Survey on Reinforcement Learning Applications in Communication Networks

Yichen Qian, Jun Wu, Rui Wang, Fusheng Zhu, Wei Zhang

***Abstract*—In recent years, intelligent communication has drawn huge research efforts in both academia and industry. With the advent of 5G technology, intelligent wireless terminals and intelligent communication networks are increasingly under intensive study. Artificial intelligence enhances the network capability with automatic and adaptive adjustment. Reinforcement learning (RL) and deep reinforcement learning (DRL) are two powerful techniques in artificial intelligence which can learn the optimal decision according to the environment feedback. In this paper, we focus on the latest research progress on RL and DRL applied in three emerging technologies including mobile edge computing (MEC), software defined network (SDN) and network virtualization in 5G. The prospect of further research and development in the future is preliminarily forecasted.**

***Keywords*—5G, intelligent communication, reinforcement learning, mobile edge computing, software defined network, network virtualization**

I. Introduction

The emerging technologies such as virtual reality (VR) and augmented reality (AR) have demanded lower latency and higher throughput in the fifth generation (5G) communications. Compared to 4G, 5G's transmission rate is increased by 10∼100 times, peak transmission rate reaches 10 Gbit/s, end-to-end delay is reduced to 1 millisecond, and traffic density is increased by 1 000 times. Besides, more applications are expected in 5G. There are three major application scenarios in 5G: enhanced mobile broadband (eMBB), ultra reliable low latency communications (URLLC) and massive machine type communications (mMTC). Each of them has different user demands and network requirements. Therefore, a more flexible and adaptive network is required to provide services with different quality of service (QoS). Mobile edge computing (MEC), software defined network (SDN) and network virtualization are three promising technologies for 5G to provide lower latency and more flexible services.

To address the high latency problem in conventional network, MEC provides the storage and computation capabilities at the edge of the mobile network near the users[1,2]. For some latency-critical services, the mobile edge network can directly give responses to users without going through the core network which may take a long time to respond. It not only shortens the transmission distance to lower the latency, but also relieve the backhaul traffic load. Storage capability gives edge network the ability to cache, so as to serve users with lower latency and decrease the redundant backhaul transmission. Computation capability enables users to offload their computation tasks to the edge network. With the much more powerful computation capability, the computation delay for users will be tremendously decreased and therefore improve user experience.

SDN is an emerging networking paradigm which simplifies the network management[3,4]. In traditional networks, the control plane and the data plane are merged together in the network devices. If a change of configurations is needed, all the network devices need to be reconfigured. So it is difficult to manage all devices in the whole network. On the other side, SDN separates the control plane from the data plane into a centralized controller. All the network controls like routing and scheduling are managed by the controller. The switches are only in charge of forwarding the packages according to the rules given by the controller. When configurations need to be changed, the network administrator only needs to rewrite the control algorithm at the central controller. Then all the switches will have the new configuration and follow the new forwarding rules. With the centralized control of SDN, the

Manuscript received May 09, 2019; accepted Jun. 13, 2019. This work was supported in part by the National Science Foundation China under Grant 61831018, Grant 61571329, Grant 61631017, Grant 61762053, and Guangdong Province Key Research and Development Program Major Science and Technology Projects under Grant 2018B010115002. The associate editor coordinating the review of this paper and approving it for publication was W. C. Cheng.

Y. C. Qian, J. Wu, R. Wang. Tongji University, Shanghai 201804, China (e-mail: 92yichenqian@tongji.edu.cn; wujun@tongji.edu.cn; ruiwang@tongji.edu.cn).

F. S. Zhu. Guangdong New Generation Communication and Networks Innovative Institute, Guangzhou 519165, China (e-mail: zhufusheng@gdcni.cn).

W. Zhang. University of New South Wales, Sydney NSW 2052, Australia (e-mail: wzhang@ee.unsw.edu.au).

network becomes more flexible. The configurations can be easily changed according to the different application requirements and user demands in 5G.

Network virtualization is proposed as an integral part of the next-generation networking paradigm which gives a new perspective to flexibly allocate the network resources[5,6]. It decouples the roles of the traditional Internet service providers (ISPs) into two independent entities: infrastructure providers (InPs) and service providers (SPs). InPs are in charge of managing the physical infrastructure, while SPs are in charge of aggregating resources from multiple InPs to create virtual networks (VNs) and also providing the end-to-end services. With this change, multiple VNs can share resources, and network administrators can reallocate the resources according to different user requirements.

Recent years, the intelligent network has been proposed by introducing artificial intelligence into network management. With artificial intelligence, the network can autonomously and adaptively control and manage itself. Reinforcement learning (RL)[7] is a typical artificial intelligence technique which fits for the network environment. With the huge development of computation capability brought by GPU, deep reinforcement learning (DRL) is further developed to improve artificial intelligence. Many works have been done on how to apply RL and DRL into the network management for more proper and accurate control. In this paper, we mainly survey the application of RL and DRL in MEC, SDN and network virtualization respectively, including network caching, task offloading, routing scheduling and resource allocation. We also provide a better understanding of why RL and DRL can help the network management. Prospect of designing intelligent networks using reinforcement learning is further given.

II. Background of Reinforcement Learning and Deep Reinforcement Learning

RL is an artificial intelligence technique which learns how to map situations to actions, so as to maximize a numerical reward signal[8]. Basically, RL follows the concept of Markov decision process (MDP) which is a general framework for modeling decision-making problems. An MDP can be described as $M = (S, A, R(s,a), p(s',r|s,a), \gamma)$, where S and A indicates a finite state space and a finite action space respectively. $R(s,a)$ denotes the reward function of states and actions. $p(s',r|s,a)$ is the state-transition probability from the given state $s \in S$ and action $a \in A$ to the next state $s' \in S$ and the corresponding reward $r \in R$. $\gamma \in [0,1]$ is the discount rate reflecting the importance of the current reward compared to the future rewards. As shown in Fig. 1, for each time step t, the agent observes the current system state s_t and chooses an action a_t. The environment will then return the immediate reward r_{t+1} and the next-step state s_{t+1}. The agent's goal is to find a policy $a = \pi(s)$ which determines the action selection, so as to maximize the accumulated reward. To indicate the average accumulated reward an agent can get under state s, value function $V_\pi(s)$ is introduced as $V_\pi(s) \doteq \mathbb{E}_\pi\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\right\}$. According to the recursive relationship of $V_\pi(s)$, we have the Bellman equation as

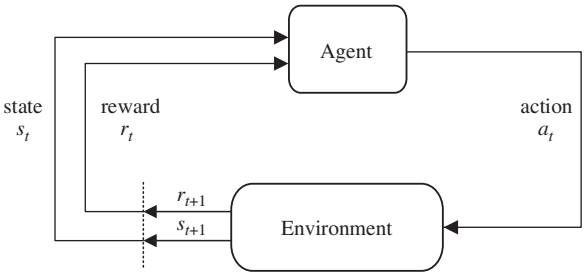


Figure 1 The agent-environment interaction in a Markov decision process

$$V_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma V_\pi(s')\right]. \quad (1)$$

Based on the Bellman equation, dynamic programming could be applied to solve the MDP when the state-transition probability $p(s',r|s,a)$ is known in advance. However, in most practical applications, the state-transition probability can not be known.

To deal with the MDP problem with unknown state-transition probability, various algorithms have been proposed. Q-leaning is one of the typical learning algorithms based on temporal difference algorithm[9]. To describe the Q-learning algorithm, we first extend the concept of value function to action-value function (or named Q-function) by including actions as $Q_\pi(s,a) \doteq \mathbb{E}_\pi\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a\right\}$. And we define the optimal Q-function with the optimal policy as $q_*(s,a) \doteq \max_\pi q_\pi(s,a)$. Similarly, we can have the Bellman equation for the optimal Q-function as

$$Q_*(s,a) = \sum_{s',r} p(s',r|s,a)\left[r + \gamma \max_{a'} Q_*(s',a')\right]. \quad (2)$$

In Q-learning, the action selection is based on ε-greedy algorithm which has ε probability to choose a random action and $1-\varepsilon$ probability to choose the greedy action with the largest Q-value. Q-value is initialized with a given value and iteratively updated during the evolving of action selection. The update of Q-value is given by

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma \max_a Q(s',a) - Q(s,a)\right], \quad (3)$$

where α is the learning rate.

As can be seen, the key to Q-leaning algorithm is to maintain a Q-table which stores the Q-values of state-action pairs.

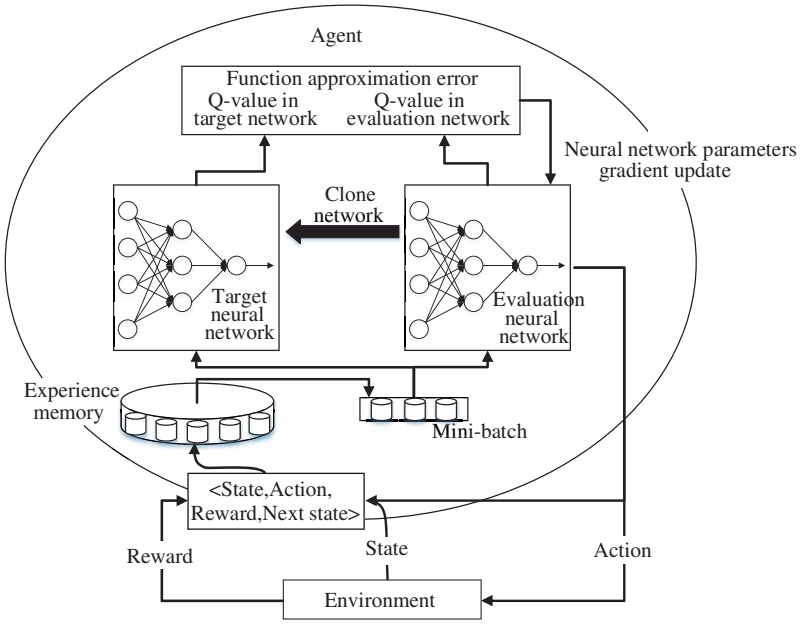


Figure 2 The framework of DQN

A limitation of this approach is that when the dimension of states grow up, the time complexity of looking-up Q-table and space complexity of storing Q-table will both exponentially increase. Q-value function approximation is then proposed by using a linear function to approximate Q-value[10]. So that Q-value can be directly calculated with the given state and action. Apparently, the linear function approximation can not accurately estimate the value function. In order to solve the problem, deep reinforcement learning (DRL)[11] was proposed with the help of deep neural network (DNN). The main idea is to use DNN which appears to be non-linear function to approximate the Q-function. Among all learning algorithms in DRL, deep Q-network (DQN)[12] was first proposed as a remedy of Q-learning. Fig. 2 shows the framework of DQN[13]. Besides the deep neural network, replay memory and separated target Q-network are further introduced to reduce the correlation between data for more stable convergence. Both experience replay and separated target network motivate to choose the off-policy Q-learning, since the sampling policy is only contingent on previously trained Q-value DNN and the updating policy is irrespective of the sampling policy. On the other hand, the DQN agent could collect the information (i.e., state-action-reward pair) and train its policy in background. Thus, the DQL could efficiently perform and timely make the decision according to its already learned policy.

There are many other DRL algorithms which have the similar idea but different purpose. For example, deep deterministic policy gradient (DDPG) is designed for the continuous action space[14] and asynchronous advantage actor-critic (A3C) mainly focuses on synchronous parallelism learning[15]. Due to the limited space, we will not introduce their details here.

III. Application of AI in MEC

The applications of RL and DRL in MEC are first presented. We focus on the topics of network caching and task offloading as they are the key milestones in MEC. We will also discuss how RL and DRL can help to solve the issues in MEC.

A. Network Caching

MEC architecture gives BSs the ability to cache, so how to effectively utilize the cache becomes a research hotspot. Caching the proper contents not only can shorten the transmission distance to reduce the latency, but also can avoid redundant transmission from content providers to the BS when different users request the same content.

Knowing users' future requests is the key to designate an effective cache policy. For this purpose, many cache policies are designed to increase the cache hit rate according to the users' requests. The least recently used (LRU) policy and the least frequently used (LFU) policy[16] are two heuristic cache replacement strategies to improve the cache hit rate. Also, the tidal effect of the network traffic gives the opportunities to proactively push the content when the network is idle. With pushing opportunities, many other works[17-20] optimize the long-term cache content based on the statistical information and use the priori knowledge of file popularity as a criterion of joint pushing and caching policy. In real world, the file popularity is hardly known and cannot be directly used. Blasco et al. proposed a learning-based method, which can automatically learn the file popularity and optimize the cache content[21]. However, these works developed cache policies

based on the stable file popularity, and cannot adjust the cache policy according to the real-time user requests. In many practical applications, user requests often show temporal correlations such as news on a website and episodes of a TV show. Based on this feature, an explicit model is built for web traces to catch the correlations of user requests[22]. It also gives some trails of designing cache replacement policies. But the stable model can be inaccurate when fitting the dynamic changes. In addition, different users may have different habits which results in different request models. A general model can hardly fit all users.

In order to adjust the cache content according to the real-time user requests, Huang et al.[23] considered online learning-aided cache policy, which can be adaptive to instantaneous content requests. The system can learn the optimal cache content without priori knowledge of statistical information of content requests. The system decides the pushing content based on the system resource state and the current user request. The paper defines a reward to indicate the user's quality of experience (QoE). When the user's request content in the next time slot is proactively pushed and cached, the QoE increases since the user will be served in low latency. The proposed RL based cache policy tries to predict the future user requests and maximize the average reward obtained by proactively serving user demands. The paper also provides a guideline to design future smart systems. However, the complexity of RL based method increases tremendously when the size of system states grows up. Therefore, the proposed method cannot be applied when the number of files is huge.

To deal with the dimensionality problem, Wei et al.[24] introduced DRL to do the joint scheduling and cache management for mobile edge network. The paper considered the scenario consisting of one macro-cell base station (MBS) and several small-cell base station (SBS). All BSs are equipped with cache ability and can serve all users. In each time slot, the MBS gets content requests from users along with the signal-to-interference-plus-noise ratio (SINR) between each user and each base station. The MBS needs to decide which base station to serve certain content and whether to save the content. The objective is to minimize the average total transmission delay. The paper applies actor-critic DRL to lower down the dimension of inputs. It uses content popularity, cache state, SINR as the input state, MBS's decision as the output action and the average total transmission delay as the feedback reward.

All works mentioned above considered the transmission to be unicast. Sun et al.[25] further considered multiuser MEC scenario with multicast. One BS with attached server is considered in the network which serves a bunch of users. Each user has a fixed size of cache. In each time slot, each user will submit a request to the BS and must be served before the end of the slot. The base station broadcasts the requested contents of all users and determines whether to push some contents to users in ahead of time or not. The users then update their cache according to the contents broadcasted by the BS. The paper defines an average transmission cost to indicate the bandwidth utilization. Lower average cost refers to higher bandwidth utilization. The paper also considered the user requests to follow Markov chain and formulated the stochastic optimization problem as an infinite horizon average cost MDP. User demands and cache states are considered to be the input states. Pushing contents and cache updates are considered as the output actions. Average transmission cost is used as the reward. To cope with the dimensionality issue, the authors separated the joint optimization problem into two sub-problems and applied per-user per-file value function approximation Q-learning to solve the problem. With this approximation, the dimension of both states and actions were significantly decreased. The numerical results demonstrate that the proposed method outperforms other joint pushing and caching policies, including heuristic policies and model based policies.

The biggest challenge in cache management is the uncertainty user's future requests. Therefore, a better knowledge of user's future request will lead to a better cache management design. So far there is no heuristic algorithm or explicit model that can properly predict future requests for different users. However, in some scenarios, the requested files show temporal correlations. Hence RL and DRL are introduced to solve the problem. Simulation results show that RL and DRL can predict users' future requests in relatively high accuracy.

B. Task Offloading

With the emergence of the VR/AR technology, an explosive growth of the computation intense mobile applications has been shown. The limited computation capabilities on mobile phones can affect the user experience in both high latency and rapid power consumption. As a remedy to this problem, task offloading has been proposed to relieve the computation workload on mobile phones. The mobile phones can offload the task to the cloud to help computing and the cloud will send back the results when the task is finished. With the help of offloading, the computation capability is enhanced and the mobile power consumption can be tremendously decreased. But the long-distance transmission between users and the cloud will cause a long transmission latency. Thus, MEC architecture is a suitable scenario for offloading. The edge computing near to the users provides both strong computation capabilities and short communication latency. As a result, there is a wide application of task offloading in MEC.

There are many existing works on offloading optimizations based on stable environment or with the knowledge of channel condition. However, in real cases, dynamic changes in the environment (such as mobilities, computation capabilities and caching states) and the channel condition can only be known

after the transmission. Due to these changes and uncertainties of the environment, stable model and optimizations can hardly get optimal results. To achieve better performance, we need accurate prediction of the environment changes. To this end, RL is introduced to learn and predict the change of the environment and choose an appropriate offloading strategy.

Xiao et al.[26] proposed an RL based offloading strategy for mobile offloading against smart attacks. The user offloads the tasks to the access point (AP), while there is a smart attacker tries to interfere the transmission. The user needs to carefully choose the offloading rate in order to mitigate the interference of the attacker. The environment is considered as dynamic with unknown attack cost, channel gain and detection accuracy. The paper formulated the problem as a secure mobile offloading game and applied Q-learning to solve it. Experimental results show that by using Q-learning, the offloading rate and security are increased by 86 and 6 percent, respectively, compared to a random offloading scheme. However, both time complexity and space complexity will rapidly increase when the size of system states grows up. Thus, the solution is unable to extend to large state space.

Chen et al.[27] investigated a DRL based dynamic computation offloading strategy in MEC network to minimize the energy consumption while completing tasks within an acceptable delay. The paper considered the MEC network consisting of one BS with attached server and multiple mobile users, where tasks arrive stochastically and channel condition is time-varying for each user. In each time slot, the system determines the power allocation of for both local execution and computation offloading. To address the continuous space of power, DDPG is adopted to learn efficient computation offloading policies. By observing user's local states of buffer length and channel condition, powers of both local execution and task offloading can be adaptively allocated using the learned policy. Numerical results demonstrate that DDPG based policy outperforms DQN based discrete power control policy with reduced computation cost.

The aforementioned works mainly considered users as stationary for a long time. Tan et al.[28] further considered the mobility of users in vehicle network. The users can offload tasks to the road side units (RSUs) or to the MEC servers when there is no RSU nearby. The dynamic changes of downlink channel conditions, computation capabilities, caching states and vehicular mobility pose great challenges for conventional method. A deep Q-learning framework was then proposed to address the dynamics of the environment. The system states include the available RSU/MEC servers, vehicles, caches and also the number of contacts. In each time slot, the system needs to decide which RSU and/or vehicle should be assigned to the requesting vehicle, whether the requested content should be cached and whether the computation task should be offloaded. The objective is to minimize the cost of communication, storage and computation. One limitation of the work is that the proposed framework is a centralized method, which needs to control all vehicles in the network. Thus, the computation complexity will exponentially grow when the network scale becomes large.

To cope with the complexity caused by the large number of users, Alam et al.[29] introduced multi-agent reinforcement learning (MARL) for computation offloading. Each user can do distributed learning while having a global reward. The paper aims to ensure low-latency service delivery towards mobile service consumers. The simulation shows that the proposed method reduces the execution time and latency of accessing mobile services while ensuring lower energy consumption of mobile devices.

In task offloading, RL and DRL are mainly used to model the change of channel condition. The feedback channel state information (CSI) is often hysteretic, thus we can not know the real time channel condition. Some works proposed that the channel condition evolves according to MDP. But the transition probabilities of the MDP are generally unknown. RL and DRL are exactly the matching solutions for the case. With the help of RL and DRL, system can make better decisions for the future.

IV. Application of AI in Software Defined Network

The centralized control of software defined network (SDN) gives the ability to control all the routing and scheduling over the network. How to effectively manage the routing to avoid congestion and guarantee the QoS of users, therefore becomes a hot topic in SDN research. The dynamics of the network environment is a big challenge in routing and scheduling. The unknown network traffic in the future will disturb the current routing or scheduling. Hence, RL has natural potential to solve the problem.

Wang et al.[30] considered software defined cognitive routing for Internet of vehicles (IoV). The paper studied the routing in a proposed architecture which includes three logical layers. In the control plane, the logical SDN controller is in charge of sensing the state of the whole network and controlling the entire IoV. In the data plane, SDN wireless access infrastructures including access points (APs), RSUs, and BSs act as the switching network of SDN. In the bottom layer, vehicles are considered as the wireless nodes which send and receive messages from the SDN wireless access infrastructures. In order to distinguish different features of various vehicles and locations, the paper grouped different average vehicle speeds and vehicle densities into different discrete states. Q-learning is used to find the optimal routing in terms of packet delivery ratio and average end-to-end delay. The states include the vehicle speed and vehicle density. The action is

the choice of the giving routing protocols. The reward function is given by the weighted sum of packet delivery ratio and average end to end delay. However, the proposed method is coarse-grained. Not only the average vehicle speeds and vehicle densities are discretized into several rough ranges, but also the routing policies are predefined and are not flexibly adjusted.

Lin et al.[31] proposed a more fine-grained RL method to control the routing in multi-layer hierarchical SDN, in which the controller can decide each single hop in routing. Specifically, when a new flow arrives at a switch, the switch forwards the first packet of the flow to the controller. The controller will gather the latest network state and calculate a feasible path. The objective is to maximize the QoS of users, including link delay, queueing delay, packet loss and available bandwidth. These factors were quantified to the same scale, so that they can directly be summed together. Also, these factors were weighted based on their priorities according to the specific application scenario. The paper considered the network information as the state, next-hop selection as the action and QoS as the reward. It then formulated the routing path selection as an MDP problem and used Q-learning to solve it. Simulation results demonstrate that the proposed Q-learning method outperforms other baseline solutions and provides fast convergence with QoS provisioning.

The aforementioned works control the routing based on Q-learning. It is noted that the fine-grained control on data flows would require huge storage space to maintain Q-tables as well as a huge amount of time to look up Q-tables. Thus, they are limited to extend to a large-scale network. To deal with the problem, DRL based routing control algorithm was proposed[32]. Deep deterministic policy gradient (DDPG) was adopted to realize the routing control of the network in continuous time. For DDPG, the state is the traffic matrix of the current network load. The action is to select the path of the data flow by changing the weights of the links. The reward is not specified in the paper. It can be a single performance parameter, such as delay, packet loss, or a comprehensive strategy with priority weighted sum. The evaluation shows that the method has lower delay and higher throughput compared with the existing solutions. However, the DRL based methods have high computation complexity, so thy are hardly applied to the real-time systems.

Recently, Chen et al.[33] developed a two-level DRL system named AuTO, which can handle flow-level traffic optimization in real time. The paper first gave some experimental results to show that the processing delay of current DRL systems is too long to handle the flow-level traffic optimization at the scale of current data centers. Inspired by the Peripheral and Central Nervous Systems in animals, the paper designed AuTO as a two-level system shown in Fig. 3[33]. Peripheral Systems (PSs) run on all application servers which generate

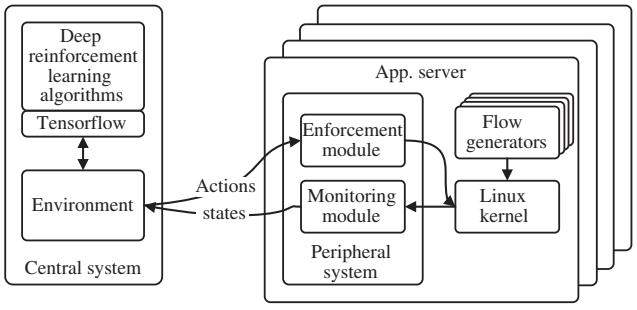


Figure 3 AuTO overview

the flows. They also collect flow information and make traffic optimizations decisions locally for short flows to minimize the delay. On the other hand, long flows can tolerate longer processing delays. Therefore, they are allocated to the Central System (CS) to make decisions. Multi-Level Feedback Queueing (MLFQ)[34] was adopted at PS in order to schedule flows without centralized control. There are several queues in MLFQ with different priorities and flows are dispatched to the queues according to their length. The calculation of the optimal thresholds which partition the priorities is the main challenge of MLFQ. The paper optimizes the thresholds using the DRL framework at CS. Considering the continuous action space for threshold, DDPG is applied as the learning model. The states are defined as the set of all active flows and the set of all finished flows. The action is a set of MLFQ threshold values. The reward is modeled as the ratio between the average throughputs of two consecutive time steps. The trained thresholds are then sent back to PSs so that PSs can handle the short flows locally without going through the DRL. Since no DRL is needed for making decisions for short flows, the processing delay is significantly decreased. Compared to other approaches, AuTO achieves superior traffic optimization performance while reducing the turn-around time.

The uncertainty in SDN is mainly caused by the unpredictable network traffic. The routing and scheduling management based on the current network state could be suboptimal in the future since there can be new traffic flows generated. RL and DRL can be used to decide each hop of the routing base on the real-time changing network state. Also, for traffic optimization which schedules each packet in switch, there is no clue for the optimal solution and only heuristics algorithms are used for the current approach. Therefore, RL and DRL can also be helpful to determine the parameters when designing the heuristics algorithms.

V. Application of AI in Network Virtualization and Network Slicing

The most important problem in network virtualization and network slicing management is the resource allocation for each slice. The advantage of network slicing is the flexibility

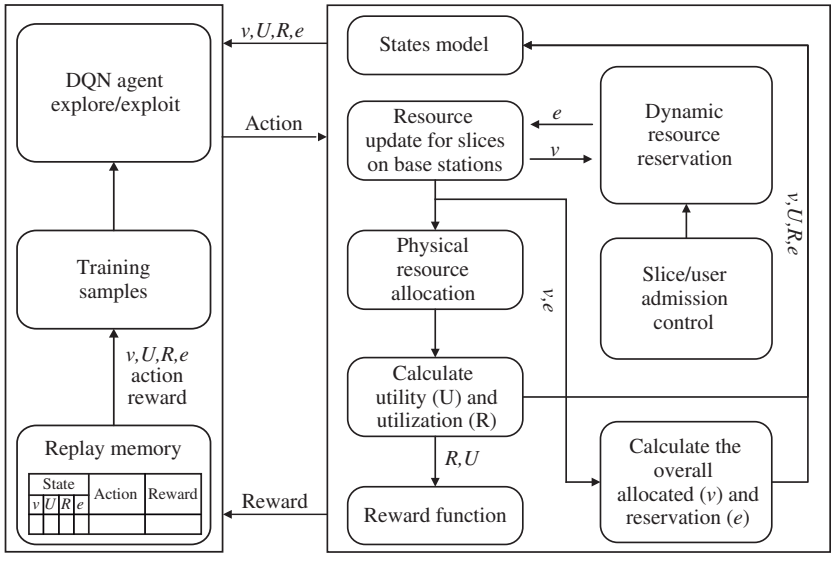


Figure 4 The DRL based framework for radio resource reservation and allocation

to adjust the resource allocated to each slice based on users' demand. Hence, the slice with active users can obtain more resource and the slice with inactive users can release the resource it takes. For effective network slicing, dynamic and adaptive resource allocation management is required. Without knowing users' future demands, existing models can hardly get an appropriate resource allocation strategy for the future. On the other hand, RL and DRL are the promising solutions to predict the trend for the adaptive resource management. Based on the location of network slicing, it can be categorized into two parts, i.e., network slicing for radio access network (RAN) and for core network (CN). Below, we will discuss how RL and DRL are applied in the two scenarios.

Raza et al.[35] proposed an RL based framework for network slicing management in RAN. A central office (CO) and a remote regional data center (RDC) were considered in the network. Both of them can do the processing, but processing at the RDC will have longer latency. CO and RDC are connected via the optical backhaul (OBH). The paper considered two kinds of services, i.e., high priority (HP) services with strict latency constraints and low priority (LP) services with non-strict latency constraints. HP services can only be placed at the CO and LP services can be assigned either at the CO or at the RDC sites. In each time step, the remaining resource and resource requirements of the new slice request are adopted as the input states. The system needs to decide whether or not a new slice request should be accepted, which is the output action. The reward is defined as the sum of the revenue generated by accepting a new slice and the penalty derived by not being able to scale up when needed. Results show that when tenants request slices with different latency requirements, the proposed policy outperforms benchmark heuristics by up to 54.5%[35]. In this paper, RL is only used to determine whether the new slice should be accepted or not. The detailed resource allocation management is done by heuristics algorithms, which may lead to inappropriate allocation and suboptimal results.

Sun et al.[36] proposed a dynamic reservation and deep reinforcement learning based autonomous virtual resource management. The paper focused on the radio resource allocation mechanism for different applications based on their requirements. The infrastructure provider first periodically reserves the unused resource to the slices based on their ratio of minimum resource requirements. Then, the slices autonomously control their resource allocation by using DRL based on the average QoS utility and resource utilization of their users. Fig. 4 shows the framework of the proposed DRL[36]. The input states are the allocated resource v, QoS utility U, resource utilization R and reserved resource e. The action is a set of discrete percentages indicating the increment or decrement of the resource allocated to the slice. The reward is defined as the weighted sum of average QoS utility and average resource utilization of the slice. The simulation shows that the proposed framework outperforms other references on both slice satisfaction and resource utilization. The DRL based framework proposed in this paper can be seen as a guideline to design radio resource reservation and allocation management.

The two works mentioned above studied the resource allocation management for RAN. Li et al.[13] considered resource allocation for both RAN and CN with different optimization objectives. The radio resource management aims to maximize the resource utilization, in the meantime, the objective of core network virtualized network functions (VNFs) resource allocation is to minimize the scheduling delay, i.e., maximize the users' QoE. The paper then formulated the joint resource management and scheduling problem for RAN and CN. Due to the

Table 1 The mapping from resource management to DRL

	Radio resource slicing	Priority-based core network slicing
State	The number of arrived packets in each slice with in a specific time window	The priority and timestamp of last arrived five flows in each service function chain (SFC)
Action	Allocated bandwidth to each slice	Allocated SFC for the flow at the current timestamp
Reward	Weighted sum of SE and QoE in 3 sliced band	Weighted sum of average time in 3 SFCs

lack of priori knowledge of volatile demand variations, DRL is introduced to solve the problem. The input states, output actions and feedback rewards are listed in Tab. 1[13]. The simulation demonstrates the advantage of the proposed DRL based management over several competing schemes. However, the resource utilization of RAN and scheduling delay of CN are on two different scales. Thus, they can not be compared with each other. Using weighted sum of these two parameters as reward can be meaningless. On the other hand, the management of RAN and CN are independent and can be separately optimized. Therefore, we individually discuss the resource management of CN below.

One key aspect in CN network virtualization is the allocation of physical resources to virtual networks (VNs). This involves embedding VNs onto substrate network (SNs), and management of the allocated resources throughout the lifecycle of the virtual network. Mijumbi et al.[37] supposed that the virtual network embedding (VNE) were done by the existing works[38-42] and mainly focused on the resource allocation management problem. Therefore, the remaining issue is how to allocate/reserve the resources for the embedded VN to ensure optimal utilization of all SN resources. The paper considered each node and link in the substrate network as a node agent and a link agent, respectively. The node agents manage node queue sizes and the link agents manage link bandwidths. An MARL based algorithm was then proposed for the resource allocation management. The state of each agent is represented by the percentage of resource allocation, the percentage of unused virtual resources, and the percentage of unused substrate resources. The action is the increment or the decrement of the allocated resources percentage. The reward is determined by the link delays, packet drops and network resource utilization. The simulation shows that the proposed approach improves virtual network acceptance ratio and the maximum number of accepted virtual network requests. Furthermore, it can also guarantee QoS requirements such as packet drop rate and virtual link delay. Nevertheless, considering the fact that resource allocation management is related to the VNE optimization, separately optimizing the resource allocation may not achieve the global optimal solution as joint optimization.

Lu et al.[43] further took the VNE step into consideration and studied both VNE and resource allocation in inter-datacenter optical network (IDCON). Previous works on VNE all assumed that the infrastructure provider (InP) is in charge of VNE. The InP can hardly realize the most effective resource, if it cannot directly forecast future VN requests from the tenants. In this paper, the calculation of VNE schemes is moved to the tenants. The InP first performs resource advertising and pricing to tell the tenants about the nodes and links that can be used to embed their VNs and the cost of using the corresponding infrastructure and bandwidth resources. Based on the advertisement from the InP, each tenant needs to determine whether the price is affordable. If yes, the tenant will distributedly calculate the VNE scheme for its VN with the lowest cost and submit the scheme to the InP. The InP first collects all the requests from the tenants, grants them based on current network status and calculates the profit from the VNT slicing. Then it feeds all the information into a DRL module to obtain the strategy of the next round resource advertising and pricing for maximizing its profit. Compared with traditional centralized VN slicing framework, the proposed approach can achieve a higher profit and a lower computation complexity.

Without knowing the requirement of users' future demand, the resource allocation in network virtualization will fall into a dilemma. Allocating too many resources to the current users and services will affect the QoS of potential high-priority services in the future. While reserving too many resources will lead to low resource utilization thus decreases the QoS of the active users. Therefore, RL and DRL can help to learn the pattern of user's demand and predict the future requests. By using RL and DRL, we are hoping to find a trade-off of resource allocation in network virtualization.

VI. Conclusion and Future Work

In 5G and beyond 5G era, various applications with different requirements are emerging. It also proposes higher requirements for the flexibility and adaptability of the network. Human-intervention control not only takes much time and cost, but also can be inappropriate and suboptimal. In some scenarios, the uncertainties in the network environments make it hard for traditional model based approach to achieve optimal performance. The uncertainties include unknown user requests in the future, unknown channel conditions, unknown network traffics and so on. In these cases, RL and DRL can be involved to make decisions.

The application scenarios of RL and DRL can be different. RL based approach is applied when the dimension of system states and actions is relatively low. The advantage of RL is that it is easy to apply and analyze. The algorithm is based on Bellman equation which has complete derivation and deep theoretical basis of the convergence. Therefore, it usually can obtain a good performance and the computational complexity of RL is low. Moreover, it can make decisions in a short time and can be easily applied to real systems. However, when the dimension of states and actions grows up, RL will have huge complexity and thus can hardly be applied.

On the other hand, DRL is often applied to deal with the high dimension inputs. Due to the continuous input space of DNN, the dimension can decrease significantly. It can handle many complex environment or large scale networks. Nevertheless, the computational complexity of DRL is high. Even with GPU, it is still challenging to apply DRL to the real time systems. Besides, the convergence of DRL based approach is not guaranteed. It may take a long time to adjust the parameters. Recently, many works have been done to enhance the convergence of DRL and achieve good progress[12,15].

This paper focuses on the application of RL and DRL in network caching, task offloading, routing scheduling and resource allocation. Undoubtedly, RL and DRL can be applied to many other fields in networks, such as load balancing, interference management, user pairing, etc. With the fast development of communication network and higher requirement of users, RL and DRL have a broad prospect in managing the intelligent network.

REFERENCES

[1] M. Satyanarayanan, P. Bahl, R. Caceres, et al. The case for VM-based cloudlets in mobile computing [J]. IEEE Pervasive Computing, 2009, 8(4): 14-23.

[2] W. Shi, J. Cao, Q. Zhang, et al. Edge computing: Vision and challenges [J]. IEEE Internet of Things Journal, 2016, 3(5): 637-646.

[3] S. Shenker, M. Casado, T. Koponen, et al. The future of networking, and the past of protocols [J]. Open Networking Summit, 2011, 20: 1-30.

[4] N. Mckeown, T. Anderson, H. Balakrishnan, et al. OpenFlow: Enabling innovation in campus networks [J]. ACM SIGCOMM Computer Communication Review, 2008, 38(2): 69-74.

[5] J. S. Turner, D. E. Taylor. Diversifying the Internet [C]//IEEE Global Telecommunications Conference, St. Louis, 2005, 2: 6-760.

[6] N. Feamster, L. Gao, J. Rexford. How to lease the Internet in your spare time [J]. ACM SIGCOMM Computer Communication Review, 2007, 37(1): 61-64.

[7] G. Tesauro. Temporal difference learning and TD-Gammon [J]. Communications of the ACM, 1995, 38(3): 58-68.

[8] S. J. Russell, P. Norvig. Artificial intelligence: A modern approach [J]. Applied Mechanics & Materials, 2009, 263(5): 2829-2833.

[9] R. S. Sutton, A. G. Barto. Reinforcement learning: An introduction [M]. Cambridge, MIT Press, 2018, 1(1): 131-133.

[10] R. S. Sutton. Learning to predict by the methods of temporal differences [J]. Machine Learning, 1988, 3(1): 9-44.

[11] K. Arulkumaran, M. P. Deisenroth, M. Brundage, et al. Deep reinforcement learning: A brief survey [J]. IEEE Signal Processing Magazine, 2017, 34(6): 26-38.

[12] V. Mnih, K. Kavukcuoglu, D. Silver, et al. Human-level control through deep reinforcement learning [J]. Nature, 2015, 518(7540): 529-533.

[13] R. Li, Z. Zhao, Q. Sun, et al. Deep reinforcement learning for resource management in network slicing [J]. IEEE Access, 2018, 6: 74429-74441.

[14] T. P. Lillicrap, J. J. Hunt, A. Pritzel, et al. Continuous control with deep reinforcement learning [J]. Computer Science, 2015, 8(6): A187.

[15] V. Mnih, A. P. Badia, M. Mirza, et al. Asynchronous methods for deep reinforcement learning [C]//International conference on machine learning, New York City, 2016: 1928-1937.

[16] J. Wang. A survey of web caching schemes for the Internet [J]. ACM SIGCOMM Computer Communication Review, 1999, 29(5): 36-46.

[17] K. Poularakis, G. Iosifidis, V. Sourlas, et al. Multicast-aware caching for small cell networks [C]//IEEE Wireless Communications and Networking Conference (WCNC), Istanbul, Turkey, 2014: 2300-2305.

[18] E. Bastu, M. Bennis, M. Kountouris, et al. Cache-enabled small cell networks: Modeling and tradeoffs [J]. EURASIP Journal on Wireless Communications and Networking, 2015, 2015(1): 41.

[19] N. Golrezaei, P. Mansourifard, A. F. Molisch, et al. Base-station assisted device-to-device communications for high-throughput wireless video networks [J]. IEEE Transactions on Wireless Communications, 2014, 13(7): 3665-3676.

[20] N. Golrezaei, A. G. Dimakis, A. F. Molisch. Scaling behavior for device-to-device communications with distributed caching [J]. IEEE Transactions on Information Theory, 2014, 60(7): 4286-4298.

[21] P. Blasco, D. Gündüz. Learning-based optimization of cache content in a small cell base station [C]//2014 IEEE International Conference on Communications (ICC), Sydney, 2014: 1897-1903.

[22] K. Psounis, A. Zhu, B. Prabhakar, et al. Modeling correlations in web traces and implications for designing replacement policies [J]. Computer Networks, 2004, 45(4): 379-398.

[23] L. Huang. System intelligence: Model, bounds and algorithms [C]// Proceedings of the 17th ACM International Symposium on Mobile Ad Hoc Networking and Computing, Paderborn, 2016: 171-180.

[24] Y. Wei, Z. Zhang, F. R. Yu, et al. Joint user scheduling and content caching strategy for mobile edge networks using deep reinforcement learning [C]//IEEE International Conference on Communications Workshops (ICC Workshops), Kansas City, 2018: 1-6.

[25] Y. Sun, Y. Cui, H. Liu. Joint pushing and caching for bandwidth utilization maximization in wireless networks [J]. IEEE Transactions on Communications, 2017, 67(1): 391-404.

[26] L. Xiao, C. Xie, T. Chen, et al. A mobile offloading game against smart attacks [J]. IEEE Access, 2016, 4: 2281-2291.

[27] Z. Chen, X. Wang. Decentralized computation offloading for multi-user mobile edge computing: A deep reinforcement learning approach [EB]. arXiv: 1812.07394.

[28] L. T. Tan, R. Q. Hu. Mobility-aware edge caching and computing in vehicle networks: A deep reinforcement learning [J]. IEEE Transactions on Vehicular Technology, 2018, 67(11): 10190-10203.

[29] M. G. R. Alam, Y. K. Tun, C. S. Hong. Multi-agent and reinforcement learning based code offloading in mobile fog [C]//2016 International Conference on Information Networking (ICOIN), Kota Kinabalu, 2016: 285-290.

[30] C. Wang, L. Zhang, Z. Li, et al. SDCoR: Software defined cognitive routing for Internet of vehicles [J]. IEEE Internet of Things Journal, 2018, 5(5): 3513-3520.

[31] S. C. Lin, I. F. Akyildiz, P. Wang, et al. QoS-aware adaptive routing in multi-layer hierarchical software defined networks: A reinforcement

learning approach[C]//13th IEEE International Conference on Services Computing, San Francisco, 2016: 25-33.

[32] C. Yu, J. Lan, Z. Guo, et al. Drom: Optimizing the routing in software-defined networks with deep reinforcement learning [J]. IEEE Access, 2018, 6: 64533-64539.

[33] L. Chen, J. Lingys, K. Chen, et al. Auto: Scaling deep reinforcement learning for datacenter-scale automatic traffic optimization [C]//Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication, Budapest, 2018: 191-205.

[34] W. Bai, L. Chen, K. Chen, et al. Information-agnostic flow scheduling for commodity data centers[C]//Usenix Conference on Networked Systems Design & Implementation, Santa Clara, 2015: 455-468.

[35] M. R. Raza, C. Natalino, P. Ohlen, et al. A slice admission policy based on reinforcement learning for a 5G flexible RAN [C]//European Conference on Optical Communication (ECOC), Rome, 2018: 1-3.

[36] G. Sun, G. T. Zemuy, K. Xiong. Dynamic reservation and deep reinforcement learning based autonomous resource management for wireless virtual networks [C]//IEEE 37th International Performance Computing and Communications Conference (IPCCC), Orlando, 2018: 1-4.

[37] R. Mijumbi, J. Gorricho, J. Serrat, et al. Design and evaluation of learning algorithms for dynamic resource management in virtual networks[C]//IEEE Network Operations and Management Symposium (NOMS), Krakow, 2014: 1-9.

[38] J. Lu, J. Turner. Efficient mapping of virtual networks onto a shared substrate [R]. DCSE Department, Washington University in St. Louis, 2006, 35: 1-11.

[39] M. Chowdhury, M. R. Rahman, R. Boutaba. ViNEYard: Virtual network embedding algorithms with coordinated node and link mapping [J]. IEEE/ACM Transactions on Networking, 2012, 20(1): 206-219.

[40] I. Houidi, W. Louati, D. Zeghlache. A distributed virtual network mapping algorithm [C]//IEEE International Conference on Communications, Reno, 2008: 5634-5640.

[41] J. Infhr, G. R. Raidl. Introducing the virtual network mapping problem with delay, routing and location constraints [J]. Lecture Notes in Computer Science, 2011, 6701(6): 105-117.

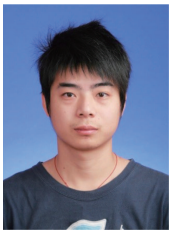
[42] A. Jarray, A. Karmouch. VCG auction-based approach for efficient virtual network embedding [C]//IFIP/IEEE International Symposium on Integrated Network Management (IM 2013), Ghent, 2013: 609-615.

[43] W. Lu, H. Fang, Z. Zhu. AI-assisted resource advertising and pricing to realize distributed tenant-driven virtual network slicing in inter-DC optical networks [C]//International Conference on Optical Network Design and Modeling (ONDM), Dublin, 2018: 130-135.

About the Authors

Yichen Qian received his B.S. degree in computer science and technology from Tongji University, Shanghai, China in 2014, and is currently working towards his Ph.D. degree in computer science and technology at Tongji University, Shanghai, China. His research interests include artificial intelligence and wireless networking.

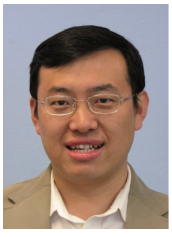
Jun Wu (M'03-SM'14) [corresponding author] received his B.S. degree in information engineering and his M.S. degree in communication and electronic system from Xidian University, Xi'an, China in 1993 and 1996, respectively, and his Ph.D. degree in signal and information processing from Beijing University of Posts and Telecommunications, Beijing, China in 1999. He joined Tongji University, Shanghai, China as a full professor in 2010. He served as a principal scientist with Huawei and Broadcom before joining Tongji University. His research interests include wireless communication, information theory and signal processing.



Rui Wang received his Ph.D. degree in 2013 from Shanghai Jiao Tong University, China. From Aug. 2012 to Feb. 2013, he was a visiting Ph.D. student at the Department of Electrical Engineering of University of California, Riverside. From Oct. 2013 to Oct. 2014, he was with the Institute of Network Coding, the Chinese University of Hong Kong as a postdoctoral research associate. From Oct. 2014 to Dec. 2016, he was with the College of Electronics and Information Engineering, Tongji University as an assistant professor, where he is currently an associate professor.

Dr. Wang received the Shanghai Excellent Doctor Degree Dissertation Award in 2015 and received the ACM Shanghai Rising Star Nomination Award in 2016. He has published over 60 papers. His research interests include wireless cooperative communications, MIMO technique, network coding, and OFDM etc. Dr. Wang is currently an associate editor of the journal of IEEE Access and editor of IEEE Wireless Communications Letters.

Fusheng Zhu graduated from Huazhong University of Science and Technology in 1996. He received his B.E. degree in Electronic and Information Engineering, and received his MBA degree from Fudan University in 2011. He was the chief engineer of ZTE Wireless, where he was responsible for research and development of ZTE since he joined the company in 1996. He was appointed as the president of Guangdong New Generation Communication and Networks Innovative Institute (GDCNi) in 2018. His research direction mainly includes 6G mobile network and B5G vertical application, network communication.



Wei Zhang (F'15) is a professor at University of New South Wales, Sydney, Australia. His current research interests include UAV communications, mmWave communications, space information networks, and massive MIMO. Currently, he serves as a TPC co-chair of IEEE/CIC ICCC 2019, Changchun, China. He is the editor-in-chief of IEEE Wireless Communications Letters and editor-in-chief of Journal of Communications and Information Networks. He also serves as the chair for IEEE Wireless Communications Technical Committee and the vice director of IEEE Communications Society Asia Pacific Board. He is a member of Fellow Evaluation Committee of IEEE Vehicular Technology Society and a member of Board of Governors of IEEE Communications Society. He is an IEEE Fellow.