IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 16, 2019, accepted January 5, 2020, date of publication January 22, 2020, date of current version February 5, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2968554

AI Inspired Intelligent Resource Management in Future Wireless Network

SIBAO FU[1], FAN YANG[2], AND YE XIAO[2]

[1]School of Economics and Management, Beijing University of Posts and Telecommunications, Beijing 100876, China
[2]Key Laboratory of Universal Wireless Communications Ministry of Education, Beijing University of Posts and Telecommunications, Beijing 100876, China

Corresponding author: Sibao Fu (fusibao@bupt.edu.cn)

This work was supported by the New Model Application Intelligent Manufacturing Projects of MIIT.

ABSTRACT In order to improve network performance, including reducing computation delay, transmission delay and bandwidth consumption, edge computing and caching technologies are introduced to the fifth-generation wireless network (5G). However, the volume of edge resources is limited, while the number and complexity of tasks in the network are increasing sharply. Therefore, how to provide the most efficient service for network users with limited resources is an urgent problem to be solved. Thus, improving the utilization rate of communication, computing and caching resources in the network is an important issue. The diversification of network resources brings difficulties to network management. The joint resource allocation problem is difficult to be solved by traditional approaches. With the development of Artificial Intelligence (AI) technology, these AI algorithms have been applied to joint resource allocation problems to solve complex decision-making problems. In this paper, we first summarize the AI-based joint resources allocation schemes. Then, an AI-assisted intelligent wireless network architecture is proposed. Finally, based on the proposed architecture, we use deep Q-network (DQN) algorithm to figure out the complex and high-dimensional joint resource allocation problem. Simulation results show that the algorithm has good convergence characteristics, proposed architecture and the joint resource allocation scheme achieve better performance compared to other resource allocation schemes.

INDEX TERMS Artificial intelligence, deep Q-network, resource management, edge computing and caching, the fifth-generation wireless network (5G).

I. INTRODUCTION

After the frozen of 5G mobile network specification Release 15 on June 2018, the first nationwide commercial 5G network launched in South Korea on April 2019 [1]. Meanwhile, the study of roadmap and enabling technologies beyond the 5G mobile network are in progress in both the academic and telecommunication industries.

The key challenges for 5G come from the increasing demands for higher data rate and spectrum utilization, lower latency, better and always-on connectivity to support communication among things and devices. Thus, software-defined networking (SDN) and network functions virtualization (NFV) are vital in the 5G and beyond 5G mobile network architecture and implementation. For example, SDN controllers are deployed to regulate the traffic and organize the virtualized network devices, which can significantly improve the efficiency and performance of the network [2], [3].

From the aspects of application scenarios, typical novel applications of Augmented Reality (AR) and Virtual Reality (VR) online game, real-time video processing and intelligent health care over the wireless network prompt the deployment of cache and fog/edge computing devices in wireless networks [4], [5]. The deployment and scheduling of storage and computing resources will be a lot more complicated than what came before. The development of AI technology brings convenience to all aspects of our life, such as smart medical care, smart factories, smart cities and so on [6]–[8].

In the practical wireless system, many traditional resource optimization methods in wireless communication networks are becoming much more performance-constrained and complicated in complex scenarios [9], [10]. Currently, AI technology which mainly includes machine learning and deep learning can extract useful information from

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

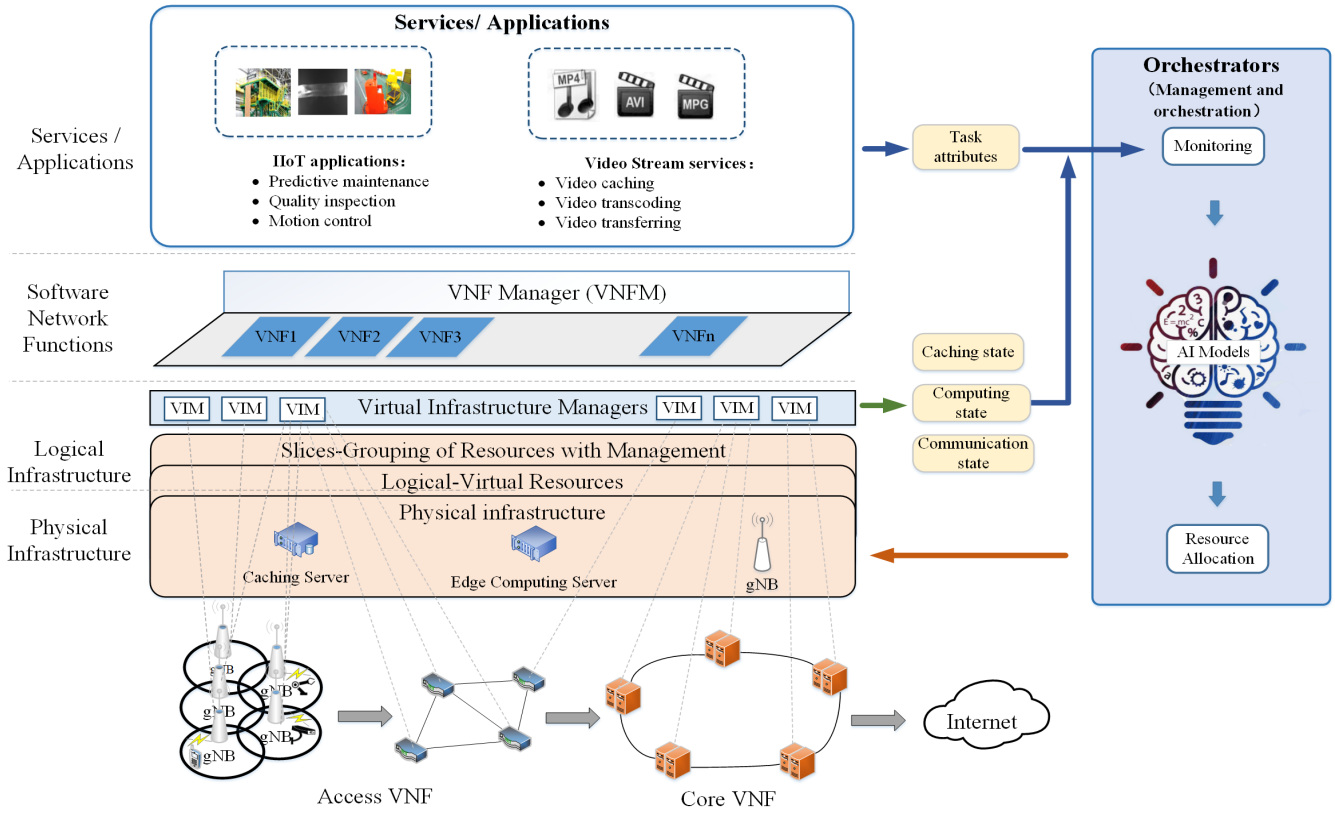

FIGURE 1. AI-assisted intelligent wireless network architecture.

wireless systems [11], learn and make decisions from the dynamic environment, are considered as potential solutions for typical complex and previously intractable problems in future wireless network [12]. In view of those observations, it is necessary to review how to apply AI technology to solve the complicated decision-making problem and boost network performance. Subsequently, an intelligent wireless network framework with self-adaptation and self-optimization capability is proposed, and an example application is addressed to illustrate the workflow. Computer evaluations are implemented to show the gain over traditional schemes. Finally, concluding remarks and implementation challenges for AI-based wireless network resource management are addressed.

II. AI ASSISTED INTELLIGENT WIRELESS NETWORK ARCHITECTURE

This section introduces a 5G-based AI-assisted intelligent wireless network architecture. The communication, computing and caching resources are virtualized as resource pools, then orchestrator jointly manages and orchestrates the three resources through AI algorithms dynamically according to the changing environment. Meanwhile, this section also introduces various AI-based resource allocation schemes.

A. ARCHITECTURE AND APPLICATION

The system architecture could be divided into four parts, which include physical infrastructure, logical infrastructure, software network functions and orchestrators. The system architecture is shown in Figure 1.

The physical infrastructure layer consists of caching servers, edge computing servers and gNodeB (gNBs), which are responsible for caching content, executing computing tasks, and provide network access for users, respectively.

The physical infrastructures are abstract into logical-virtual resources in the logical infrastructure layer. Moreover, the logical resources are sliced according to the requirements. Virtual Infrastructure Managers (VIMs) dynamically monitor and manage the infrastructures.

In the software network functions layer, Virtual Network Functions (VNFs) are softwares that provide some kinds of network services. And VNF Manager (VNFM) is responsible for VNF lifecycle management.

Orchestrator is responsible for orchestrating infrastructures based on the tasks attribute and resource status. The orchestrator, with the AI algorithm built in, analysis the system resources status and task attributes to dynamically allocate corresponding computing, caching and communication resources for specific tasks.

The workflow of the architecture is shown in Figure 2. When a gNB receives a service request from a user, it determines whether the task is cache-related or not, for example, video stream service task is a cache-related task, and quality inspection task in the industrial network is not a cache-related task.

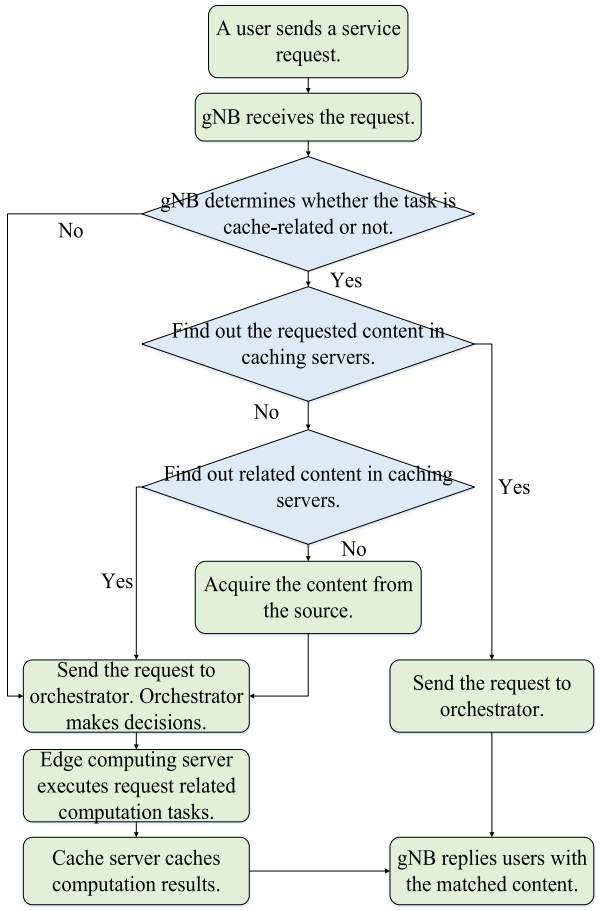

FIGURE 2. Workflow of AI-assisted intelligent wireless network architecture.

Video streaming task refers to caching video in caching server near users, to meet large number of requests for videos from users [13]. In general, videos with high bit rates are cached in the system. Even if the user requests video with a low bit rate, edge computing server can meet the requirement through transcoding. However, if videos with low bit rate are requested with high frequency, the caching server can also cache video with low bit rate to reduce the amount of edge computation. This kind of task usually needs the coordination of communication, caching and computing resources.

The quality inspection task is usually to take pictures of the products and identify the collected pictures, and classify the products through the identification results. This kind of task usually needs the coordination of communication and computing resources.

If a task is not a cache-related task such as a quality inspection task, gNB sends the request directly to the orchestrator. The orchestrator runs AI algorithms including Reinforcement Learning (RL) algorithm, DQN algorithm, and so on. With the help of AI algorithms, orchestrator dynamically allocates computing and communication resources to the user according to the task attributes, edge computing and communication resource status. The user sends the task-related data, such as the product images to the chosen gNB, and the gNB sends the data to the chosen edge computing server. After receiving the data, the edge computing node performs the computing task, such as image identification tasks to find out the defects on the products. Then, edge computing server returns the computing results to the user via the chosen gNB.

If the task is a cache-related task, such as a video stream service, the system will find out if the requested video version is cached in the system or not. Supposing that the required video version exists, the video will be sent to the user via a chosen gNB. If the requested video version is not cached in the system, the system checks if the system cached video with a higher bit rate version. While there is no higher bit rate version, the system will acquire the content from the source. After the acquiring original version or higher bit-rate version, the task is sent to the orchestrator, and the orchestrator chooses an edge server to transcode the video. Moreover, orchestrator also decides whether to cache the video transcoded by the edge computing server. If the orchestrator decides to cache the video, then a caching server will be allocated to cache the video. Finally, the video will be sent back to the user via a chosen gNB.

In view of this, a resource allocation scheme running in orchestrator is quite important. Reasonable resource allocation can bring higher system benefits. However, in the network, the problem of joint resource allocation is a high dimensional and complex problem and is difficult to be solved by conventional algorithms. Therefore, many resource allocation schemes based on AI algorithms have been proposed.

B. AI FOR RESOURCE ALLOCATION PROBLEM

In this section, recent technical progress in the AI-based resource allocation schemes is briefly discussed. AI approaches can not only be applied in tradition areas, but also can be used in wireless communications fields.

- Supervised Learning & Unsupervised Learning:
  The high-quality training dataset in wireless network environment is difficult to obtain, which constraints the application of supervised learning [9], [14], and unsupervised learning [12], [15] in the wireless network.
- Deep Learning (DL):
  DL is usually used for classification or regression [11], [18]. However, the performance of deep learning is highly related to model training tricks and experiences.
- Reinforcement Learning:
  RL algorithms is suitable for making decisions on resource allocation [16], [17]. However, some tabular based RL algorithms such as Q-learning, stores experiences in a Q-table. However, the size of Q-table grows exponentially with the levels of state space, and the convergence rate to the optimal policy is relatively low.
- Deep Reinforcement Learning (DRL):
  DRL combines the advantages of deep learning and reinforcement learning [19]. In typical DRL scheme,

TABLE 1. Summarized of AI-based approaches used in resource management.

AI categories	Characterizes	Advantages	Limitations	Application in resource management
Supervised learning(NN, SVM) [9] [14]	extract features from labeled data	simple and easy to deploy	sensitive to the quality of data	classification; prediction of performances
Unsupervised learning(K-Means, PCA) [12] [15]	learned from unlabeled data	simple and easy to deploy	sensitive to the quality of data	clustering; reducing dimensions
Reinforcement learning(Q-Learning, MDP) [16] [17]	learned policy from own experiences	no need of priori knowledge of data	complexity increasing with the dimension of state and action space; low convergence rate	automatic control and decision making
Deep learning(DNN) [11] [18]	learning from raw data	better learning performance	ininterpretable; long training time	prediction
Deep reinforcement learning(DQN) [19] [20] [21]	learning policy from experiences	better performance and quick convergence	continuous state and action space; ininterpretable	automatic control and decision making; resource allocation policy

deep neural network (DNN) is used to estimate the action-value function by learning high-dimensional raw data. Compared to conventional reinforcement learning, it could be applied to high-dimensional state spaces problem. Thus, there are many researches focusing on this topic [20], [21].

Table 1 summarizes the AI-based approaches used in resource management. Generally, supervised learning and unsupervised learning are suitable for executing regression, classification and clustering tasks for assisting resource management. While, reinforcement learning algorithms are suitable for making resource allocation decisions in low-scale wireless networks with low-complexity. Recently, DQN has been deployed in solving joint resource allocation problems in various scenarios, such as vehicle network, satellite-to-ground integrated network.

III. SYSTEM MODELS

In this section, we propose three models under the AI-assisted intelligent wireless network architecture, including a communication model, a caching model and a computing model respectively. Let $\boldsymbol{U} = \{1, \ldots, U\}, \boldsymbol{C} = \{1, \ldots, C\}, \boldsymbol{C_S} = \{C_1, \ldots, C_S\}$ denote sets of users, gNBs, and computing servers.

A. COMMUNICATION MODEL

In this part, we take the realistic wireless channel between users and gNB into consideration and we model this channel as the finite-state Markov channel (FSMC). SNR is usually used to measure channel quality. Here, we denote h_u^c as the receive SNR of user u from a gNB c while we regard h_u^c as a continuous random variable.

We divide the SNR data into H' levels, each level has the same interval distance and corresponds to one Markov chain's state. Therefore, we form a H'-elements state space. The channel state realization of received SNR at time slot t is denoted as $h_u^c(t)$. With a certain transition probability, the state varies from one to another at time interval t, and the transition probability of state changes could be defined as a state transition probability matrix

$$\Gamma_u^c(t) = [\vartheta_{f_s f_{s'}}(t)]_{H' \times H'} \tag{1}$$

where $\vartheta_{f_s f_{s'}}(t)$ denotes the transition probability of $h_u^c(t)$ from one state f_s to another state $f_{s'}$ at time instants t. Besides, the communication rate of a user u can be expressed

$$R_{u,c}^{comm}(t) = a_{s,c}^{comm}(t) B_u^c(t) e_u^c(t), \quad \forall u \in \boldsymbol{U}, \tag{2}$$

$$s.t. \sum_{u \in \boldsymbol{U}} R_{u,c}^{comm}(t) \leq A^c, \quad \forall c \in \boldsymbol{C}, \tag{3}$$

if $a_{s,c}^{comm}(t) = 1$, it shows that user u utilizes networking resources of gNB c. we use B_u^c to denote the available spectrum bandwidth of the gNB c which is allocated to a user u. What's more, the user's spectrum efficiency at time slot t is $e_u^c(t)$ and the backhaul capacity of gNB c is set to A^c bps.

B. CACHING MODEL

Assuming that there exist I video contents in the network system and users will request to obtain certain video content. Note that different versions of certain video are considered as different contents. We denote the set of the video content as $C_I = \{1, 2, \ldots, I\}$ and we rank the set in descending order of popularity. In our model, we consider the arrival of contents request as the Poisson process, and we describe the probability of video contents request with Zipf-like distribution. Since the capacity of cache video content is limited, the caching server only stores some of the I video contents. In this case, we are not able to exactly know about whether the content we request is cached in the network system.

Here, we suppose user u requesting for a video content v_i and caching state $\gamma_u^{v_i}$ indicates that whether or not the content is cached, $i \in C_I$. Therefore, we can model a two-state Markov chain, i.e., state 0 and state 1. In view of a certain transition probability, we denote the transition probability matrix

$$\Psi_u^{v_i}(t) = \left[\varpi_{l_s l_{s'}}(t)\right]_{2 \times 2} \tag{4}$$

where $\varpi_{l_s l_{s'}}(t)$ is denoted as the transition probability of the state of $\gamma_u^{v_i}(t)$ varying from state l_s to state $l_{s'}$ at time slot t.

C. COMPUTING MODEL

If the version of contents requesting by users matches with the version of users' requirement, gNB can return users with the matched video content according to the decision of orchestrator; otherwise a chosen edge computing server should build a computation task to do transcoding. The computation task $T_u^{v_i}$ is considered to have two attributes, the size of the video content s_u and the number of CPU cycles n_u required for finishing this computational task.

We consider the computation capability of computing server c_s assigned to user u as random variables, which can be denoted as $x_u^{c_s}$. We can partition $x_u^{c_s}$ into P discrete levels, i.e. $\boldsymbol{Q} = \{O_0, O_1, \ldots, O_{P-1}\}$. Furthermore, the transition of the computation capability level can be modelled as a Markov process.

The variable $x_u^{c_s}$ can be denoted as $x_u^{c_s}(t)$ at time instant t, which will change from one state y_s to another state $y_{s'}$ at time instant t, we denote this process as $\zeta_{y_s y_{s'}}(t)$. Here, the transition probability matrix can be expressed

$$\Phi_u^{c_s}(t) = \left[\zeta_{y_s y_{s'}}(t)\right]_{P\times P} \tag{5}$$

We also calculate the time consumed during the process of the computation task $T_u^{v_i}$:

$$t_c^u = \frac{n_u}{x_u^{c_s}(t)} \tag{6}$$

In this case, the computation rate can be given:

$$R_{u,c_s}^{comp}(t) = a_{s,c}^{comp}(t)\frac{s_u}{t_c^u} = a_u^{c_s}(t)\frac{s_u x_u^{c_s}(t)}{n_u}, \tag{7}$$

$$s.t. \sum_{u\in U} a_{s,c}^{comp}(t) s_u \le S_m, \tag{8}$$

If user u decides to adopt computational device in server c, i.e., $a_{s,c}^{comp}(t) = 1$, the rate $R_{u,c_s}^{comp}(t)$ is represented as the ratio of the required number of CPU cycles s_u and the time consumed to finish the computation task $T_u^{v_i}$; otherwise, $a_{s,c}^{comp}(t) = 0$, $R_{u,c_s}^{comp}(t) = 0$. S_m shows the maximum size of the required content which can be loaded on a computer server simultaneously.

IV. DQN BASED JOINT RESOURCE ALLOCATION PROBLEM FORMULATION

In this section, state space, action space and reward function are successively given below in detail for the purpose of masting the policy of optimization. Then, we introduce the deep Q-network approach to solve the problem of the optimization of resource allocation.

A. STATE SPACE

In our work, the state space s actually contains three components: the channel state $\mathrm{h}_u^c(t)$, the caching state $\gamma_u^{v_i}(t)$ and the computation capability state $\mathrm{X}_u^{c_s}(t)$. Here, a state space vector can be arranged by the three elements introduced earlier

$$s(t) = \begin{bmatrix} \mathrm{h}_u^1(t) & \mathrm{h}_u^2(t) & \cdots & \mathrm{h}_u^c(t) & \cdots & \mathrm{h}_u^C(t) \\ \gamma_u^{v_1}(t) & \gamma_u^{v_2}(t) & \cdots & \gamma_u^{v_i}(t) & \cdots & \gamma_u^{v_I}(t) \\ \mathrm{X}_u^{c_1}(t) & \mathrm{X}_u^{c_2}(t) & \cdots & \mathrm{X}_u^{c_s}(t) & \cdots & \mathrm{X}_u^{C_S}(t) \end{bmatrix}, \tag{9}$$

B. ACTION SPACE

In our system, orchestrator requires to choose the gNB that can be assigned to users. At the same time, the orchestrator also requires to decide whether or not the caching server should cache the requested video content, and whether or not the edge computing server should perform the computation task. Therefore, action space $\boldsymbol{a}$ at time slot t is given by three row vectors, $\mathbf{a}_s^{\mathrm{comm}}(t)$, $\mathbf{a}_s^{\mathrm{cache}}(t)$ and $\mathbf{a}_s^{\mathrm{com}}(t)$, respectively.

$$\mathbf{a}_s(t) = \left\{\mathbf{a}_s^{\mathrm{comm}}(t), \mathbf{a}_s^{\mathrm{cache}}(t), \mathbf{a}_s^{\mathrm{comp}}(t)\right\} \tag{10}$$

The vector $\mathbf{a}_s^{\mathrm{comm}}(t)$ consists of $a_{s,c}^{comm}(t)$ and $a_{s,c}^{comm}(t) \in \{0, 1\}$. If a gNB is not allocated to user u, $a_{s,c}^{comm}(t) = 0$, otherwise $a_{s,c}^{comm}(t) = 1$. Note that, only one gNB could be selected to provide access service to the user.

The next vector $\mathbf{a}_s^{\mathrm{cache}}(t)$ contains element $a_{s,c}^{cache}(t)$ and element $a_{s,c}^{cache}(t)$ denotes whether or not the caching server should cache the requested video content. Here, $a_{s,c}^{cache}(t) \in \{0, 1\}$. To be more specific, $a_{s,c}^{cache}(t) = 1$ indicates the requested content is cached by the caching server. Note that, only one caching server could be selected to cache the video content.

The final vector $\mathbf{a}_s^{\mathrm{comp}}(t)$ includes $a_{s,c}^{comp}(t)$ and $a_{s,c}^{comp}(t) \in \{0, 1\}$. If the orchestrator has decided that the computation task should offload to the edge computing server, $a_{s,c}^{comp}(t) = 1$; otherwise $a_{s,c}^{comp}(t) = 0$. Note that, only one edge computing server could be selected to perform the task.

C. REWARD FUNCTION

In this part, we regard the *Return on Rent (RoR)* as the system reward. If the proper gNB assigned to users u, the orchestrator should pay for it and we can denote it as η_u unit price per bps. Besides, the network should also pay for the fee of caching the requested video content and executing the computation task, which is represented as ι_u unit price per Hz and λ_u unit price per bps.

Furthermore, there are backhaul bandwidth and wireless spectrum which should be paid. This total cost includes the usage of wireless resource, caching cost and energy consumed by the computation task. All of this cost can be denoted by κ_c per Hz, ω_c per unit space and ζ_c per unit CPU cycle.

In order to optimize the system reward r, the system action space plays an important role. In our paper, we formulate a reward function $R_{\mathrm{u}}^{v_i}(t)$ for the user u at time interval t as *RoR*. This function represents the ratio of the fee of using resources to the fee of owning resources

$$\begin{aligned} R_{\mathrm{u}}^{v_i}(t) = &\sum_{u\in U} a_{s,c}^{comm}(t)(R_{u,c}^{comm}(t)\eta_u/\kappa_c\, B_u^c(t)) \\ &+ \sum_{c\in C} a_{s,c}^{cache}(t)(R_{u,c}^{cache}(t)\iota_u/\omega_c s_u\ ) \\ &+ \sum_{c_s\in C_S} a_{s,c}^{comp}(t)(R_{u,c_s}^{comp}(t)\lambda_u/\zeta_c n_u\ ) \\ &\times (utility/resource) \end{aligned} \tag{11}$$

D. DEEP Q-NETWORK

In our paper, we adopt deep Q-network algorithm to solve the joint resource allocation problem. The well-trained neural networks can guide the orchestrator resource allocation action. In this network system, orchestrator needs to allocate resources for communication, video content caching and computation task execution.

DQN is a kind of reinforcement learning, which is different from the traditional tabular RL algorithms. It is a kind of reinforcement learning algorithm based on function approximation, which mainly applies artificial neural network and statistical curve fitting to estimate the value function in the reinforcement learning problem.

Thus, before introducing DQN, this section will introduce a kind of tradition tabular RL algorithms called Q-learning first. In Q-learning, there is an interaction between environment and agent during the process. First, the environment state is sensed by agents. Then, the agent decides to choose an action based on a proper policy under the environment state. Action-state value function is an evaluation of the policy, and usually a policy with higher state-action value is a better policy. Then, we can adjust the parameters so that the action with high action-state value is more likely to be selected. The learning process of Q-learning is the process of continuous iteration of the value function.

In tradition Q-learning algorithm, action-state value, we also call it Q-value, is stored in a Q-table. However, joint resource allocation problems have high dimensional state space, and the size of the Q-table grows exponentially as the state increases. It is difficult to put all states and actions into Q-table. Therefore, we utilize the neural network to evaluate Q-values. The neural network parameters which include weights and biases are trained to minimize the loss function. The weights and biases are updated during the training process to gain well estimation results. The loss function is defined as follows:

$$L(w)=(R(t)+\gamma_q \max_{a'} Q(S(t+1), a', w')-Q(S(t), a, w))^2 \tag{12}$$

DQN needs data set to train the neural networks. The data set of DQN comes from the interaction between agent and environment. A piece of data generated from each interaction is stored and used for training the neural network. However, there are correlations between the sequence of data. In order to train the neural networks, the correlations need to be removed. Thus, the experience replay mechanism is introduced. Experience replay is a biologically inspired mechanism which can reduce the correlation of obtained sequences during the training process. Moreover, the novel DQN proposed another mechanism called fixed target deep networks which is used for reducing correlations with the target, so as to guarantee the stability of the convergence process [9]. Thus, the DQN approach is shown in Algorithm 1. In order to illustrate the process of DQN more visually, Figure 3 shows the training process of DQN.

Algorithm 1 Deep Q-Network Based Joint Resource Allocation Algorithm

Initialization:
Initialize evaluated Q-networks and target Q-networks with parameters w and w'.
for $t = 1 : T$ **do**
  Orchestrator receives the task request $T_u^{v_i}(t)$.
  Orchestrator senses the current environment state $S(t)$.
  while $S(t)! = S_{terminal}$ **do**
    Orchestrator selects action $a(t)$ according to $T_u^{v_i}(t)$ and $S(t)$ based on $\epsilon-$greedy policy.
    Orchestrator obtains reward $R(t)$ and next state $S(t+1)$.
    Orchestrator stores $(S(t), a(t), R(t), S(t+1))$ in the experience replay memory.
    Randomly sample some pieces of experiences from the experience replay memory.
    Estimate target Q-value $Q_{target}(k)$ based on target Q-networks:
    if $S(k+1) == S_{terminal}$
      $Q_{target}(k) = R(k)$,
    else
      $Q_{target}(k) = R(k) + \gamma_q \max_{a'} Q(S(k+1), a', w')$.
    Train evaluated Q-networks to minimize $L(w)$.
    Every some steps, update target Q-networks.
    $S(t) \leftarrow S(t+1)$
  end while
end for

V. RESULTS AND ANALYSIS

In the simulation, we use a GPU-based server with 8GB 1867MHz lpddr3, 2GHz internet core i5. and 256G memory. And the software environment is Python 2.7.10 with Tensorflow 1.4.0.

The DQN based joint resource allocation algorithm aims at optimizing the resource allocation decision according to the task attributes and the resources status. Generally, we need to analyze the performance of the algorithm to measure whether the algorithm is suitable to solve such problems. Therefore, we will analyze the DQN based joint resource management algorithm in two steps. Firstly, the convergence performance is analyzed. Secondly, we analyze the algorithm performance on improving the system reward.

At the beginning of the simulation, we initialize the state transition probability matrices to show the computing, caching and communication status. The state transition probability matrices denote the regularity of the environment which the agent needs to learn. Let the content size of the task is 2 Mbits, the required number of CPU cycles are 5 Mcycles. The bandwidth between user and gNB is 5 MHz. And the unit paid-fee of the user for using caching, edge computing server and wireless spectrums are 5 units/MBits, 7 units/Mcycles, and 2 units/MHz. And the charging-fee from users of using

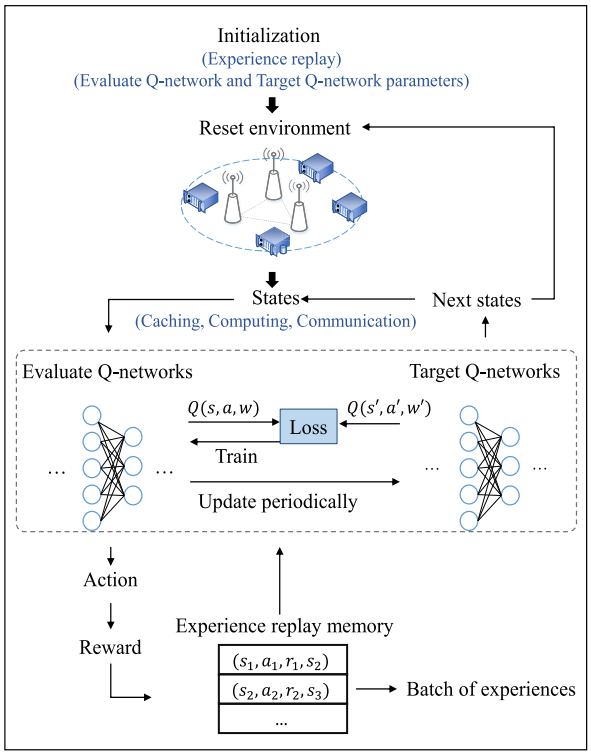

FIGURE 3. Workflow of DQN.

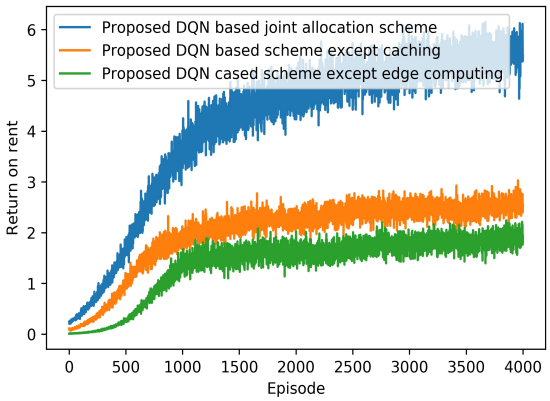

FIGURE 4. *Return on rent* versus episode under different scheme.

caching, computing and communication resources are all 0.2 unit/Mbps.

Here, we compare the performance of four simulation schemes.

- Proposed DQN based joint allocation scheme.
- Proposed DQN based scheme except edge computing.
- Proposed DQN based scheme except caching.
- Static allocation scheme.

Under the proposed DQN based joint resource allocation algorithm, the time required for DQN to train 4000 episodes is 227s, and each episode includes 50 steps. After the training process, the well-trained neural networks are used for generating the resource allocation policy.

Figure 4 shows the *return on rent* varies with the training episodes. Considering the convergence characteristic of

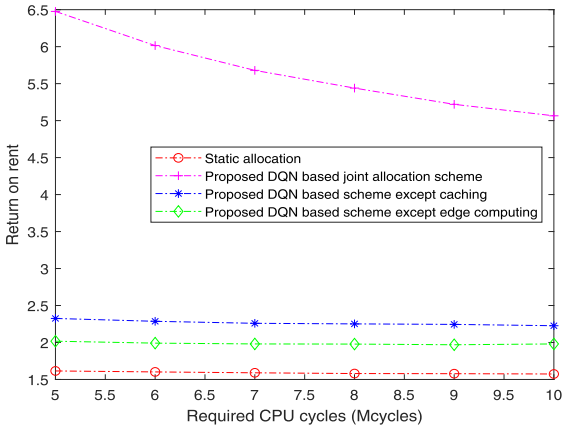

FIGURE 5. *Return on rent* versus task attribute under different scheme.

the algorithm, the neural networks are trained for 4000 episodes in this simulation. In the beginning, the *ROR* of the three schemes increase, which indicated that the neural network parameters are still updated iteratively. At the same time, it can be seen that the proposed DQN based joint allocation scheme has the highest growth rate. The reason is that the reward obtained by each iteration is relatively higher. After 1500 episodes, the curves start to be flattened, which means the DQN based algorithm begins to converge. This shows that the neural networks are trained well to guide the agent to an optimize action selection strategy. Note that, the larger *ROR* indicates a higher utilization efficiency of the resources. It is obvious, that the proposed DQN based joint allocation scheme performs best.

Figure 5 shows the relationship between the *return on rent* and the task attributes. We want to know if the algorithm performs best under different task attributes. So we dynamically change the number of required CPU cycles to see the performance changes under different schemes. With the increasing of the CPU cycles, the *ROR* under different schemes decreases. Moreover, the required CPU cycles take the computing server more time to execute the task and the computation rate decreases. Thus the system gets less fee from users and pays more for computing which cause a decrease of the total *return on rent*. This shows that in the charging approach defined in this paper, the system can obtain more benefits to handle the tasks with smaller computation requirements. But it is obvious that the proposed DQN based joint allocation scheme gains the highest *ROR*. This means it is necessary to jointly allocate the communication, computing and caching resources.

VI. CHALLENGES AND SOLUTIONS

Although DQN is an effective solution for high-dimensional, complex resource allocation problem, there still exist some challenges. In this section, we list possible challenges and suggest solutions.

A. ENVIRONMENT CONSTRAINT

In practice, due to the huge scale of the network, the training process usually consumes much time and cannot be

completed online. Therefore, we will choose offline learning, but after offline learning, online learning is used to constantly adjust and optimize the model. During this process, the AI models make choose resource allocation actions. However, when dealing with some extremely time-sensitive tasks, the imperfect neural network may choose bad actions so that the task cannot be processed. Therefore, we hope to introduce the constraints based deep reinforcement learning to limit the action selection in some conditions, so as to reduce the harm caused by some bad action selections.

B. MODEL GENERALIZATION

Since the wireless network environment is changing dynamically, it is likely that the numbers of working computing servers, caching servers, and communication access points increase or decrease. This leads to changes in the neural networks feature dimension and output which means the neural network needs to be retrained. Thus, the changing system environment is considered as a big challenge. Therefore, we wish to discuss the possibility of solving such problems by introducing transfer learning. Transfer learning can model the target domain better using knowledge from the source domain, when the data distribution, feature dimension and model output change [22]. At the same time, transfer learning can also make a huge difference when we need to apply the well-trained model to different network environments.

C. TRAINING PROBLEMS FOR LARGE-SCALE NETWORKS WITH LIMITED COMPUTING CAPABILITIES

In particular, the curse of dimensionality for learning, and the complexity of decision making are the main barriers to most of the ML-based schemes. The running time of algorithms grows exponentially with the increase of wireless network scale. The reason is that a larger scale will lead to larger state space. Therefore, we need to train the model with a large amount of data. However, this will take a lot of time and requires high computation capability for a single server. Therefore, we hope to introduce the distributed machine learning, by means of data parallelism, so that different machines have multiple copies of the same model, each machine is assigned to different data, and then the results of all the machines are merged [23].

In addition to the above problems, it is relatively difficult to perceive real-time state from reality, and the interpretability of deep learning is also a problem of DQN.

VII. CONCLUSION

In this paper, we proposed an AI-assisted intelligent wireless network architecture based on 5G, edge computing and caching technologies. Through the learning of environment status and task attributes by AI algorithms, the AI algorithms based orchestrator can reasonably allocate resources according to different situations. Among them, we choose DQN algorithm to solve the complex and high-dimensional resource allocation problem. Simulation results show that the proposed resource allocation scheme has good convergence characteristics, and the joint allocation of communication, computing and caching resources bring higher benefits. Future work should focus on improving the mobility of the system.

COMPETING INTERESTS

The authors declare that they have no competing interests.

REFERENCES

[1] A. Hoglund, D. P. Van, T. Tirronen, O. Liberg, Y. Sui, and E. A. Yavuz, "3GPP release 15 early data transmission," *IEEE Commun. Standards Mag.*, vol. 2, no. 2, pp. 90–96, Jun. 2018.
[2] I. Farris, T. Taleb, Y. Khettab, and J. Song, "A survey on emerging SDN and NFV security mechanisms for IoT systems," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 812–837, 1st Quart., 2019.
[3] Z. Zhou, J. Gong, Y. He, and Y. Zhang, "Software defined machine-to-machine communication for smart energy management," *IEEE Commun. Mag.*, vol. 55, no. 10, pp. 52–60, Oct. 2017.
[4] S. A. Shah, X. Yang, and Q. H. Abbasi, "Cognitive health care system and its application in pill-rolling assessment," *Int. J. Numer. Model., Electron. Netw., Devices Fields*, vol. 32, no. 6, p. e2632, 2019.
[5] S. Shah, A. Ren, D. Fan, Z. Zhang, N. Zhao, X. Yang, M. Luo, W. Wang, F. Hu, M. Rehman, O. Badarneh, and Q. Abbasi, "Internet of Things for sensing: A case study in the healthcare system," *Appl. Sci.*, vol. 8, no. 4, p. 508, Mar. 2018.
[6] X. Yang, "Monitoring of patients suffering from REM sleep behavior disorder," *IEEE J. Electromagn., RF Microw. Med. Biol.*, vol. 2, no. 2, pp. 138–143, Jun. 2018.
[7] X. Yang, S. A. Shah, A. Ren, D. Fan, N. Zhao, S. Zheng, W. Zhao, W. Wang, P. J. Soh, and Q. H. Abbasi, "*S* -band sensing-based motion assessment framework for cerebellar dysfunction patients," *IEEE Sensors J.*, vol. 19, no. 19, pp. 8460–8467, Oct. 2019.
[8] J. Wan, J. Yang, Z. Wang, and Q. Hua, "Artificial intelligence for cloud-assisted smart factory," *IEEE Access*, vol. 6, pp. 55419–55430, 2018.
[9] B. Cao, L. Zhang, Y. Li, D. Feng, and W. Cao, "Intelligent offloading in multi-access edge computing: A state-of-the-art review and framework," *IEEE Commun. Mag.*, vol. 57, no. 3, pp. 56–62, Mar. 2019.
[10] Z. Zhou, Y. Guo, Y. He, X. Zhao, and W. M. Bazzi, "Access control and resource allocation for M2M communications in industrial automation," *IEEE Trans. Ind. Informat.*, vol. 15, no. 5, pp. 3093–3103, May 2019.
[11] Q. Mao, F. Hu, and Q. Hao, "Deep learning for intelligent wireless networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2595–2621, 4th Quart., 2018.
[12] M. Yao, M. Sohul, V. Marojevic, and J. H. Reed, "Artificial intelligence defined 5G radio access networks," *IEEE Commun. Mag.*, vol. 57, no. 3, pp. 14–20, Mar. 2019.
[13] Y. Li, P. A. Frangoudis, Y. Hadjadj-Aoul, and P. Bertin, "A mobile edge computing-based architecture for improved adaptive HTTP video delivery," in *Proc. IEEE Conf. Standards Commun. Netw. (CSCN)*, Berlin, Germany, Oct. 2016, pp. 1–6
[14] A. Khairy, H. H. Ammar, and R. Bahgat, "Smartphone energizer: Extending smartphone's battery life with smart offloading," in *Proc. IEEE IWCMC*, Jul. 2013, pp. 36–329.
[15] M. Xiong and J. Cao, "A clustering-based context-aware mechanism for IEEE 802.21 media independent handover," in *Proc. IEEE WCNC*, Apr. 2013, pp. 1569–1574.
[16] G. Yunjie, H. Yuxiang, D. Yuehang, and X. Jichao, "Joint optimization of resource allocation and service performance in vEPC using reinforcement learning," in *Proc. IEEE 4th Int. Conf. Cloud Comput. Big Data Anal. (ICCCBDA)*, Chengdu, China, Apr. 2019, pp. 306–310.
[17] Y. Wei, F. R. Yu, M. Song, and Z. Han, "User scheduling and resource allocation in HetNets with hybrid energy supply: An actor-critic reinforcement learning approach," *IEEE Trans. Wireless Commun.*, vol. 17, no. 1, pp. 680–692, Jan. 2018.
[18] Z. Zhou, H. Liao, B. Gu, K. M. S. Huq, S. Mumtaz, and J. Rodriguez, "Robust mobile crowd sensing: When deep learning meets edge computing," *IEEE Netw.*, vol. 32, no. 4, pp. 54–60, Jul. 2018.

[19] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
[20] F. Xu, F. Yang, S. Bao, and C. Zhao, "DQN inspired joint computing and caching resource allocation approach for software defined information-centric Internet of Things network," *IEEE Access*, vol. 7, pp. 61987–61996, 2019.
[21] C. Qiu, H. Yao, F. R. Yu, F. Xu, and C. Zhao, "Deep Q-learning aided networking, caching, and computing resources allocation in software-defined satellite-terrestrial networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 5871–5883, Jun. 2019.
[22] L. Shao, F. Zhu, and X. Li, "Transfer learning for visual categorization: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 5, pp. 1019–1034, May 2015.
[23] Z. Zhang, L. Yin, Y. Peng, and D. Li, "A quick survey on large scale distributed deep learning systems," in *Proc. IEEE 24th Int. Conf. Parallel Distrib. Syst. (ICPADS)*, Singapore, Dec. 2018, pp. 1052–1056.

SIBAO FU received the B.S., M.S., and Ph.D. degrees from the Beijing University of Posts and Telecommunications (BUPT). He serves as an Associate Professor and the Finance Director with the Beijing University of Posts and Telecommunications, Beijing, China. His research interests include innovation management, performance-based budget management, health economy performance, and university financial management.

FAN YANG received the B.S. degree in e-commerce engineering with the Law from Queen Mary University of London, in 2017, and the B.Admin. degree in e-commerce engineering with law from the Beijing University of Post and Telecommunications (BUPT), China, in 2017, where she is currently pursuing the M.S. degree with the Key Laboratory of Universal Wireless Communications Ministry of Education. Her research interests include edge computing and blockchain.

YE XIAO received the B.S. degree in communication engineering from Hunan Normal University, Changsha, China, in 2018. She is currently pursuing the M.S degree with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications. Her current research interest is edge computing.