t,a_t\right) \\ &= Q\left(s_t,a_t\right) + \alpha\left(r_{t+1} + v\max_{a} Q\left(s_{t+1},a\right) - Q\left(s_t,a_t\right)\right)\end{aligned} \quad (29)$$

Among them, $Q^{*}(s,a)$ is the sum of the maximum reward discounts that can be obtained by taking action a in state s. It can be concluded that choosing the action that maximizes the value of Q in state s will be the optimal strategy.

The Q-learning algorithm is an effective algorithm that is not related to the model. It does not depend on the optimal selection strategy of the model, but only depends on the greedy strategy to select actions. Therefore, effective convergence can be guaranteed.

B. RESOURCE ALLOCATION BASED ON DOUBLE DQN

In order to solve the problem of limited capacity and high latency of a single MEC server, consider an IoV scenario with multiple MEC servers in multiple cells. Compared with the traditional static user scene, this dynamic scene makes the problem more complicated. And the task offloading problem is a mixed integer nonlinear programming. Traditional optimization methods or heuristic algorithms can only obtain suboptimal solutions to the problem [35]. Considering the user's mobility, dynamic scenarios, and more complex models, deep reinforcement learning is proposed to solve the problem. The centralized control distribution method is adopted, and the controller of the multi-MEC server located in the core network is used as an intelligent body. The controller coordinates MEC servers of all cells. Because reinforcement learning is model-free. First, the problem needs to be modeled based on three elements except the state transition probability.

(1) Status: The status of each time slot is set to the computing power that each MEC server has at the beginning of this time slot. That is, the remaining computing power of MEC server. Because the size of the tasks performed by MEC server is different, the computing power allocated to each task is also different. This leads to differences in the remaining computing power of MEC server after a time slot ends. In addition, the computing power of each time slot of MEC server is only related to the remaining computing power of the previous time slot and the computing power released after the previous time slot is calculated. This state change satisfies the Markov property. Therefore, the state space S is defined as follows:

$$S(t) = \{s_1(t), \cdots s_i(t) \cdots s_M(t)\} \quad (30)$$

In the formula, $S(t)$ is the state space of the t time slot. $s_i(t)$ represents the computing power of the i MEC server at the beginning of the t time slot.

(2) Action: The core of Double DQN is the Q-learning algorithm. In order to avoid the continuous action causing the action space to be too large, the action is discretized. According to the modeling problem, the variables to be optimized mainly include the offloading decision of the user task of the connected car, the transmission power and the computing power allocated to the user by MEC. In the multi-cell scenario, there are three calculation modes for the tasks of IoV user: local calculation, offloading to MEC server of the cell for calculation, and offloading to the MEC server of other nearby cells for calculation. In order to show the offloading scheme of the agent more intuitively, define the action vector as A:

$$A(t) = \{\Omega, c_1, \cdots, c_i, \cdots, c_N, p_1, \cdots, p_i, \cdots p_N\} \quad (31)$$

In the formula, Ω is the offloading decision vector of the user task, which describes the offloading decision of the user task. c_i represents the computing power allocated by MEC server for the i user. p_i represents the transmission power of the i IoV user.

(3) Reward: The agent expresses the degree of satisfaction with the action through the expected value of the reward over a period of time. Combined with the objective function C_s, the goal of the original problem is to minimize the cost function. The goal of reinforcement learning is to maximize immediate rewards. Considering that the immediate reward and the cost function of this article are negatively correlated, the immediate reward function is defined as follows:

$$R(s,a) = \frac{C_l - C_s(s,a)}{C_l} \quad (32)$$

In the formula, $R(s,a)$ represents the immediate reward for choosing action a in state s. C_l represents the cost of all tasks calculated locally, which can be understood as the upper

limit of the cost function. $C_s(s, a)$ represents the cost of s performing action a when the current time slot is in the state.

Then the long-term cumulative discount reward value $Q(s, a)$ of the original problem is expressed as follows:

$$Q_\pi(s, a) = E_\pi\left[\sum_{t=1}^{T}\beta^{t-1}R(t)\right] = E_\pi\left[\sum_{t=1}^{T}\beta^{t-1}\frac{C_l - C_s(t)}{C_l}\right] \quad (33)$$

Rewrite the above formula through Bellman optimization function:

$$Q_\pi(s, a) = (1-\alpha)\,Q_\pi(s, a) + \alpha\left(R(s, a) + \beta\max_{a'}Q_\pi(s', a')\right) \quad (34)$$

For MEC controller, the purpose of learning is to find a strategy to maximize long-term accumulated rewards:

$$\pi^* = \arg\max_{a\in A} Q_\pi(s, a) \quad (35)$$

In order to fully explore the action space, the ε greedy algorithm in exploration and utilization is adopted. The ε greedy method is based on probability to compromise exploration and utilization. That is, every time you try, explore with the probability of ε and use it with the probability of $1-\varepsilon$. MEC controller will randomly generate a value ϕ between (0, 1) during the training process. When ϕ is greater than ε, MEC controller will randomly select a strategy from the action space to explore the action space a. Otherwise, MEC controller selects the strategy with the largest $R(s, a)$ in the current state through the Double DQN network.

Although the action has been discretized, the action strategy assigned by the centralized controller covers all action possibilities. This may include multiple non-existent situations. The proposed strategy filters the action space after the completion of the construction of the action space, and eliminates impossible situations. This further reduces the action space, speeds up the training, and reduces the training delay [36]. In addition, due to the mobility of connected car users, the cell where the user is located may change. And with the increase of users, the action space will also increase exponentially. Therefore, a certain amount of preprocessing was carried out for this situation. For vehicle network user v, if the time delay $t_v^l \le T_v^{\max}$ is calculated locally, the task will be calculated locally. Otherwise, it will choose to offload to MEC server for calculation. Through a series of data preprocessing, the increase of action space can be well controlled.

The task offloading and resource allocation process based on Double DQN is shown in Fig.3.

V. EXPERIMENT SCHEME AND RESULT ANALYSIS

In the experiment, we consider the scenario of multiple cells with multiple MEC servers, where each cell deploys one MEC server. And IoV users in the community are evenly distributed. In addition, the centralized controller deployed in the core network can schedule all base stations and MEC servers. The proposed task offloading strategy is deployed in a centralized controller. The actual scenario uses offline training and online resource scheduling. The specific simulation parameters are shown in Table 1. These parameters are obtained from the experience of many experiments, and a large number of similar cases have been referred to.

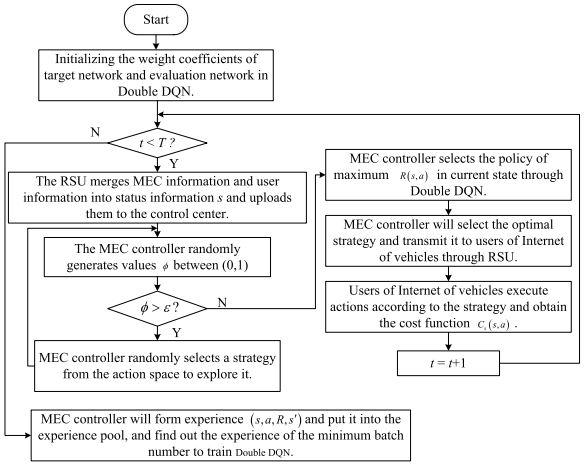

FIGURE 3. Task offloading and resource allocation process based on Double DQN.

TABLE 1. Simulation parameters.

Parameters	Value
Local computing ability	1GHz
Task packet size	[500,900] Kbits
Number of CPU revolutions required by the task	[800,1200] Megacycles
Maximum capacity of MEC server	5GHz
Learning rate (Lr)	0.1
Reward discount factor	0.095
Greedy factor	0.99
Experience pool size	550
Optimizer / activation function	Adam/ReLu

A. CONVERGENCE ANALYSIS

For comparison and explanation, three other offloading schemes are introduced, namely, user tasks are only calculated locally, user tasks are only offloaded in the cell (local and cell MEC servers), and user tasks are randomly offloaded (local and cell MEC). Server and nearby community MEC server). In order to simplify the description, use all local, local offloading, and random offloading respectively.

Since different learning rates will affect the convergence of the algorithm, the convergence curve of the reward value under different learning rates in the Double DQN algorithm is shown in Fig.4.

As can be seen from the above figure, when the learning rate is 0.1, the convergence speed is the fastest. When the number of iterations is 450, the reward value gradually converges to 0.42. When the learning rate is 0.01, when the number of iterations is 600, it gradually converges to 0.4.

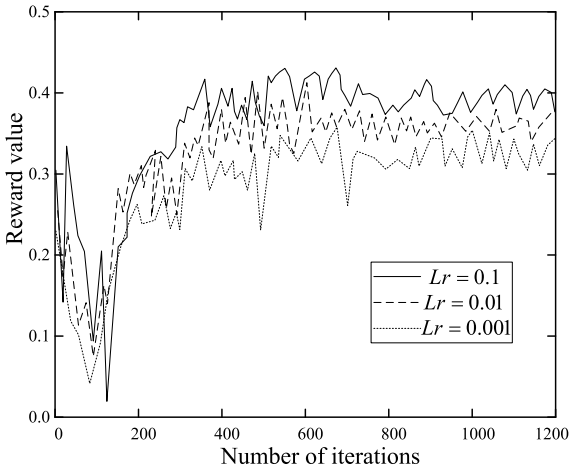

FIGURE 4. Convergence of reward value under different learning rates.

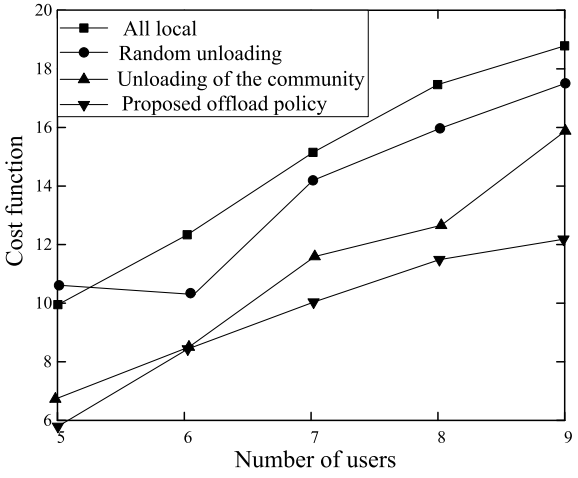

FIGURE 5. The influence of the number of IoV users on cost function.

When the learning rate is 0.001, when the number of iterations is 800, it gradually converges to 0.39. The smaller the learning rate, the easier it is to fall into a local optimal solution. Conversely, if the learning rate is too large, the reward value may not converge. It is observed through simulation that the learning rate of 0.1 is the best.

B. IMPACT ANALYSIS FOR DIFFERENT FACTORS ON COST FUNCTION

1) IMPACT OF THE NUMBER OF USERS ON COST FUNCTION

The relationship between the number of different users and the cost function is shown in Fig.5

As can be seen from the above figure, as the number of user increases, the cost functions of the four schemes also increase. The proposed resource allocation scheme based on Double DQN is superior to the scheme in which tasks are only offloaded in the cell. When the number of users is small, the gap between the two is small. But with the gradual increase in the number of users, the cost gap between the two allocation schemes has gradually increased. When the number of users is 9, it can be seen that the Double DQN solution reduces the cost by about 29% compared to the offloading solution in this cell alone. Mainly because the number of users gradually increased and the task was only offloaded in the cell, the load on MEC server gradually increased. The proposed strategy can offload tasks to nearby MEC servers to achieve load balancing of the entire network environment.

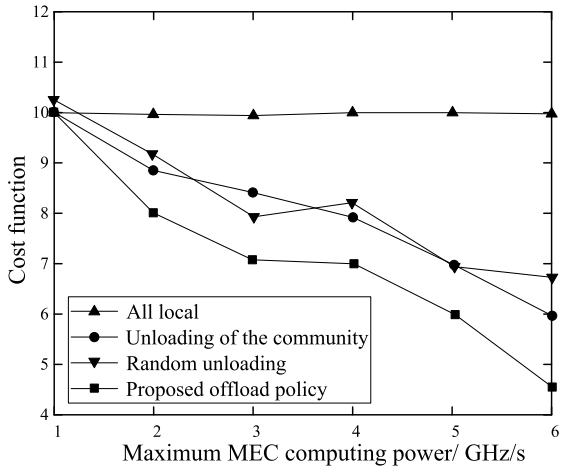

FIGURE 6. Influence of MEC capacity on cost function.

2) IMPACT OF MEC CAPACITY ON COST FUNCTION

The relationship between the maximum computing capacity of MEC server and the cost function is shown in Fig.6.

Since the solution where tasks are all calculated locally does not involve MEC server, the cost remains unchanged. The random offloading scheme has random resource allocation, so the cost will fluctuate greatly. The cost of the proposed task offloading solution based on Double DQN and the task offloading solution only in the cell gradually decreases with the increase of MEC server capacity. When the computing power of MEC server is 4GHz/s, the task offloading solution based on Double DQN reduces the cost by about 15% compared to the offloading solution in this cell only. However, as the capacity of MEC server increases, the gap between the proposed scheme and the task of only offloading in the cell gradually decreases. This is mainly because the resources of MEC server are sufficient to meet the task requirements of the current cell.

3) IMPACT OF VEHICLE SPEED ON COST FUNCTION

Speed itself is not a variable to optimize the original problem. However, the change in the speed of vehicles will cause changes in the channel conditions, which will affect the delay and energy consumption in the transmission phase. Therefore, the cost of the system will gradually increase as the moving speed of vehicles increases. Then the relationship between vehicles moving speed and the cost function is shown in Fig.7.

As can be seen from the above figure, the local calculation scheme of the task does not involve data transmission, so the cost remains unchanged. The random offloading scheme is completely random, so the cost will fluctuate greatly.

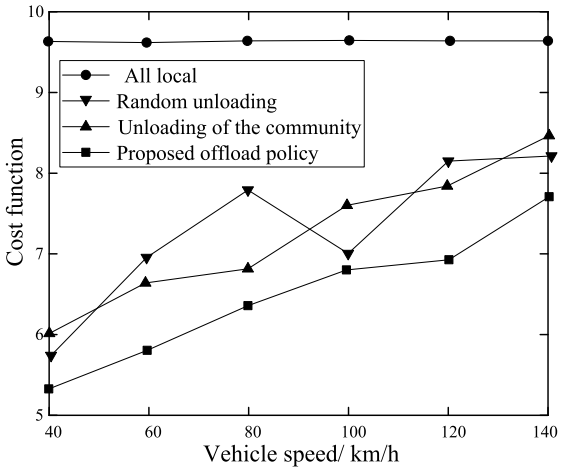

FIGURE 7. Influence of vehicle speed on cost function.

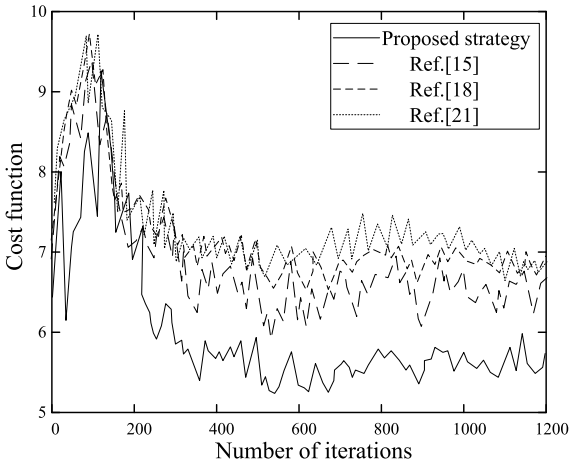

FIGURE 8. Convergence of different offloading strategies.

The proposed task offloading scheme based on Double DQN has the lowest cost. And it is about 12% lower than the cost of offloading only in this cell.

C. COMPARATIVE ANALYSIS WITH OTHER OFFLOADING STRATEGIES

In order to demonstrate the performance of the proposed task offloading strategy, it is compared with the strategies in reference [15], reference [13], and reference [20]. The convergence curve of the cost function of various offloading strategies with the number of iterations is shown in Fig.8.

As can be seen from the above figure, after 500 iterations, the cost function gradually converges to about 5.6. It can be seen that the image convergence has certain fluctuations. The main reason is that the amount of task data for each user is different, and the remaining computing power of MEC server in each time slot is different. Therefore, there will be certain fluctuations in the calculation of the cost function. In addition, the offloading strategies in reference [15], reference [13] and reference [20] are compared. It can be clearly observed that the cost function of Double DQN is better than the other three. Mainly because the Double DQN algorithm

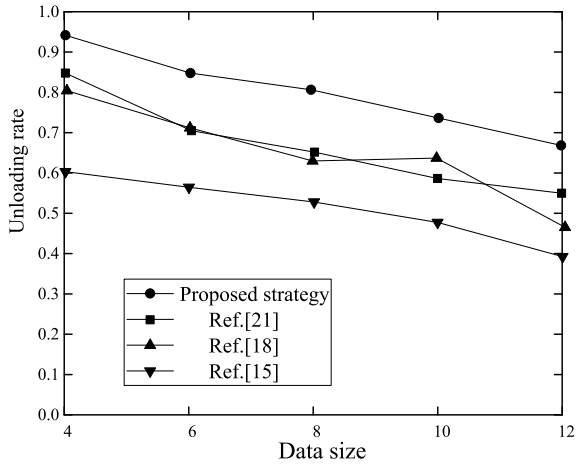

FIGURE 9. Comparison of offloading rates of different strategies.

solves the problem of overestimation by improving the loss function.

In addition, the comparison result of the offloading rate of different offloading strategies is shown in Fig.9.

As can be seen from the above figure, the offloading rate decreases with the increase of data. This means that a large amount of data may make offloading difficult. But the proposed strategy uses Double DQN to perform optimal task offloading. Compared with other strategies, its offloading rate is the highest. Reference [15] offloaded the service in the vehicle network to the car based on the collaboration method of MEC and cloud computing. However, the local calculation of vehicles is not considered, so the offloading rate is low. Reference [13] developed a joint edge computing and caching scheme, and used deep reinforcement learning to realize computing resource management. However, the resources of vehicles itself and edge nodes are not fully utilized. Therefore, the offloading rate is not ideal. Reference [20] uses linear programming optimization and binary particle swarm optimization to trigger dynamic task allocation, and considers the mobility of vehicles. But the use of fog node resources is not ideal. Therefore, the offloading rate is lower than the proposed strategy.

VI. CONCLUSION

With the continuous development of IoV technology, new IoV service applications continue to emerge. The application range of IoV has also been extensively expanded. Research hotspots such as smart cities and smart transportation are inseparable from driving networking technology, and IoV data has also shown explosive growth. This puts a lot of pressure on the existing IoV and core networks. In order to relieve the pressure on core network and meet the strict latency requirements of IoV applications, a task offloading strategy based on reinforcement learning computing in IoV edge computing architecture is proposed. Based on the designed system architecture of IoV, the calculation model, communication model, interference model and corresponding privacy protection model of task offloading strategy are

constructed. Moreover, user cost function is minimized the objective function. Double DQN algorithm is used to solve the problem to realize the reasonable allocation of computing resources and complete effective offloading of tasks in IoV. Simulation results show that the proposed offloading strategy can achieve rapid convergence. In addition, when the number of users increases, vehicle speeds increase and MEC computing power increases, the cost is the lowest compared to other offloading solutions. At the same time, the proposed strategy has the highest offloading rate. It can be seen that it has better performance and can be well applied to IoV.

The scenario considered is a one-way straight road with no intersection, but the actual road scene is very complicated. The complex road scenes in reality still need further research. Besides, IoV users move all the time, and the network topology changes rapidly. Frequent changes in communication user links may cause communication interruption. In the next work, we can consider combining relay communication, and the reliability of communication can be improved by selecting a suitable relay.

REFERENCES

[1] Z. Sharmin, A. W. Malik, A. Ur Rahman, and R. M. D. Noor, "Toward sustainable micro-level fog-federated load sharing in Internet of vehicles," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3614–3622, Apr. 2020.

[2] J. Feng, Z. Liu, C. Wu, and Y. Ji, "Mobile edge computing for the Internet of vehicles: Offloading framework and job scheduling," *IEEE Veh. Technol. Mag.*, vol. 14, no. 1, pp. 28–36, Mar. 2019.

[3] X. Xu, Y. Xue, L. Qi, Y. Yuan, X. Zhang, T. Umer, and S. Wan, "An edge computing-enabled computation offloading method with privacy preservation for Internet of connected vehicles," *Future Gener. Comput. Syst.*, vol. 96, pp. 89–100, Jul. 2019.

[4] S. Wan, X. Li, Y. Xue, W. Lin, and X. Xu, "Efficient computation offloading for Internet of vehicles in edge computing-assisted 5G networks," *J. Supercomput.*, vol. 76, no. 4, pp. 2518–2547, Apr. 2020.

[5] A. Koubáa, B. Qureshi, M.-F. Sriti, A. Allouch, Y. Javed, M. Alajlan, O. Cheikhrouhou, M. Khalgui, and E. Tovar, "Dronemap planner: A service-oriented cloud-based management system for the Internet-of-Drones," *Ad Hoc Netw.*, vol. 86, pp. 46–62, Apr. 2019.

[6] Z. Ning, J. Huang, X. Wang, J. J. P. C. Rodrigues, and L. Guo, "Mobile edge computing-enabled Internet of vehicles: Toward energy-efficient scheduling," *IEEE Netw.*, vol. 33, no. 5, pp. 198–205, Sep. 2019.

[7] J. Yao and N. Ansari, "Online task allocation and flying control in fog-aided Internet of drones," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5562–5569, May 2020.

[8] J. Liu, W. Wang, D. Li, S. Wan, and H. Liu, "Role of gifts in decision making: An endowment effect incentive mechanism for offloading in the IoV," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 6933–6951, Aug. 2019.

[9] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, "Joint load balancing and offloading in vehicular edge computing and networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4377–4387, Jun. 2019.

[10] A. Ashok, P. Steenkiste, and F. Bai, "Vehicular cloud computing through dynamic computation offloading," *Comput. Commun.*, vol. 120, pp. 125–137, May 2018.

[11] A. Tolba, "Content accessibility preference approach for improving service optimality in Internet of vehicles," *Comput. Netw.*, vol. 152, pp. 78–86, Apr. 2019.

[12] L. Zhang, M. Luo, J. Li, M. H. Au, K.-K.-R. Choo, T. Chen, and S. Tian, "Blockchain based secure data sharing system for Internet of vehicles: A position paper," *Veh. Commun.*, vol. 16, pp. 85–93, Apr. 2019.

[13] Y. Dai, D. Xu, S. Maharjan, G. Qiao, and Y. Zhang, "Artificial intelligence empowered edge computing and caching for Internet of vehicles," *IEEE Wireless Commun.*, vol. 26, no. 3, pp. 12–18, Jun. 2019.

[14] R. Hao, H. Yang, and Z. Zhou, "Driving behavior evaluation model base on big data from Internet of vehicles," *Int. J. Ambient Comput. Intell.*, vol. 10, no. 4, pp. 78–95, Oct. 2019.

[15] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7944–7956, Aug. 2019.

[16] M. Vondra, Z. Becvar, and P. Mach, "Vehicular network-aware route selection considering communication requirements of users for ITS," *IEEE Syst. J.*, vol. 12, no. 2, pp. 1239–1250, Jun. 2018.

[17] Q. Zhang, L. Gui, F. Hou, J. Chen, S. Zhu, and F. Tian, "Dynamic task offloading and resource allocation for mobile-edge computing in dense cloud RAN," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3282–3299, Apr. 2020.

[18] Y. Chen, N. Zhang, Y. Zhang, and X. Chen, "Dynamic computation offloading in edge computing for Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4242–4251, Jun. 2019.

[19] M. Chen, Y. Tian, G. Fortino, J. Zhang, and I. Humar, "Cognitive Internet of vehicles," *Comput. Commun.*, vol. 120, pp. 58–70, May 2018.

[20] C. Zhu, J. Tao, G. Pastor, Y. Xiao, Y. Ji, Q. Zhou, Y. Li, and A. Yla-Jaaski, "Folo: Latency and quality optimized task allocation in vehicular fog computing," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4150–4161, Jun. 2019.

[21] J. Kang, R. Yu, X. Huang, and Y. Zhang, "Privacy-preserved pseudonym scheme for fog computing supported Internet of vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2627–2637, Aug. 2018.

[22] C. Wu, X. Chen, T. Yoshinaga, Y. Ji, and Y. Zhang, "Integrating licensed and unlicensed spectrum in the Internet of vehicles with mobile edge computing," *IEEE Netw.*, vol. 33, no. 4, pp. 48–53, Jul. 2019.

[23] J. I. Naser, H. A. G. Alsalman, and A. J. Kadhim, "Authentication and secure communications for Internet of vehicles (IOV)-assisted fog computing," *Telecommun. Radio Eng.*, vol. 78, no. 18, pp. 1659–1670, 2019.

[24] H. Zhou, W. Xu, J. Chen, and W. Wang, "Evolutionary V2X technologies toward the Internet of vehicles: Challenges and opportunities," *Proc. IEEE*, vol. 108, no. 2, pp. 308–323, Feb. 2020.

[25] B. Ji, X. Zhang, S. Mumtaz, C. Han, C. Li, H. Wen, and D. Wang, "Survey on the Internet of vehicles: Network architectures and applications," *IEEE Commun. Standards Mag.*, vol. 4, no. 1, pp. 34–41, Mar. 2020.

[26] C.-R. Dow, D.-B. Nguyen, S. Cheng, P.-Y. Lai, and S.-F. Hwang, "VIPER: An adaptive guidance and notification service system in Internet of vehicles," *World Wide Web*, vol. 22, no. 4, pp. 1669–1697, Jul. 2019.

[27] W. Chen, D. Wang, and K. Li, "Multi-user multi-task computation offloading in green mobile edge cloud computing," *IEEE Trans. Services Comput.*, vol. 12, no. 5, pp. 726–738, Sep. 2019.

[28] X. He, R. Jin, and H. Dai, "Peace: Privacy-preserving and cost-efficient task offloading for mobile-edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 1814–1824, Mar. 2020.

[29] C. Shu, Z. Zhao, Y. Han, G. Min, and H. Duan, "Multi-user offloading for edge computing networks: A dependency-aware and latency-optimal approach," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 1678–1689, Mar. 2020.

[30] U. Saleem, "Latency minimization for D2D-enabled partial computation offloading in mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 99, pp. 4472–4486, Apr. 2020.

[31] Q.-V. Pham, L. B. Le, S.-H. Chung, and W.-J. Hwang, "Mobile edge computing with wireless backhaul: Joint task offloading and resource allocation," *IEEE Access*, vol. 7, pp. 16444–16459, 2019.

[32] T. Zhang, Y. Xu, J. Loo, D. Yang, and L. Xiao, "Joint computation and communication design for UAV-assisted mobile edge computing in IoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 8, pp. 5505–5516, Aug. 2020.

[33] J. Wang, J. Hu, G. Min, W. Zhan, Q. Ni, and N. Georgalas, "Computation offloading in multi-access edge computing using a deep sequential model based on reinforcement learning," *IEEE Commun. Mag.*, vol. 57, no. 5, pp. 64–69, May 2019.

[34] C. Zhong, M. C. Gursoy, and S. Velipasalar, "Deep reinforcement learning-based edge caching in wireless networks," *IEEE Trans. Cognit. Commun. Netw.*, vol. 6, no. 1, pp. 48–61, Mar. 2020.

[35] L. Huang, X. Feng, C. Zhang, L. Qian, and Y. Wu, "Deep reinforcement learning-based joint task offloading and bandwidth allocation for multi-user mobile edge computing," *Digit. Commun. Netw.*, vol. 5, no. 1, pp. 10–17, Feb. 2019.

[36] J. Yao and N. Ansari, "Task allocation in fog-aided mobile IoT by Lyapunov online reinforcement learning," *IEEE Trans. Green Commun. Netw.*, vol. 4, no. 2, pp. 556–565, Jun. 2020.