Artificial Intelligence Empowered Edge Computing and Caching for Internet of Vehicles

Article *in* IEEE Wireless Communications · June 2019

DOI: 10.1109/MWC.2019.1800411

Yueyue Dai
University of Electronic Science and Technology of China

Sabita Maharjan
Simula Research Laboratory

Yan Zhang
Chinese Academy of Sciences

Artificial Intelligence Empowered Edge Computing and Caching for Internet of Vehicles

Yueyue Dai, Du Xu, Sabita Maharjan, Guanhua Qiao, and Yan Zhang

***Abstract*—Recent advances in edge computing and caching have significant impacts on the developments of vehicular networks. Nevertheless, the heterogeneous requirements of on-vehicle applications and the time-variability on popularity of contents, bring great challenges for edge servers to efficiently utilize their resources. Moreover, the high mobility of smart vehicles adds substantial complexity in jointly optimizing edge computing and caching. Artificial Intelligence (AI) can greatly enhance the cognition and intelligence of vehicular networks and thus assist in optimally allocating resources for the problems with diverse, time-variant, and complex features. In this article, we propose a new architecture that can dynamically orchestrate edge computing and caching resources to improve system utility by making full use of AI based algorithms. Then we formulate a joint edge computing and caching scheme to maximize system utility and develop a novel resource scheme by exploiting deep reinforcement learning. Numerical results demonstrate the effectiveness of the proposed scheme.**

I. Introduction

Emerging applications enabled by the Internet of Things (IoT) and wireless technologies, such as automated driving, smart navigation, on-vehicle videos play a crucial role towards a safer and more intelligent transportation system. However, the resource-constrained vehicles pose significant challenges for supporting these computation-intensive and delay-sensitive applications. Furthermore, huge amount of data and popular contents generated by on-vehicle sensors and applications make it challenging for vehicles to provide the required amount of resources to process and store. Remote cloud servers can offer a large amount of computing and caching resources. However, considerable transmission cost and relatively long latency may degrade user experience, especially for delay-sensitive applications [1].

Mobile Edge Computing (MEC) is a promising paradigm that brings computing and caching resources to mobile users at the network edge, which allows applications and contents to be processed and cached quickly by edge servers, instead of remote cloud servers, to satisfy the requirements of mobile users via computation offloading and distributed content caching. Computation offloading enables execution of computation-intensive and delay-sensitive tasks at resource-rich edge servers that helps prolong battery life of mobile users [2], [3] and also considerably reduces task processing latency [4]. Distributed content caching can alleviate backhaul pressure and content access delay by caching popular contents in proximity to mobile users [5], [6]. To meet the stringent latency requirements of computation-intensive and rich-media tasks, the authors in [7] proposed to jointly optimize computation and caching by caching the task that needs to be computed or caching the task result after being processed on edge servers.

A few studies such as [8], [9], [10] investigated the performance of edge computing and caching in vehicle networks. The authors in [8] proposed two computation transfer strategies in Vehicle-to-Infrastructure (V2I) and Vehicle-to-Vehicle (V2V) communications, respectively. The authors in [9] integrated computation offloading with load balancing in vehicular networks to maximize system utility. The authors in [10] introduced a cooperative edge caching architecture where vehicles were taken as collaborative caching agents for sharing content caching tasks with base stations. However, the aforementioned edge computing and caching policies are of limited application for dynamic systems since the wireless channel condition, the diverse requirements of specific applications, and the popularity of content are time-varying in practice.

Artificial Intelligence (AI) is a promising technique to address such challenges and can assist in complex resource allocation [11]. Utilizing AI cognitive capability, the authors in [12] proposed an edge cognitive computing architecture which can provide dynamic computing services based on the cognition of users and computation resources to improve energy efficiency and user experience. One of the characteristic features of AI: the ability to interact with the wireless environment, has the potential to facilitate resource management and orchestration [13]. However, these works may be low efficiency for vehicular networks since they ignore the feature of mobility of smart vehicles.

Smart vehicles' mobility has considerable influence on computation and caching resource allocation. The high mobility of smart vehicles will cause frequent handover among edge servers. Furthermore, because of mobility (i.e., time-varying moving speed and direction), the duration that a special vehicle stays in the coverage of a Road Side Unit (RSU) or Base Station (BS) is different. Thus, it is necessary to take mobility into consideration for cooperative edge computing and caching policy design.

In this article, we first propose a novel AI-empowered vehicular network architecture for smart vehicular edge computing and caching. The proposed architecture can intelligently orchestrate edge computing and caching resources to achieve cross-layer offloading, cooperative multi-point caching and

Yueyue Dai, Du Xu, and Guanhua Qiao are with the University of Electronic Science and Technology of China, Chengdu, China (email:yueyuedai@ieee.org; xudu.uestc@gmail.com; qghuestc@126.com).

Sabita Maharjan is with Simula Metropolitan Center for Digital Engineering and the University of Oslo, Norway, (e-mail: sabita@simula.no).

Yan Zhang (corresponding author) is with the University of Oslo, Norway, (e-mail:yanzhang@ieee.org).

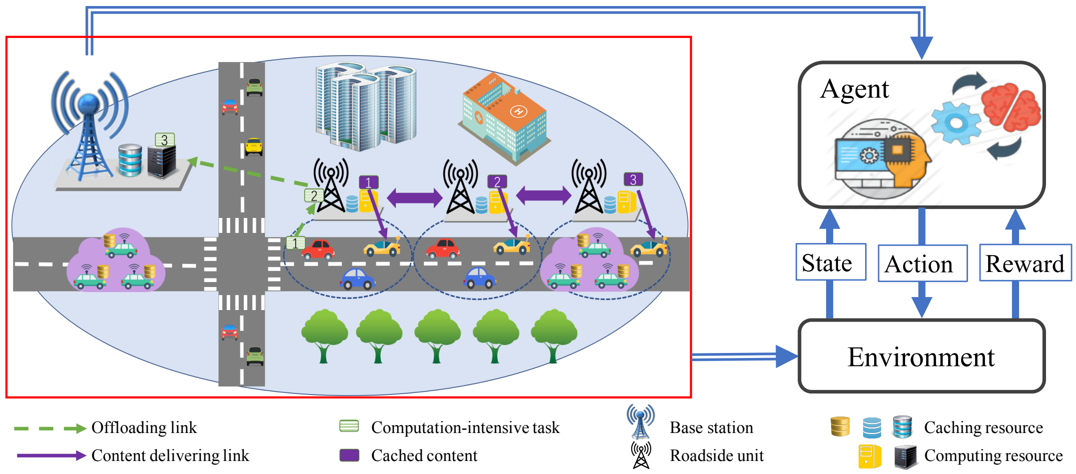

Fig. 1: AI-empowered vehicular network architecture

delivery, and V2V edge caching. Then, we formulate a joint edge computing and caching problem to maximize system utility and exploit the advanced deep reinforcement learning algorithm, Deep Deterministic Policy Gradient (DDPG), to design an efficient resource allocation scheme. Numerical results demonstrate that the proposed scheme can efficiently allocate resources to improve system utility.

II. AI-Empowered Vehicular Edge Network Architecture

The variability in terms of computing and caching capabilities, the diverse requirements of different vehicular applications, time-varying content popularity, and the randomness of duration that a special vehicle stays in the coverage of an RSU or a BS, makes it challenging for vehicular networks to offer such flexibility while also satisfying diverse performance and specific requirements. To address this, we propose a novel AI-empowered vehicular network architecture with cognitive and intelligent abilities for smart vehicular edge computing and caching, as shown in Fig. 1.

In the proposed architecture, RSUs are located along a road and act as edge servers to provide communication, computing, and caching capabilities. Unlike existing RSUs, these RSUs are deployed with AI functions and can provide diverse resources for smart vehicles to implement attractive applications, such as navigation, video streaming, and smart traffic lights. Smart vehicles are key computation task generators and content requesters moving on the road with a certain speed and direction. The computation tasks and requested content of smart vehicles often need to be processed and obtained in time. RSUs and smart vehicles are located within the coverage area of an existing BS. Due to a larger coverage, the BS is usually deployed stronger computing and caching resources than RSUs. Thus, when RSUs cannot support computation requirements of smart vehicles, such computation intensive tasks can be offloaded to the BS. Because of relatively high caching capability, the BS often caches large-size and less popular contents while RSUs often cache small-size and more important contents, such as the latest news and emergency warning.

To exploit AI, we regard the vehicular network as environment and deploy an intelligent agent on the BS. The information of vehicular environment, such as real-time behavior/demand of vehicles and wireless channel condition, can be cognized based on AI methods [12]. The agent can interact with the environment through state, action, and reward. For meeting the requirements of smart vehicles, the agent automatically designs sophisticated actions based on current state, which include cross-layer offloading, cooperative multi-point caching and delivery, and V2V edge caching.

1) Cross-layer Offloading: To provide sufficient computation resources for vehicles, in this architecture, we combine heterogeneous network with MEC to establish a two-tier cross-layer offloading. Heterogeneous network brings small cell base stations, such as access points and RSUs, into cellular networks to form a two-tier cross-layer infrastructure. MEC introduces computing capability to small cell base stations. The two-tier cross-layer offloading is able to process the tasks that demand ultra-low latency and ultra-high computation resources. If the task can be arbitrarily divided into several parts, the two-tier cross-layer offloading will support partial offloading and parallel computing [2]. Specifically, as shown in Fig. 1, a red smart vehicle generates a computation-intensive task with a stringent deadline and the vehicle determines to offload it to the nearest RSU for computing. Since the available computation resources of the RSU is not sufficient to process the task, the RSU further offloads the task to the nearest BS for collaboratively computing. That is, the computation-intensive task is divided into three parts: the first part is processed locally, the second part is processed by the RSU, and the third part is processed by the BS. Since the backhaul link between RSUs and the BS is wired line, the capacity of backhaul link can guarantee a robust cross-layer offloading.

To implement cross-layer offloading, it is necessary to make a fast and dynamic offloading policies. Leveraging in-deep cognition of AI, the proposed architecture can intelligently make offloading decisions and resource allocations for cross-layer offloading. For example, the BS can obtain current states of RSUs and diverse requirements of vehicles in advance by utilizing AI algorithms. Then, the BS can generate the dynamic offloading policy for each vehicle, and specific resource allocations for each RSU and itself based on these parameters.

2) Cooperative Multi-point Caching and Delivery: Caching popular contents at RSUs enables one-hop fast V2R content delivery and the alleviation of backhaul congestion [12]. However, due to the limitation of caching resources, a single RSU cannot store the whole content with a very large size. Since neighboring RSUs can communicate with each other via high-capability wired line, a popular large-size content can be divided into several segments and cooperatively cached at several RSUs. With the help of AI, content features, such as current popularity, potential popularity, storage size, and location are easily learnt.

The proposed architecture facilitates mobility-aware content delivery. A detailed example of content delivery is presented in Fig. 1. The content that the yellow smart vehicle requests can be separated as three segments and distributed cached into three adjacent RSUs. Let us discuss two separate scenarios about how the yellow smart vehicle can obtain its requested content. If the yellow smart vehicle passes all three RSUs, it can receive each segment directly from the corresponding RSU. If it enters into the coverage of the first RSU and then stops, it can receive segment 1 from the first RSU but cannot obtain the other two segments from the second and the third RSUs. To meet the request of the yellow smart vehicle, the first RSU calls for segment 2 and segment 3 via wired line to complete content delivery process.

3) V2V Edge Caching: V2V communication is proposed with the main advantage of enabling multiple direct communication between pairs of near-by vehicles. Since state-of-the-art smart vehicles are equipped with certain caching resources, the accumulative caching power from a group of these vehicles is sufficient to store reasonable amount of content. The cache-enabled vehicles can be utilized as mobile caching servers. Therefore, if a specific vehicle requests a content, it can obtain from a mobile caching server.

V2V edge caching has two potential advantages. Since vehicles can deliver a certain contents from one place to another without any wireless and wired communication, one advantage is that V2V edge caching can save fronthaul and backhaul bandwidth. The other advantage is that V2V edge caching can allow contents to reach the place out of the coverage of RSUs and BSs.

A. State-of-the-art AI Approaches

Convex optimization [2], [4], [5], [7], [9] and game theory [1], [8], [10] have been extensively used to address the edge computing and caching problems. Nevertheless, these existing optimization methods may suffer from the following issues: 1) Some key factors, such as wireless channel condition and content-popularity are assumed to be known. In reality, they are time-varying and difficult to obtain. 2) High mobility of smart vehicles results in dynamic communication topology and complicated correlation between vehicular communication pairs. As a result, efficient and reliable data transmission in vehicular networks becomes even more challenging; 3) Most of the proposed algorithms are optimal or close-to-optimal only for a snapshot of the system without considering the long-term effect of the current decision on resource allocation.

AI is a promising approach to facilitate in-depth feature discovery such that the uncertain input fields for edge computing and caching problems can be obtained in advance. In addition, AI is also a promising tool to tackle complex optimization problems, such as resource allocation [12], [13]. Deep reinforcement learning is a branch of AI research where an agent learns by interacting with environment. There are three common deep reinforcement learning algorithms: Q-learning, Deep Q Network (DQN), and DDPG:

Q-learning: Q-learning is a classical model-free and off-policy deep reinforcement learning algorithm based on bellman equation. It iteratively approximates a Q-function $Q(s,a)$, where s, a are the state and action, respectively. The agent of Q-learning needs to compute the Q-function of each state-action pair and stores its result, named Q-value, into a unique Q-table. Q-learning is a simple incremental algorithm developed from dynamic programming. But it is not an ideal algorithm for the problem with a high-dimensional observation space, since the possible states of a system may be more than ten thousands making it difficult to store all Q values of states into a Q-table.

DQN: DQN is a kind of deep reinforcement learning algorithm which uses deep neural networks instead of Q-function to explore actions [14]. DQN has two major factors to make it powerful, namely experience replay and fixed Q-target. Experience replay is a method to destroy the dependence of collected experience, allowing the network to learn more from past experience and fixed Q-target. DQN is a powerful tool that can learn optimal policies with high-dimensional observation spaces. However, applying DQN to a continuous domain is still a challenge since it relies on finding the action that maximizes action-value function.

DDPG: DDPG is an actor-critic and model-free algorithm that can learn policies in high-dimensional, continuous action spaces [15]. DDPG combines the actor-critic approach with DQN. The actor network is used to explore policy and the critic network is used to estimate the performance of the proposed policy. To improve the learning performance, techniques such as experience replay and batch normalization of DQN are employed. Compared to the other two algorithms, the most significant feature of DDPG is that it can make decisions or allocations in a continuous action space while Q-learning and DQN only can support the problem with a discrete action space.

B. Design Challenges

The proposed AI-empowered vehicular network utilizes AI approaches to perform in-deep cognition and intelligent orchestration for vehicular edge computing and caching. However, AI has mostly been studied in the area of computer vision and data processing, while little attention has been paid to the vehicular network and MEC system. In order to realize the proposed architecture, there are still some challenges to be addressed.

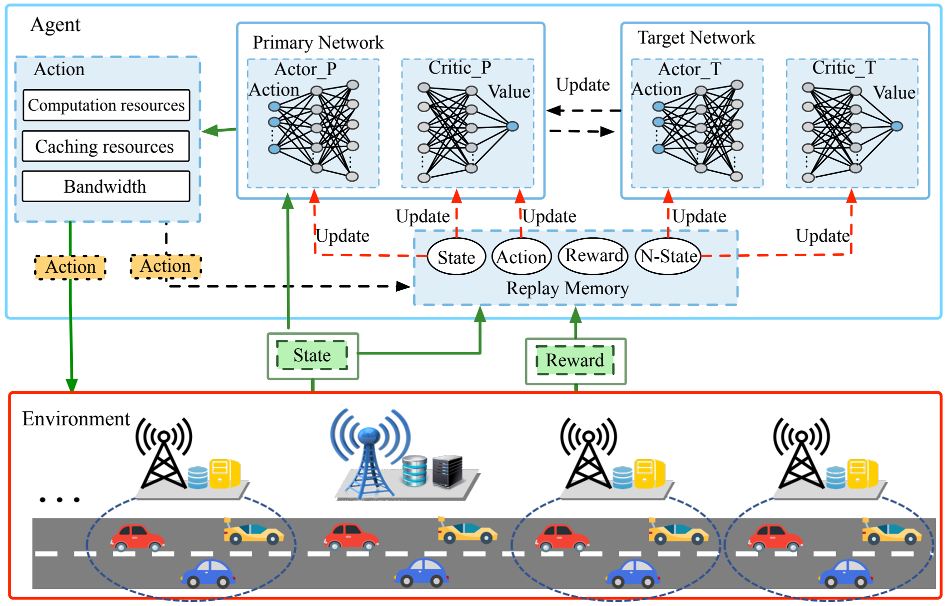

Fig. 2: The process of DDPG to solve joint vehicular edge caching and computing problem

One of the key architecture principles is to utilize AI to intelligently monitor vehicular network and manage resources. However, because of mobility, handovers between smart vehicles and RSUs are quite frequent which make edge computing and caching resource management even more challenging. Since a computation task or content needs to be divided into several segments to support computation offloading or edge caching, how to divide each one of them to improve system utility while satisfying diverse requirements of vehicles should be taken into policy design. Furthermore, interference management also should be incorporated for a wireless communication scenario.

Since lots of uncertain and time-variant states need to be well predicted, AI algorithms should store plenty of historical data for accurate prediction. This behavior will consume a certain amount of caching resources. Since the caching resources are limited and mainly used to popular content caching, there exists a trade-off between prediction accuracy and content caching.

III. DDPG-Empowered Vehicular Edge Computing and Caching Scheme

A. Problem Formulation

The joint vehicular edge computing and caching problem can be formulated as an optimization problem to maximize system utility. In this problem, system utility consists of computing utility, caching utility, and cost. Specifically, computing utility is equal to $\Delta t_{ij} \cdot B_i^C$, where Δt_{ij} is the saving time when edge server j fulfills computation task i before its deadline, and B_i^C is the price that smart vehicle i pays to edge server j. If edge server j completes the task before its deadline, the saving time is positive and the corresponding computing utility is also positive. Otherwise, both saving time and computing utility are negative. This is reasonable because for a delay-sensitive task, it may result in a certain loss if it is not completed before its deadline. Caching utility is equal to $\rho_i \cdot T_{ij} \cdot B_i^S$, where T_{ij} is the caching duration that edge server j caches content i, B_i^S is the price to request the content, and ρ_i is the popularity of content i. The cost is the energy consumptions on communication, computing, and caching, respectively.

Exploiting the proposed architecture, the information in terms of computing and caching capabilities, the requirements of different vehicular applications, content popularity, mobility parameters, can be collected and sent to the agent. Here, we consider three mobility parameters, i.e., location, speed, and direction. Location can be utilized to determine the nearest base station for a specific vehicle. Speed and direction are formulated as the sojourn time that a specific vehicle will stay in the coverage of its nearest base station.

After collecting the above information, the agent designs an action to perform resource allocation. There are three key elements in the deep reinforcement learning, namely state, action, and reward:

1) State: The state in deep reinforcement learning is a space to reflect the situation of vehicular environment. The state consists of four components $S = (D_i, F_j, G_j, B_j)$. D_i is the state of smart vehicle i, which includes the size of its computation task, the size of the requested content, the required computation resources, the required cache resources, deadline, popularity of the requested content, and mobility parameters. F_j, G_j, and B_j are the available computation resources, available caching resources, and available bandwidth of edge server j, respectively.

2) Action: The objective of an agent is to map the space of states to the space of actions. In this system, the action consists of three parts: f_{ij}, g_{ij}, and b_{ij}, which are the amount of computation resources, caching resources, and bandwidth that the BS or the RSU allocates to smart vehicle i, respectively.

Note that f_{ij}, g_{ij}, and b_{ij} are continuous values here.

3) Reward: Based on current state and action, the agent obtains a reward from the environment. Since reward function is related to the objective function, in this scenario, system utility can be regarded as the reward function.

B. DDPG-empowered Vehicular Edge Computing and Caching Scheme

Since the action space of the joint vehicular edge computing and caching problem contains plenty of continuous variables, we propose a DDPG-based vehicular edge computing and caching resource allocation, as shown in Fig. 2.

Three components constitute the functioning of the DDPG algorithm (i.e., primary network, target network and replay memory.)

The primary network consists of two deep neural networks, namely an actor network and a critic network. The actor network is used to explore the policy, which specifies the current policy by deterministically mapping state to a specific action. The critic network estimates the performance and provides the critic value which helps the actor to learn the gradient of the policy. Thus, the input of the primary network is the current state from vehicular environment, the training state and the training action from replay memory. The output of the primary network is the action which is adopted by the BS, RSUs and smart vehicles.

The target network can be defined as an old version of the primary network, which is used to generate the target value for training Critic-P. It includes a target actor network and a target critic network. The input of the target network is the next state (i.e. N-State) from replay memory and the output is a critic value for training Critic-P.

The replay memory stores experience tuples which include current state, the selected action, reward, and next state. The stored experience tuples can be randomly sampled for training primary network and target network. Randomly sampled experience tuples aim to reduce the effects of data correlation.

In DDPG, the vehicular environment consists of a BS, RSUs and vehicles, as shown in Fig. 2. The agent is at the BS to design action and it will send the designed action to RSUs and vehicles. Based on the action, the BS, RSUs, and vehicles cooperatively execute edge computing and caching. Then, vehicles give feedback to reward the received services. The detailed processes of DDPG works as follows.

First, the state of vehicular environment is sent to the Actor-P of the primary network and replay memory.

Second, based on current state and experience tuples, the agent uses primary network and target network to determine next action. The action includes three continuous values (i.e., computation resources, caching resources, and bandwidth). In this step, Actor-T and Critic-T in target network update target policy value $\theta^{\mu'}$ and target Q value $\theta^{Q'}$ based on experience tuples from replay memory, respectively. Then, Critic-T in target network sets $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$ and transmits it to Critic-P in primary network. After receiving y_i, Critic-P in primary network updates primary Q value θ^Q by solving an optimization problem which aims to minimize the loss function $Ls(\theta^Q) = (y_i - Q(s_i, a_i|\theta^Q))^2$. Based on θ^Q and experience tuples, Actor-P in primary network makes policy gradient and then generates next action.

Third, the BS, RSUs, and smart vehicles, based on the determined action, execute edge computing and caching. After executing resource allocation, each state of the BS, RSUs, and smart vehicles is transited from one to another, respectively. Then, the environment feedbacks an immediate system reward to the agent, based on the new state. If current resource allocation satisfies all constraints of the proposed joint edge computing and caching problem, and the system utility of current resource allocation is greater than the existing maximal system utility, the immediate system reward is updated as current system utility and the environment updates its state based current resource allocation. If current resource allocation satisfies all constraints of the proposed problem but the system utility of current resource allocation is smaller than the existing maximal system utility, it indicates that DDPG does not generate a better solution such that the immediate system reward and the state of the environment are updated based on the existing best resource allocation policy. If current resource allocation cannot satisfy any constraint of the proposed problem, the agent will receive a penalty. There exists a long-term reward in DDPG which can be denoted as the cumulative system utility. When the cumulative system utility converges, the optimal resource allocation is successfully trained.

IV. Numerical Results

We evaluate the performance of the proposed AI-empowered vehicular network architecture with joint edge computing and caching through extensive simulations. The proposed architecture implements two parts, i.e., the environment and the agent. In the environment, there are one BS and 4 RSUs located along a unidirectional road. The computation resources of the BS and RSUs are 10 GHz and 5 GHz, respectively. The caching capabilities of the BS and RSUs are 20 GB and 10 GB, respectively. The bandwidth of the BS and RSUs are 20 MHz and 10 MHz, respectively. There are 20 smart vehicles travelling in the same direction with speed $v_i = 80$ km/hr. The size of computation tasks and the size of contents are randomly taken from $[0.15, 0.25, 0.3, 0.4, 0.45, 0.6]$ GB. The content-popularity is uniformly distributed in $U\ [1, 6]$. The required computation resource of computation tasks is randomly taken from $[0.5, 0.6, 0.7, 0.8, 1.2]$ GHz. The penalty is -1000. The agent is implemented using Python and TensorFlow.

Fig. 3 shows the comparison of cumulative average system utility of different schemes using DDPG algorithm. We can draw several observations from Fig. 3. First, the system utility of the proposed joint computing and caching scheme is obviously the highest compared to the other two baseline schemes. The reason is that, in DDPG-based joint computing and caching scheme, computation resources, caching resources, and bandwidth of the BS and RSUs are fully utilized while DDPG-based computing scheme only optimizes computation resources and bandwidth and DDPG-based caching scheme only optimizes caching resources and bandwidth. Second, we

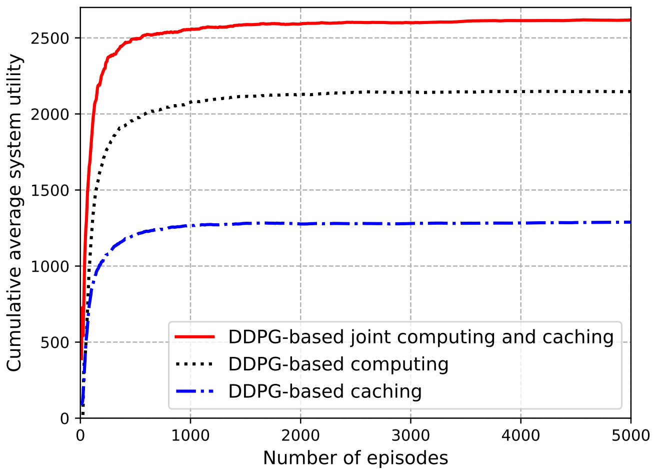

Fig. 3: Comparison of cumulative average system utility under different schemes.

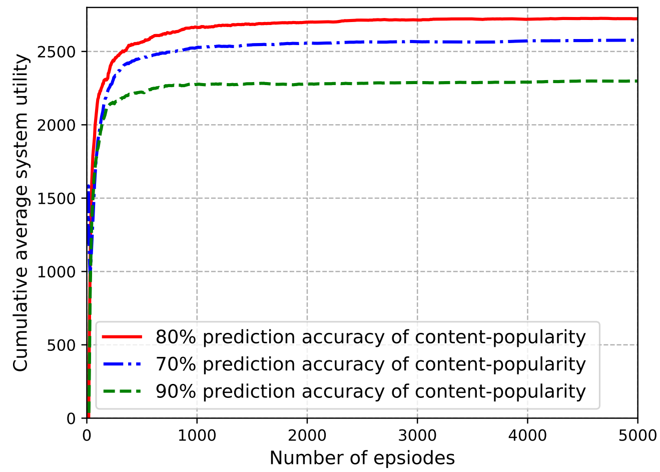

Fig. 5: Comparison of cumulative average system utility under different content-popularity prediction accuracy.

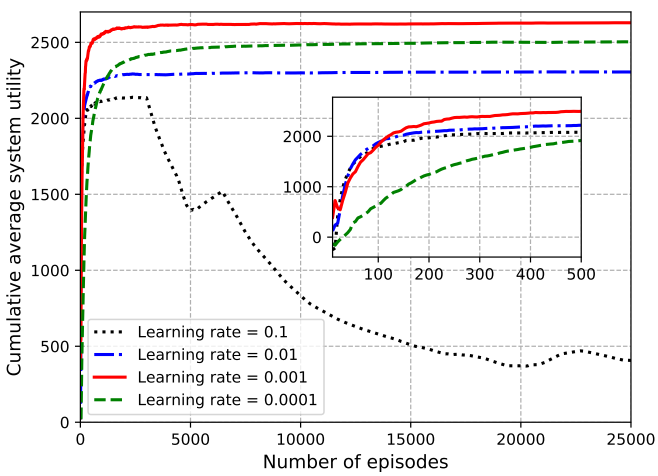

Fig. 4: Convergence performance under different learning rates.

observe that the cumulative average system utility based on DDPG is very low at the beginning of the learning process. With the number of episodes, the system utility increases dramatically and then converges after running 2000 episodes. The reason is that, at the beginning of the learning process, because of lacking historical information, DDPG is easy to explore an inefficient resource allocation strategy such that the system utilitity is low. After training for a while, based on the learned information, DDPG can explore a better resource allocation strategy faster and earier, such that the system utility increases dramatically. The convergence of the system utility implies that the agent on the BS has learnt the best resource allocation strategy.

Fig. 4 shows the convergence of the proposed scheme under different learning rates. First, the cumulative average system utilities achieve convergence when the learning rate is 0.01, 0.001, and 0.0001, respectively. However, when the learning rate is 0.1, the convergence performance dramatically decreases after running 3000 episodes iterations. The reason is that 0.1 is a too large learning rate, which results in a large update step such that the system utility converges to a local optimal solution. Second, the performance of learning rate 0.001 is obviously better than the performance of learning rate 0.01 and 0.0001. Therefore, 0.001 is the best learning rate for the proposed scheme. In fact, an appropriate learning rate depends on the architecture of the mode being optimized, as well as on the state of the environment in the current optimization process [13].

We consider that each edge server has to utilize 40%, 20%, and 5% of its caching resource to store the past experience of user preference for achieving 90%, 80%, and 70% prediction accuracy of content-popularity, respectively. Fig. 5 shows the comparison of cumulative average system utility under different content-popularity prediction accuracy. We can see that when the prediction accuracy is 80%, the system utility is the highest compared to the cases that prediction accuracy is 90% and 70%. Thus, with the limitation of caching resources, it is necessary to make a better trade-off between resources for prediction accuracy and resources for content caching to improve system utility.

V. Conclusion and Future Work

In this article, we first propose an AI-empowered vehicular network architecture which can intelligently orchestrate edge computing and caching to enable cross-layer offloading, cooperative multi-point caching and delivery, and V2V edge caching. In addition, we formulate a joint edge computing and caching problem to maximize system utility, which is solved by a novel deep reinforcement learning approach, named DDPG. Utilizing DDPG, the proposed scheme allocates resources quickly and efficiently. Numerical results are presented to validate the effectiveness of the proposed scheme.

In our future work, AI will be leveraged to make more elaborate and intelligent management. First, we plan to utilize AI algorithms to mine the relationship between wireless channels and network configurations for guiding channel assignment, power control, and interference management. Further, we will utilize AI algorithms to forecast potential handover and make pre-allocation of bandwidth. Moreover, since some popular contents are shared by V2V edge caching, we plan to propose a secure caching and sharing mechanism to enhance security and user privacy.

ACKNOWLEDGEMENT

This research has partially received funding from the National key Research and Development Program of China-No. 2016YFB0800105; the European Unions Horizon 2020 research and innovation programme under the Marie Skodowska-Curie grant agreement No 824019; the projects 240079/F20 funded by the Research Council of Norway; the projects 18H86301ZT00100303; the 111 project (B14039).

REFERENCES

[1] K. Zhang, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Mobile edge computing and networking for green and low-latency internet of things," *IEEE Commun. Mag.*, vol. 56, no. 5, pp. 39–45, May 2018.

[2] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, "Joint computation offloading and user association in multi-task mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 12, pp. 12 313–12 325, Dec 2018.

[3] C. X. Mavromoustakis, J. M. Batalla, G. Mastorakis, E. Markakis, and E. Pallis, "Socially oriented edge computing for energy awareness in iot architectures," *IEEE Commun. Mag.*, vol. 56, no. 7, pp. 139–145, 2018.

[4] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 587–597, 2018.

[5] Z. Ye, C. Pan, H. Zhu, and J. Wang, "Tradeoff caching strategy of outage probability and fronthaul usage in cloud-ran," *IEEE Trans. Veh. Technol.*, 2018.

[6] A. Chattopadhyay, B. Błaszczyszyn, and H. P. Keeler, "Gibbsian on-line distributed content caching strategy for cellular networks," *IEEE Trans. Commun.*, vol. 17, no. 2, pp. 969–981, 2018.

[7] M. Chen, Y. Hao, L. Hu, M. S. Hossain, and A. Ghoneim, "Edge-cocaco: Toward joint optimization of computation, caching, and communication on edge cloud," *IEEE Commun.*, p. 2, 2018.

[8] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading," *IEEE Veh. Technol. Mag.*, vol. 12, no. 2, pp. 36–44, 2017.

[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.*, pp. 1–1, 2018.

[10] K. Zhang, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Cooperative content caching in 5g networks with mobile edge computing," *IEEE Commun.*, vol. 25, no. 3, pp. 80–87, JUNE 2018.

[11] P. Valente Klaine, M. A. Imran, O. Onireti, and R. D. Souza, "A survey of machine learning techniques applied to self organizing cellular networks," *IEEE Commun. Surv. Tutor.*, 2017.

[12] M. Chen, W. Li, G. Fortino, Y. Hao, L. Hu, and I. Humar, "A dynamic service-migration mechanism in edge cognitive computing," *ACM Trans. Internet. Technol., https://arxiv.org/pdf/1808.07198, 2018.*

[13] Y. He, F. R. Yu, N. Zhao, V. C. Leung, and H. Yin, "Software-defined networks with mobile edge computing and caching for smart cities: A big data deep reinforcement learning approach," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 31–37, 2017.

[14] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[15] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *Proceedings of ICLR*, 2016.

Yueyue Dai [S17] (yueyuedai@ieee.org) received the B.Sc. degree in communication and information engineering from the University of Electronic Science and Technology of China (UESTC), in 2014, where she is currently pursuing the Ph.D. degree. Since October 2017, she has been a visiting student with the Department of Informatics, University of Oslo, Norway. Her current research interests include wireless network, mobile edge computing, Internet of Vehicles, blockchain, and deep reinforcement learning.

Du Xu [M17](xudu.uestc@gmail.com) is a professor at University of Electronic Science and Technology of China (UESTC) in Chengdu, China. He received a B.S., an M.S. and a Ph.D. from the South-East University and UESTC in 1990, 1995 and 1998, respectively. His research interests include network modelling and performance analysis, switching and routing, network virtualization and security. He presided over many advanced research projects, including NSFC, National 863 Plans and National key Research and Development Program of China.

Sabita Maharjan [M09] (sabita@simula.no) received her Ph.D. degree in networks and distributed systems from the University of Oslo and Simula Research Laboratory, Norway, in 2013. She is currently a senior research scientist in Simula Metropolitan Center for Digital Engineering (CDE), Norway, and an associate professor at the University of Oslo. Her current research interests include wireless networks, network security and resilience, smart grid communications, the Internet of Things, machine-to-machine communications, software defined wireless networking, and the Internet of Vehicles.

Guanhua Qiao (qghuestc@126.com) is currently working toward the Ph.D. degree at the School of Information and Communication Engineering, University of Electronic Science and Technology of China (UESTC). His current research interests include mobile edge computation and caching, deep reinforcement learning-based resource management and network optimization.

Yan Zhang [SM10] (yanzhang@ieee.org) is a full professor at the University of Oslo. He is an Editor of several IEEE publications, including IEEE Communications Magazine, IEEE Network, IEEE Transactions on Green Communications and Networking, IEEE Communications Surveys & Tutorials, and IEEE Internet of Things. His current research interests include next generation wireless networks leading to 5G and cyber physical systems. He is an IEEE VTS Distinguished Lecturer and a Fellow of IET.