QoE-driven Task Offloading with Deep Reinforcement Learning in Edge intelligent IoV

Xiaoming He*, Haodong Lu†, Yingchi Mao*, and Kun Wang‡
*College of Computer and Information, Hohai University, Nanjing, China.
†College of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing, China.
‡Department of Electrical and Computer Engineering, University of California, Los Angeles, CA, USA.
Email: *isxmhe@gmail.com, †ihaodonglu@gmail.com, *yingchimao@hhu.edu.cn, ‡wangk@ucla.edu

GLOBECOM 2020 - 2020 IEEE Global Communications Conference | 978-1-7281-8298-8/20/$31.00 ©2020 IEEE | DOI: 10.1109/GLOBECOM42002.2020.9348050

***Abstract*—In the transportation industry, task offloading services of edge intelligent Internet of Vehicles (IoV) are expected to provide vehicles with the better Quality of Experience (QoE). However, the various status of diverse edge servers and vehicles, as well as varying vehicular offloading modes, make a challenge of task offloading service. Therefore, to enhance the satisfaction of QoE, we first introduce a novel QoE model. Specifically, the emerging QoE model restricted by the energy consumption, (1) intelligent vehicles equipped with caching spaces and computing units may work as carriers; (2) various computational and caching capacities of edge servers can empower the offloading; (3) unpredictable routings of the vehicles and edge servers can lead to diverse information transmission. We then propose an improved deep reinforcement learning (DRL) algorithm named RA-DDPG with the prioritized experience replay (PER) and the stochastic weight averaging (SWA) mechanisms based on deep deterministic policy gradients (DDPG) to seek an optimal offloading mode, saving energy consumption. Extensive experiments certify the better performance, *i.e.*, stability and convergence, of our RA-DDPG algorithm compared to existing work. Moreover, the experiments indicate that the QoE value can be improved by the proposed algorithm.**

***Index Terms*—Internet of Vehicles (IoV), Edge, Task Offloading, Deep Deterministic Policy Gradients (DDPG), QoE**

I. Introduction

THE Internet of Things (IoT) are required to adaptively offer various services for diverse applications. In the IoT service scenario, edge [1]–[3] module can provide low service latency, low energy consumption, appropriate caching, rapid communication, and flexible computation. In particular, for the edge intelligent Internet of Vehicles (IoV) [4], [5], the vehicles can either offload all tasks to edge servers or to other vehicles, accelerating the processing. However, due to the condition of network channels, the size of edge servers or vehicles, and the number of feedback by vehicles, the existing offloading services cannot meet the Quality-of-Experience (QoE) [6]–[8].

QoE can be regarded as the most direct experience of vehicles in service interactions. Currently, the majority of offloading services to enhance the QoE level become comprehensively attractive attention. Wang *et al.* [9] proposed a scalable mobile edge computing architecture via software defined network technology integrating the heterogeneous vehicular networks to offload the traffic from the edges, decreasing the whole latency and enhancing the QoE of vehicles. Dai *et al.* [10] introduced a novel issue that jointly optimizes the QoE value of task caching and offloading in vehicular edge networks with the constraint of computational resources and caching resources in terms of less energy consumption. As a result, vehicles without caching spaces distribute tasks to the remaining vehicles or edge servers for QoE processing.

Due to the limited caching and computational capacities of the vehicles, as well as the uncertain transmission path, the task offloading process can increase the task repetition, resulting in inconsistent vehicle offloading modes. Efficient offloading strategies in edge intelligent IoV with various modes is thus a challenge. Lu *et al.* [11] proposed a D^3PG algorithm based on the DDPG to address the trade-off between the choice of target servers and algorithm convergence. Wu *et al.* [12] redesigned the critic network of DDPG and proposed the multi-critic DDPG method to make a balance of the DDPG stability and offloading performance. To sum up, DDPG-based task offloading may bring the slow convergence and arouse the reward instability in the training process, due to traffic density, load speed, driving time and vehicular characteristics.

Motivated by the above issues, we concentrate on QoE-based task offloading in edge intelligent IoV. First, we redesign the edge networks combining edge computational capacities, caching spaces, and communication bandwidths with task offloading. The novel framework can expand the service category and enhance the service complexity since the roadside units (RSU) equipped with an edge server, and the vehicles with caching spaces. We fully consider a novel QoE model with energy consumption restricted by service delay according to diverse offloading modes. Then, an efficient task offloading mechanism for vehicles is proposed while considering the QoE value and algorithm performance. Specifically, we introduce the prioritized experience replay (PER) [13] and the stochastic weight averaging (SWA) [14] mechanisms to reshape the DDPG and propose a novel algorithm named RA-DDPG, enhancing the QoE value, speeding up the training efficiency while stabilizing the solutions.

The main contributions of this paper can be summarized.

- We add task offloading service into edge intelligent IoV to bring a novel experience. We redesign the architecture by combining three influencing factors: caching spaces, computational capacities, and channel conditions. Based

978-1-7281-8298-8/20/$31.00 ©2020 IEEE

on this, we further propose a novel QoE model with the energy consumption restriction to measure QoE.
- We develop an improved RA-DDPG algorithm combined with SWA and PER mechanisms to improve QoE. The proposed algorithm seeks an optimal offloading mode to minimize energy consumption while improving algorithm performance, *i.e.*, stability and convergence.
- We conduct extensive experiments in the vehicular test bed to evaluate the algorithm performance and rewards. The numeric results show that RA-DDPG outperforms the existing DRL methods, and achieves a better QoE value.

The rest of this paper is as follows. Section II summarizes the related work. Section III proposes the network model and task offloading model. Section IV introduces the task offloading mechanism. Section V performs some experimental analysis and evaluations. Section VI comes to the conclusion.

II. System Model and Problem Formulation

We are the first to describe the network model in this part of task offloading services for edge intelligent IoV. Then the task offloading model is summarized for the QoE model with *energy consumption* restricted by *service latency*.

A. Network Model

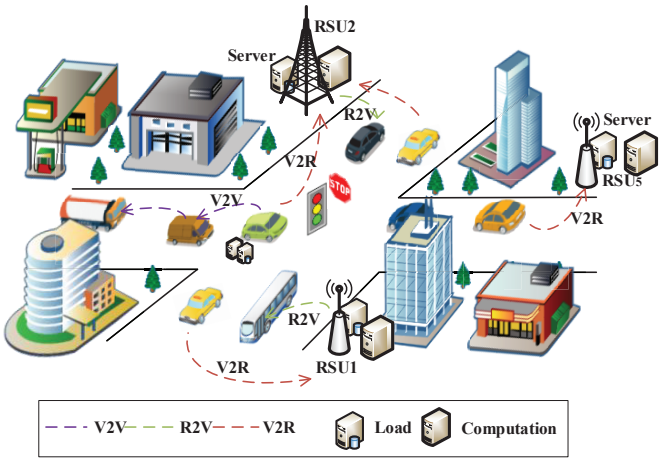

Fig. 1: Task offloading in Edge intelligent IoV.

In Fig. 1, RSUs with the capacity of high-speed communication are located in an area. The caching capacity of the RSUs is defined as ca^{ϵ}. Moreover, the RSUs with edge servers can process the offloaded tasks that are intensive computation. The computational capacity in edge servers is defined as co^{ϵ}. Vehicles driven on the road can produce task-data chunks (*e.g*, mobile sensing messages, driving patterns and control indications of vehicles, or even road congestion conditions) on edge networks. To combine views of diverse vehicles, extract the main features of traffic, and identify legal indications of control, the generated task-data chunks need to be processed. Meanwhile, the produced task-data chunk can be transmitted among vehicles and the RSU.

Task offloading is used to present the merging of task-data chunks processing and transmission process. The task-data chunks have diverse sizes and computational demands, *i.e.*, vehicular sensing devices can gather traffic status and transmit to other vehicles after extracting features. The amount of information is small but the processing is complex L types of task-data chunks are considered, and type l ($l \in L$) is defined as $\{s_l,c_l,t_l^{max},E_l^{max}\}$. Note that s_l is the size of type l task-data chunk, c_l represents the amount of computation, t_l^{max} is the maximal service latency, and E_l^{max} is the maximal energy consumption, respectively. Each task-data chunk can either be transmitted to the edge server to perform, or be processed on a vehicle with computational resources. If vehicles and the RSU have free caching spaces, they cache the processed task-data chunks. A vehicle only caches one processed task-data chunk at time slot t since the limited caching capacity. In addition, vehicles and the RSU can share the cached task-data chunks.

B. Task Offloading Model

In the task offloading process, two task offloading modes are proposed, *i.e.*, offloading via V2V and R2V, respectively.

- **Task offloading in V2V mode**

For type l task-data chunk, the time consumed by a local vehicle computation is

$$t_l = \frac{s_l}{cv}, \tag{1}$$

where cv is the computational capacity of a vehicle.

After computing, the vehicle caches the task-data chunk by and transmits it to other vehicles. Since the coverage of signals is limited, the vehicles can communicate with others restricted by distances. We can obtain the transmission rate $\chi()$ associated with distance from

$$\chi(v) = \int_0^d r(v_l)f(v_l,P_l)dv_l, \tag{2}$$

where d is maximal distance; v_l refers to the distance among the vehicles; P_l is the parameter of Zipf distribution [13].

According to the transmission mode, the time consumed by task offloading includes three categories, (1) the computation time to process the type l task-data chunk, and (2) the time for caching the proposed task, (3) the transmission time between two vehicles. Meanwhile, the popularity of task-data chunks follows a Zipf distribution. The popularity of task-data chunks can be calculated by

$$Q_l = \frac{l^{-\beta}}{\Sigma_{l=1}^{L} l^{-r}}, \tag{3}$$

where β is the parameter of popularity distribution; r is the parameter of skewness.

As a result, the time consumption completing the offloading of type l task-data chunk among vehicles via V2V mode at the t can be calculated by

$$t_{v2v,l} = \frac{1}{L}\Sigma_{l=1}^{L} Q_l\{t_l + \frac{1}{\chi_{v2v}(1-\Sigma_{l'=1}^{l}\frac{n_{1,l'}^t}{N})} + \frac{s_l}{\chi_{v2v}}\}, \tag{4}$$

where $\frac{n^t_{1,l'}}{N}$ means the number of vehicles that have cached type l' task-data chunk at the t; N refers to the total number of vehicles; χ_{v2v} denotes the V2V transmission rate.

The corresponding energy consumption for type l task-data chunk can be obtained by

$$E_l = \frac{1}{L}\Sigma_{l=1}^{L} Q_l\{c_y t_l + P_{v2v}\frac{s_l}{\chi_{v2v}} + \frac{S_v}{\chi_{v2v}(1-\Sigma_{l'=1}^{l}\frac{n^t_{1,l'}}{N})}\}, \tag{5}$$

where P_{v2v} refers to transmission power via V2V mode; c_y denotes to CPU cycles; S_v is the caching capacity of a vehicle.

- **Task offloading in R2V mode**

A vehicle transmits the task-data chunk to the RSU after the task-data chunk is processed. Then the processed task-data chunk is transmitted to the requested vehicles via the RSU. The time cost at the t for a vehicle, transmitting type l task-data chunk to RSU m, is

$$t_{v2r,l,m} = \frac{1}{L}\Sigma_{l=1}^{L} Q_l\{\frac{1}{\chi_{v2r}} + \frac{s_l}{\chi_{v2r}}\}, \tag{6}$$

and the time cost for the RSU m, caching and forwarding it to other vehicles, is

$$t_{r2v,l,m} = \frac{1}{L}\Sigma_{l=1}^{L} Q_l\{\frac{1}{\chi_{r2v}(1-\Sigma_{l'=1}^{l}\frac{n^t_{1,l'}}{N})} + \frac{s_l}{\chi_{r2v}}\}. \tag{7}$$

Consequently, the time consumption of task offloading in R2V mode can be calculated by

$$t_v = t_l + t_{v2r,l,m} + t_{r2v,l,m}. \tag{8}$$

The corresponding average energy consumption for type l task-data chunk can be acquired by

$$E_{2,l,m} = \frac{1}{L}\Sigma_{l=1}^{L} Q_l\{c_y t_l + P_v t_{v2r,l,m} + \frac{ca^\epsilon}{\chi_{r2v}(1-\Sigma_{l'=1}^{l}\frac{n^t_{1,l'}}{N})}\}, \tag{9}$$

where P_v refers to the transmission power via V2R/R2V mode.

In addition, various types of task-data chunks may be chosen in the RSU for computation, a task-data chunk can wait to be processed in the queue of the RSU. Note that the task-data chunk has a smaller amount of data. When multiple types of task-data chunks are randomly produced by vehicles at the t, the task-data chunks with less data can be transmitted to the RSU at a faster speed. Then the processed task-data chunk is transmitted to the required vehicle. Therefore, the queuing, processing, and transmission time for type l task-data chunk in RSU m via R2V mode can be obtained as

$$t_{r2v,l,m} = \frac{1}{L}\Sigma_{l=1}^{L} Q_l\{\frac{c_l(1-\rho_{m,l})}{co^\epsilon} + \frac{s_l}{ca^\epsilon} + \frac{s_l}{\chi_{r2v}}\}, \tag{10}$$

where $\rho_{m,l}$ is the probability that type l task-data chunk is offloaded to RSU m.

The corresponding average energy consumption for type l task-data chunk can be obtained by

$$E_{r2v,l,m} = \frac{1}{L}\Sigma_{l=1}^{L} Q_l\{k_{server}\{\frac{c_l(1-\rho_{m,l})}{co^\epsilon} + \frac{s_l}{ca^\epsilon}\} + P_v\frac{s_l}{\chi_{r2v}}\}, \tag{11}$$

where k_{server} is the energy parameter of RSUs.

C. QoE Model

Based on the task offloading, we give the energy consumption to measure the satisfaction of QoE. Since various targets of computation, diverse strategies of caching, and multi modes of transmission, target-data chunks present different offloading performance. For type l target-data chunk, the energy consumption in an area under service delay t_l^{max} constraint can be expressed as

$$QoE = -\{\xi_l\delta_l E_l + \xi_l\eta_{2,l,m}E_{2,l,m} + (1-\xi_l)\rho_{m,l}E_{r2v,l,m}\}, \tag{12}$$

where ξ_l is the probability that a vehicle computes type l task-data chunk on itself. δ_l and $\eta_{2,l,m}$ are the probabilities that vehicles transmit type l task-data chunk via the modes of V2V and R2V with the processed task-data chunk, respectively.

The task offloading can maximize the satisfaction of QoE for vehicles, which acquire multiple types of task-data chunks under service latency constraints. To this end, the objective function $f(t)$ concerning the maximal value of QoE is

$$\begin{aligned}
\textbf{f(t)} \quad & \max \quad QoE(E_l, \quad E_{2,l,m}, \quad E_{r2v,l,m}) && (13)\\
& \text{s.t.} \quad \text{C1}: \quad \Sigma_{l=1}^{L}\{\xi_l\eta_{2,l,m}s_l + (1-\xi_l)\rho_{m,l}s_l\} \le ca^\epsilon; && (13\text{-}1)\\
& \qquad \text{C2}: \quad 0 \le \xi_l,\delta_l,\eta_{2,l,m},\rho_{m,l} \le 1; && (13\text{-}2)\\
& \qquad \text{C3}: \quad \delta_l + \eta_{2,l,m} = 1; && (13\text{-}3)\\
& \qquad \text{C4}: \quad \rho_{m,l} = 1, && (13\text{-}4)
\end{aligned}$$

where C_1 represents that the number of cached task-data chunks cannot be higher than the caching capacity of RSU m; C_2 shows the ranges of variables; C_3 indicates that the task-data chunk can either choose V2V mode or V2R mode for transmission; C_4 gives that the task-data chunk can select RSU m to process.

III. DDPG-BASED TASK OFFLOADING ALGORITHM

Since the optimal strategy in (13) forms a complex relationship and cannot be obtained a satisfactory solution in direct, we design a DDPG-based task offloading mechanism through the DRL method. In the offloading process, aiming to speed up the training process, the PER [13] mechanism is proposed to enhance the availability of the experience replay buffer. Moreover, reducing the noise of model training and thus stabilizing the solutions, the SWA [14] scheme is introduced to average weights along with the optimal process. As a result, a RA-DDPG algorithm is proposed based on SWA and PER to carry out the task offloading.

A. RA-DDPG Definition

According to the RA-DDPG algorithm, we can design the action $\mathbb{A}(t)$, state $\mathbb{S}(t)$, and reward $\mathbb{R}(t)$.

State: At the t, the state is given as $\mathbb{S}(t) = [E_l,E_{2,l,m},E_{r2v,l}]$. E_l presents the energy consumption of type l task-data chunk via V2V mode; $E_{2,l,m}$ shows the energy consumption of the processed type l task-data chunk via R2V mode; and $E_{r2v,l,m}$ gives the energy consumption of type l task-data chunk processed in RSU via R2V mode, respectively.

Action: At the t, let us denote the action $\mathbb{A}(t)$ consisting of the following components. ξ_l indicates that a vehicle computes type l task-data chunk on itself, δ_l shows that a vehicle chooses to transmit type l task-data chunk via V2V mode, $\eta_{2,l,m}$ gives that a vehicle chooses to transmit type l task-data chunk via R2V mode after the task-data chunk is processed, and $\rho_{m,l}$ presents that type l task-data chunk is offloaded to the RSU, *i.e.*, $\mathbb{A}(t) = \{\xi_l, \quad \delta_l, \quad \eta_{2,l,m}, \quad \rho_{m,l}\}$.

Reward: At the t, the reward $\mathbb{R}(t) = f(t)$ is set as the maximal value of QoE.

B. RA-DDPG Training

The DDPG agent learns the optimal policy and value function through interactions with the network environment. θ^f is updated with sampled policy gradient via primary actor neural network, which can be defined as

$$\nabla_{\theta^f} Q(s,a,\theta^g) = E[\nabla_f Q(s^t,a^t|\theta^g)\nabla_{\theta^f}\mu(S^t|\theta^f)], \qquad (14)$$

where $\mu(S^t|\theta^f)$ refers to the explored policy; $Q(S^t,A^t,\theta^g) = E[f(t) + \gamma Q(S^{t+1},\mu(S^{t+1})|\theta^g)]$.

The action-value network is defined by θ^g, and the loss function is defined by $L(\theta^g)$. θ^g is updated by $L(\theta^g)$,

$$L(\theta^g) = (r(s_t,a_t) + \gamma Q'(s_{t+1},a_{t+1},\theta^f) - Q(s_t,a_t,\theta^g))^2. \qquad (15)$$

Recently, in the training process, deep neural networks tend to diverge. Fortunately, the introduced prioritized experience replay mechanism in this paper is used to replay experiences frequently, associated with worse performance or extremely successful attempts.

Specifically, we update the value of estimation for the value function $Q(s,a)$ of action using TD error in most RL algorithms, frequently, so the value of TD error can act as the correlation. In addition, TD error can present the measurement that an agent can learn from past experience. The greater the TD error is, the more positive the correlation is. As a result, the high experience of TD error is linked to the extremely successful attempts, and likely to gain the high value. Moreover, the experience of TD error limited by the big negative magnitude refers to the agent behavior condition. The state of the condition cannot be learnt by the agent. Replying the above experience frequently cannot only achieve the real results of bad behavior in the relevant states, but also avoid the occurrence of bad behavior. Therefore, the value of awful learned experiences can act as a high value. To evaluate the value of experiences, the absolute TD error $|\psi|$ can be selected as the index. The TD error ψ_j of experience j is shown as

Algorithm 1: Task Offloading with RA-DDPG Algorithm

```
Initialize the policy μ(S,θ^f), action-value function
  Q_{S,A,θ^g} of main network with random weights θ^f
  and θ^g.
Initialize target networks with weights θ^{f'} ← θ^f and
  θ^{g'} ← θ^g.
Initialize the prioritized experience replay buffer.
begin
    for episodes do
        Reset environment state s(0);
        for time slot t do
            Select an action 𝔸(t) based on θ^f.
            Execute 𝔸(t), observe 𝕊(t+1) and obtain
              f(t).
            Store (𝕊(t),f(t),𝕊(t+1)) into the
              prioritized experience replay buffer.
            Obtain a batch of samples from the buffer.
            Compute f(s_t,a_t).
            Update θ^f and μ(θ^g).
            Update target value using μ(θ^{f'}) and
              μ(θ^{g'}).
        end
    end
end
```

$$\psi_j = r(s_t,a_t) + \gamma Q'(s_{t+1},a_{t+1},\theta^f) - Q(s_t,a_t,\theta^g), \qquad (16)$$

where $Q'(s_{t+1},a_{t+1},\theta^f)$ is the target network of action-value.

The value is used to promote the process of experience. We define the probability of sample experience j as

$$P(j) = \frac{H_j^\zeta}{\Sigma_k H_k^\zeta}, \qquad (17)$$

where $H_j > 0$ refers to the rank of the j with the ψ_j. The ζ controls the extent of the prioritization.

Because low TD error has a replayed probability, the sampling probability acts as an approach of a stochastic factor in the process of the selected experiences, so as to ensure the variety of j. Note that the proposed variety can stop the over-fitting of neural networks. Unfortunately, replaying experiences with high TD error frequently can lead to the changes in state access frequency. Apart from that, the changes can result in oscillation or divergent in the training process of neural networks.

To handle the issue, the SWA is proposed in the process of weight changes as

$$\theta_{SWA}^f = \frac{\theta_{SWA}^f n_{SWA} + \theta^{f'}}{n_{SWA} + 1}, \qquad (18)$$

where θ_{SWA}^f initializes the weight using $\theta^{f'}$; n_{SWA} is the number of weights.

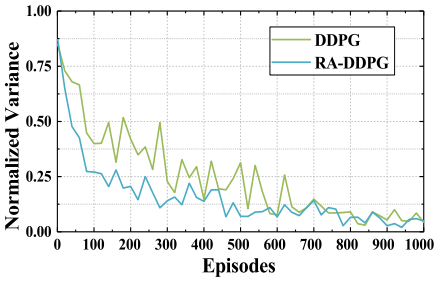

Fig. 2: Normalized Variance.

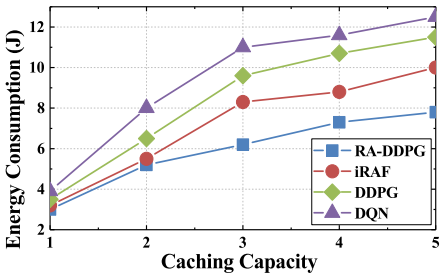

Fig. 3: Energy Consumption v.s. Caching Capacity.

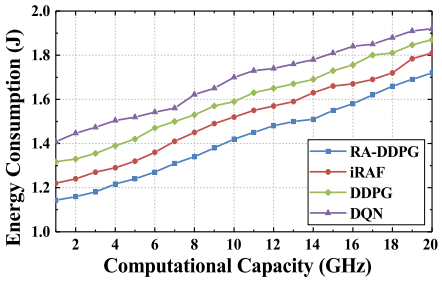

Fig. 4: Energy Consumption v.s. Computational Capacity.

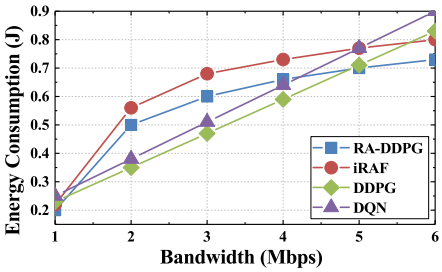

Fig. 5: Energy Consumption v.s. Channel Condition.

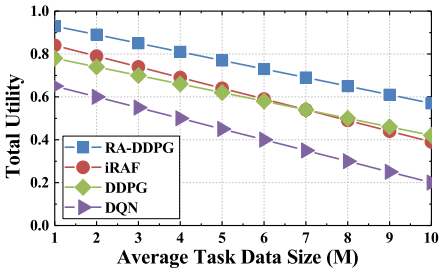

Fig. 6: Total Utility v.s. Task Data Size.

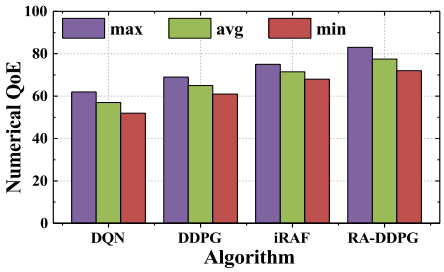

Fig. 7: QoE Performance v.s. Various Algorithm.

The proposed task offloading mechanism with RA-DDPG algorithm can be explained as follows. Specifically, the element of two networks, *i.e.*, the main network and target network include an actor and a critic, respectively. The actor neural networks of main network searches offloading policy $\mu(S^t,\theta^g)$. The critic neural networks use a gradient method to help actor neural networks learn a better policy. The target network as a primary network can generate the value to train the neural network, *i.e.*, critic. Here, the prioritized experience replay buffer can store experiences including current network states, rewards, and next states. We can update parameters using past experiences for the actor network or the critic network. SWA is applied to update the weight θ^f.

To this end, in the learning process of action, the satisfactory offloading strategy can be acquired, separately. The task-data chunks, transmitted at the t, are generated locally and excepted to be updated restrained by the service latency. The primary steps of the task offloading algorithm with RA-DDPG is illustrated in Algorithm 1.

IV. PERFORMANCE EVALUATION

In this section, we evaluate the performance of our RA-DDPG based on the real traffic data.

A. Experimental Settings

We consider a 150×120 km^2 area in a urban. At the t, there are 5 RSUs with tasks and 10 vehicles are driving at constant speed around RSUs. Specifically, the computational capacity of each vehicle is set as 1MHz-4MHz, and the RSUs are taken from 10GHz to 35GHz. The bandwidth is set as 5Mbps-20Mbps. The caching capacity is set as 1, 5, and 9, respectively. The random size of task-data chunk is from 1MB to 200MB. The service latency allowed for the task-data chunks is randomly chosen from 10ms to 70ms. The RA-DDPG, DQN [15], iRAF [16], and DDPG [17], [18] are implemented by the PyTorch. The learning rate is set as 10^{-3} for the critic network, and 10^{-4} for the actor network. The discount factor is set as 0.95, and the update rate of the target network is set as 0.01.

B. Experimental Results

The loss functions are shown in Fig. 2 to compare the convergence of DDPG and RA-DDPG algorithms. The value of loss function trained by DDPG is higher than the value of RA-DDPG from episode 1 to around 700. After 700 episodes, the values of loss function trained by the DDPG and RA-DDPG remain stable gradually. One of the reasons is that the PER can frequently replay experiences including extremely successful attempts or very poor performance. Another reason is that the SWA scheme can average weights in the optimal process and reduce the noise on training. From the training process, the efficiency of PER and SWA is verified.

In Fig. 3, we compare the proposed four offloading algorithms due to the various caching capacities, *i.e.*, 1, 5, and 9. When the loads are 5 and 9, the energy consumption based on RA-DDPG is the smallest and the upward trend is also slower than others.Note that the larger the caching space is, the better our RA-DDPG algorithm performs. Consequently, task offloading only depends on RSUs when the load is large, and depends on V2V mode when the load is only 1. In addition, the mechanism cannot only explore the caching capabilities of both vehicles and RSUs, but also can adaptively optimize offloading modes through various tasks.

In Fig. 4, the energy consumption is defined by the computational capability of vehicles. When the algorithm, *i.e.*, RA-DDPG or DDPG, is adopted, along with the decrease of computational capability of edge servers, the energy consumption of vehicles increases, the reason is that two schemes tend to upload task-data chunks to RSUs. Obviously, the novel RA-DDPG algorithm enhances the performance in term of service latency by 47.8%, 36.9%, and 30.9% compared with DDPG, iRAF, and DQN, respectively.

In Fig. 5, when the bandwidth cannot reach 5Mbps, task-data chunks prefer to be processed onboard, and be delivered to the required vehicles via V2V mode. Otherwise, task offloading mainly depends on R2V mode because R2V has a higher bandwidth than V2V mode. We can observe that when RA-DDPG is adopted, the average energy consumption decreases as the communication bandwidth increases, because the proposed scheme can dynamically adapt to diverse tasks. That is, when the computational capability of a vehicle reaches to 11GHz, a type of task-data chunk can be satisfied by joint processing and transmission with R2V mode.

In Fig. 6, the computational capacity is fixed to 15GHz, and the size of task-data chunk changes from 18MB to 180MB. We can see that the total utility decreases with the increase of task-data size due to the larger energy consumption.

Obviously, the overall efficiency of task offloading in edge intelligent IoV has been improved through the three factors. The results prove that RA-DDPG can adapt to optimize the task offloading, dynamically.

Finally, in [19], [20], the satisfactory value of QoE is higher than 60. The proposed algorithm, *i.e.*, RA-DDPG, proved to save energy consumption, brings a satisfactory QoE to vehicles. In detail, from Fig. 7, the average QoE of RA-DDPG is 64, the maximal QoE is 70, and the minimal QoE is 62. Compared to other three algorithms, the RA-DDPG algorithm generates a better QoE. From the discussed above, the QoE value is negatively proportional to energy consumption.

V. Conclusion

In this paper, we have researched the task offloading service with the purpose of QoE in the edges for IoV. To improve the satisfaction of QoE restricted by energy consumption, the influential factors of caching spaces, computational capacities, and channel conditions are considered continuously. According to the high complexity of the proposed QoE-based task offloading, we have introduced an improved DRL algorithm named RA-DDPG based on DDPG to seek an optimal offloading mode. By replacing the experience replay buffer and stochastic gradient descent using PER and SWA, the proposed RA-DDPG can improve the instability and speed up the training. Finally, the experimental results indicate that RA-DDPG has better performance than other DRL algorithms.

VI. Acknowledgment

This work is supported by National Natural Science Foundation of China under Grant No. 61872195, 61832005. The National Key Research and Development Program of China under Grant No. 2018YFC0407105.

References

[1] "Edge Cast", https://www.edge.org/

[2] X. He, H. Lu, H. Huang, Y. Mao, K. Wang, and S. Guo, "QoE-based cooperative task offloading with deep reinforcement learning in mobile edge networks", *IEEE Wireless Communications*, vol. 27, no. 3, pp. 111-117, Jun. 2020.

[3] C. Xu, K. Wang, P. Li, S. Guo, J. Luo, B. Ye, and M. Guo, "Making big data open in edges: A resource-efficient blockchain-based approach", *IEEE Transactions on Parallel and Distributed Systems*, vol.30, no.4, pp.870-882, Apr.2019.

[4] X. He, H. Lu, M. Du, Y. Mao, and K. Wang, "QoE-based task offloading with deep reinforcement learning in edge-enabled Internet of vehicles", *IEEE Transactions on Intelligent Transportation Systems*, to be published, doi: 10.1109/TITS.2020.3016002.

[5] K. Zhang, S. Leng, X. Peng, P. Li, S. Maharjan, and Y. Zhang, "Artificial intelligence inspired transmission scheduling in cognitive vehicular communications and networks", *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1987-1997, Apr. 2019.

[6] X. He, K. Wang, and W. Xu,"QoE-driven content-centric caching with deep reinforcement learning in edge-enabled IoT", *IEEE Computational Intelligence Magazine*, vol. 4, no. 4, pp. 12-20, Nov. 2019.

[7] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, "Computation offloading and resource allocation in wireless cellular networks with mobile edge computing", *IEEE Transactions on Wireless Communications*, vol. 16, no. 8, pp. 4924-4938, Aug. 2017.

[8] X. He, K. Wang, H. Huang, Y. Wang, and S. Guo, "Green resource allocation based on deep reinforcement learning in content-centric IoT", *IEEE Transactions on Emerging Topics in Computing*, vol. 8, no. 3, pp. 781-796, July-Sept. 2020

[9] Y. Wang, K. Wang, H. Huang, T. Miyazaki, and S. Guo,"Traffic and computation co-offloading with reinforcement learning in fog computing for industrial applications", *IEEE Transactions on Industrial Informatics*, vol. 15, no. 2, pp. 976-986, Feb. 2019.

[10] Y. Dai, D. Xu, K. Zhang, S. Maharjan, and Y. Zhang, "Deep reinforcement learning and permissioned blockchain for content caching in vehicular edge computing and networks", *IEEE Transactions on Vehicular Technology*, to be published, doi: 10.1109/TVT.2020.2973705.

[11] H. Lu, X. He, M. Du, X. Ruan, Y. Sun, and K. Wang, "Edge QoE: Computation offloading with deep reinforcement learning for internet of things", *IEEE Internet of Things Journal*, to be published, doi: 10.1109/JIOT.2020.2981557.

[12] J. Wu, R. Wang, R. Li, H. Zhang, and X. Hu, "Multi-critic DDPG method and double experience replay", in Proc. of the *2018 IEEE International Conference on Systems, Man, and Cybernetics*, Miyazaki, Japan, Oct. 2018.

[13] Y. Hou, L. Liu, Q. Wei, X. Xu, and C. Chen, "A novel DDPG method with prioritized experience replay", in Proc. of the *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Banff Center, Banff, Canada, Oct. 2017.

[14] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson, "Averaging weights leads to wider optima and better generalization", *arXiv: 1803.05407v3*, Feb. 2019.

[15] J. Pan, X. Wang, Y. Cheng, and Q. Yu, "Multisource transfer double DQN based on actor learning", *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 6, pp. 2227-2238, Mar. 2018.

[16] J. Chen, S. Chen, Q. Wang, B. Cao, G. Feng, and J. Hu, "iRAF: A deep reinforcement learning approach for collaborative mobile edge computing IoT networks", *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 7011-7024, Aug. 2019.

[17] X. He, K. Wang, H. Lu, W. Xu, and S. Guo, "Edge QoE: Intelligent big data caching via deep reinforcement learning", *IEEE Network*, vol. 27, no. 3, pp. 111-117, Jun. 2020.

[18] C. Xu, K. Wang, P. Li, R. Xia, S. Guo, and M. Guo, "Renewable energy-aware big data analytics in geo-distributed data centers with reinforcement learning", *IEEE Transactions on Network Science and Engineering*, vol.7, no.1, pp. 205-215, Mar.2020.

[19] X. He, K. Wang, H. Huang, and B. Liu, "QoE-driven big data architecture for smart city", *IEEE Communications Magazine*, vol. 56, no. 2, pp. 88-93, Feb. 2018.

[20] S. T. Hong and H. Kim, "QoE-aware computation offloading to capture energy-latency-pricing tradeoff in mobile clouds", *IEEE Transactions on mobile computing*, vol. 18, no. 9, pp. 2174-2189, Sept. 2019.