Deep Reinforcement Learning Based Computation Offloading in Fog Enabled Industrial Internet of Things

Yijing Ren, Yaohua Sun, and Mugen Peng, *Fellow, IEEE*

***Abstract*—Fog computing is seen as a key enabler to meet the stringent requirements of industrial Internet of Things (IIoT). Specifically, lower latency and IIoT devices' energy consumption can be achieved by offloading computation-intensive tasks to fog access points (F-APs). However, traditional computation offloading optimization methods often possess high complexity, making them inapplicable in practical IIoT. To overcome this issue, this article proposes a deep reinforcement learning (DRL) based approach to minimize long-term system energy consumption in a computation offloading scenario with multiple IIoT devices and multiple F-APs. The proposal features a multi-agent setting to deal with the curse of dimensionality of the action space by creating a DRL model for each IIoT device, which identifies its serving F-AP based on network and device states. After F-AP selection is finished, a low complexity greedy algorithm is executed at each F-AP under a computation capability constraint to determine which offloading requests are further forwarded to the cloud. By conducting offline training in the cloud and then making decisions online, iterative online optimization procedures are avoided and, hence, F-APs can quickly adjust F-AP selection for each device with trained DRL models. Via simulation, the impact of batch size on system performance is demonstrated and the proposed DRL-based approach shows competitive performance compared to various baselines including exhaustive search and genetic algorithm based approaches. In addition, the generalization capability of the proposal is verified as well.**

Manuscript received May 10, 2020; revised July 8, 2020; accepted August 9, 2020. Date of publication September 1, 2020; date of current version April 2, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 62001053, Grant 61925101, and Grant 61831002, in part by the State Major Science and Technology Special Project under Grant 2018ZX03001023, in part by the Beijing Natural Science Foundation under Grant JQ18016, in part by the National Program for Special Support of Eminent Professionals, and in part by the Fundamental Research Funds for the Central Universities under Grant 24820202020RC11. Paper no. TII-20-2394. *(Corresponding author: Mugen Peng.)*

Yijing Ren and Mugen Peng are with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: renyijing1996@bupt.edu.cn; pmg@bupt.edu.cn).

Yaohua Sun is with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: sunyaohua@bupt.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2020.3021024.

Digital Object Identifier 10.1109/TII.2020.3021024

***Index Terms*—Computation offloading, fog computing, industrial Internet of Things (IIoT), multi-agent deep reinforcement learning (DRL).**

I. Introduction

In recent years, Internet of Things (IoT) has developed rapidly, being applied to mobile healthcare, smart metering, smart home, etc. As a special scenario of IoT, particular attention has been paid to industrial IoT (IIoT) [1]–[4], whose typical services include industrial monitoring [5], smart manufacturing [6], virtual and augmented reality, etc. Different from traditional IoT scenarios, industrial sensors and machines continuously generate and collect large amounts of critical data. To achieve safe production and high-level industrial automation, the data should be processed and analyzed in time meanwhile keeping data privacy. To this end, fog access points (F-APs) are deployed in IIoT scenarios to offload computation tasks for IIoT devices, which also greatly saves much energy consumption.

Currently, computation offloading performance optimization has attracted significant attention [7]–[11]. Munoz *et al.* [12] presented a framework for the joint optimization of the radio and computation resource usage, assuming multiple antennas are available at the mobile terminal. In [13], an offloading strategy is proposed with one IoT device and one F-AP for minimizing task execution cost. In [14], a computation offloading scheme is developed with a single mobile device, which assumes all the offloading tasks are executed at the F-AP. Although these prior studies achieve good performance, only a single device or a single F-AP is considered.

Furthermore, some researchers study more practical scenarios with multiple devices and multiple F-APs. Jŏsilo *et al.* [15] proposed a polynomial complexity algorithm to compute allocations of cloud resources and make offloading decisions for multiple devices. The computational resource and allocated power for each device is optimized in [16] under latency and energy constraints. Nevertheless, the optimization of F-AP selection is not considered in [15] and [16]. In [17], a computation offloading policy is proposed by jointly optimizing offloading decisions, physical resource block allocation, and computation resource allocation, which selects a computation node with the lowest offloading overhead. A heuristic search method, which iteratively adjusts binary offloading decisions, is studied in [18]. Though computation offloading node selection is addressed, the impact of limited computation capacity of F-APs is not taken into account in both articles, and meanwhile the proposals are not suitable for IIoT environments due to high complexity.

In fact, in practical IIoT scenarios, channel quality, requested tasks, and energy status at an IIoT device are all dynamic over time, leading traditional computation offloading optimization approaches with high complexity inapplicable. Fortunately, machine learning frameworks can be exploited to accomplish efficient network performance optimization [19]–[21]. Particularly, being capable of learning policies directly from high dimensional raw state data, deep reinforcement learning (DRL) has been applied in dynamic computation offloading scenarios. In [22], a DRL-based online offloading framework is proposed to learn the optimal offloading decision. An online computation offloading policy based on DRL under random task arrivals is presented in [23]. In [24], Q-learning and DRL-based schemes are combined to jointly optimize the offloading policies and resource allocation.

In [22]–[24], the proposals are all based on a single-agent setting, and hence, the number of actions can grow extremely with network scale enlarging. In addition, they mainly optimize the allocation of computing resource and transmission power, without considering the computation offloading node selection. Compared to single-agent DRL, multi-agent DRL can well overcome the explosion of action space by using each DRL for a specific action dimension and meanwhile it can implement in a distributed fashion based on local information.

In this article, a multi-agent DRL based F-AP selection approach is developed for computation offloading in IIoT, which intends to minimize long-term system energy consumption. Through moving complexity from online to offline, low-complexity online F-AP selection is realized. Moreover, due to limited computing capability of each F-AP, when the offloaded computation burden is larger than its computation capacity, it needs to decide, which tasks are forwarded to the cloud. The contributions of this article are as follows.

1) In order to alleviate the computation burden on IIoT devices, F-APs with fog computing capability are deployed to realize computation offloading. The system model features multiple devices and multiple F-APs and meanwhile the dynamics of channel states and task demands are considered, which are modeled by Markov process. Then, a computation offloading optimization problem under per-F-AP computation capability constraint is formulated, aiming at minimizing long-term system energy consumption.
2) To make the problem more tractable, the concerned problem is further decoupled into an offloading request forwarding subproblem device-F-AP association and an F-AP selection subproblem. The former is addressed by a low-complexity greedy algorithm. For the latter, a multi-agent DRL based approach is developed, which builds a DRL model for each IIoT device to identify its F-AP for task offloading, and each model comprehensively takes networkwise device-F-AP association, local channel gain and the characteristics of the offloaded task into account. By transforming online decision-making complexity into offline training complexity, the proposal can quickly respond to the dynamics of IIoT environments.
3) By simulation, the impact of training batch size on system performance is illustrated, and the proposal is compared with other benchmark schemes including exhaustive search, genetic algorithm based schemes, and so on. Furthermore, the generalization capability of the DRL based approach is verified.

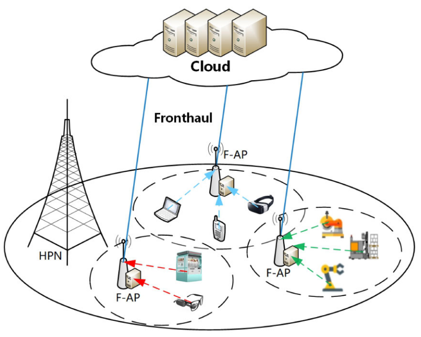


Fig. 1. Fog computing enabled IIoT scenario.

The remainder of this article is organized as follows. In Section II, the system model is described. In Section III, the long-term energy consumption minimization problem of computation offloading is formulated. Section IV proposes multi-agent DRL and genetic algorithm based approaches to F-AP selection and a low-complexity greedy algorithm for F-AP's task forwarding decisions. Section V presents the simulation results. Finally, Section VI concludes this article.

II. System Model

The considered IIoT scenario is shown in Fig. 1, which consists of F single-antenna F-APs, a high-power-node (HPN), U single-antenna IIoT devices, and a remote cloud center. The set of F-APs and IIoT devices are denoted by $\mathcal{F} = \{1, 2, \ldots, F\}$ and $\mathcal{U} = \{1, 2, \ldots, U\}$, respectively. The HPN with a large coverage range is responsible for control information delivery. To reduce the burden on energy-constrained IIoT devices, which is incurred by computation task execution, such as industrial data analysis to detect some anomalies, tasks are offloaded to F-APs for local computation. The system operation time is divided into consecutive offloading durations of equal time length T, and durations are indexed by t. For each offloading duration t, each device u requests one task characterized by $(O_u(t), D_u(t))$. Specifically, $O_u(t)$ (in bits) represents the data volume to be uploaded for task execution and $D_u(t)$ indicates the number of CPU cycles needed for completing device u's task.

In this article, we try to optimize the system energy consumption caused by computation offloading. Since the amount of downloaded data can be quite small, such as an alarm signal, the energy consumption incurred by result downloading is assumed to be ignored and we primarily focus on energy consumption incurred by task uploading and task computing. Denote T_1 and T_2 as the time length of task uploading phase and task

TABLE I
SUMMARY OF NOTATIONS

Notation	Definition
$\mathcal{F}$	The set of F-APs
$\mathcal{U}$	The set of IIoT devices
$\mathcal{N}_u$	The set of computing tasks potentially requested by device u
$O_u(t)$	The data volume uploaded by device u in offloading duration t
$D_u(t)$	The CPU cycles needed for completing device u's task in offloading duration t
T_1	The time length of task uploading phase
T_2	The time length of task computing phase
$b_{u,f}(t)$	F-AP selection indicator for device u and F-AP f
$h_{u,f}(t)$	Channel gain between device u and F-AP f
$p_u(t)$	The transmit power of device u
$N_f(t)$	The number of devices connected to F-AP f
$x_u(t)$	Offloading request forwarding indicator for device u's task
φ_f	The CPU frequency of F-AP f
φ_{cloud}	The CPU frequency of the cloud center
δ_f	The computing constant factor of F-AP f
δ_{cloud}	The computing constant factor of the cloud
$E_u^w(t)$	Energy consumption incurred by the wireless transmission of device u
$E_{u,f}^{ex}(t)$	Energy consumption for executing device u's task at F-AP f
$E_u^{fr}(t)$	Fronthaul transmission energy consumption of device u
$E_{u,cloud}^{ex}(t)$	Energy consumption for executing device u's task in the cloud

computing phase, respectively, which satisfies $T_1 + T_2 = T$. Moreover, in the considered system, channel state and the task requested by each IIoT device are dynamically changing across offloading durations, which is captured by Markov Process. Define $b_{u,f}(t)$ as a 0-1 variable indicating whether device u's task is offloaded to F-AP f in offloading duration t and define $\mathbf{b}_u(t) = [b_{u,1}(t), \ldots, b_{u,F}(t)]$. In addition, in each task executing phase, the computing capability of each F-AP f is constrained by C_f in CPU cycles. In the following, channel state, device task request, and energy consumption models will be elaborated. Important notations are listed in Table I.

A. Channel State and Device Task Request Dynamics

As mentioned previously, wireless channel states and device task requests dynamically change over offloading durations, and they both have certain randomness. Specifically, Celik and Modiano [25] applied Markov process to characterize the time-varying channel states, while the task arrival and requesting process are also modeled by Markov process in [32], [26] and [27], respectively. Thus, in this section, the dynamics of channel states and device task requests follow similar assumptions. For device u, define the set of its potentially requested computing tasks as $\mathcal{N}_u = \{n_{u1}, \ldots, n_{uz}, \ldots, n_{uZ}\}$. Define $Pr_{u,z'z} = Pr_u\{n_u(t) = n_{uz} | n_u(t-1) = n_{uz'}\}$ with $n_u(t)$ representing the task requested by device u in duration t, which is the probability that device u requests task n_{uz} in offloading duration t if device u has requested task $n_{uz'}$ in offloading duration $t-1$. Similarly, the transition probabilities for the channel state of each device-F-AP link can be defined.

B. Energy Consumption Model

During task uploading phase, all IIoT devices transmit to associated F-APs based on time-division multiple access. Assume the frequency bands occupied by each F-AP is nonoverlapping and, hence, there is no interference between devices. Denote the data rate of device u accessing F-AP f in duration t by $R_{u,f}(t)$, which can be expressed as

$$R_{u,f}(t) = \frac{1}{N_f(t)} W \log_2\left(1 + \frac{p_u(t)h_{u,f}(t)}{n_0}\right) \quad (1)$$

where $N_f(t)$ indicates the number of devices connected to F-AP f in duration t, $p_u(t)$ is the transmit power of device u, and W and n_0 represent the bandwidth and noise power, respectively. Recall that the time length of task uploading is T_1 and then $R_{u,f}(t) = \frac{O_u(t)}{T_1}$, based on which the required $p_u(t)$ can be derived. The wireless transmission energy consumption for device u can be finally got by

$$E_u^w(t) = \frac{p_u(t) \cdot T_1}{N_f(t)}. \quad (2)$$

According to [28], the energy consumption of F-AP f to execute one CPU cycle is given by $\varphi_f^2 \delta_f$, in which φ_f (in cycles/second) denotes the CPU frequency of F-AP f and δ_f is the computing constant factor [29]. Thus, when device u's task is executed at F-AP f, corresponding energy consumption is as follows:

$$E_{u,f}^{ex}(t) = D_u(t) \cdot \varphi_f^2 \cdot \delta_f. \quad (3)$$

Considering that each F-AP has limited computation capability, it is possible that some offloaded tasks have to be further forwarded to the cloud for remote computation, and we use $x_u(t) \in \{0, 1\}$ to indicate whether device u's task is executed remotely in duration t. Specifically, $x_u(t) = 1$ means device u's task is executed in the cloud while $x_u(t) = 0$ means the opposite. For tasks going to the cloud, the energy consumption is led by fronthaul transmission and task execution at the cloud. Denote ε the energy consumed by transmitting one bit over fronthaul. Then, fronthaul transmission energy consumption for device u with $x_u(t) = 1$ is given by

$$E_u^{fr}(t) = O_u(t) \cdot \varepsilon. \quad (4)$$

As for the energy consumption for executing device u's task at the cloud, it is calculated as

$$E_{u,cloud}^{ex}(t) = D_u(t) \cdot \varphi_{cloud}^2 \cdot \delta_{cloud} \quad (5)$$

where φ_{cloud} (in cycles/second) denotes the CPU frequency of the cloud center, and δ_{cloud} is the computing constant factor of the cloud.

Overall, for device u offloading its task to F-AP f, total energy consumption can be expressed as

$$E_{u,f}(t) = (1 - x_u(t))E_{u,f}^{ex}(t) + x_u(t)(E_u^{fr}(t) + E_{u,cloud}^{ex}(t)) + E_{u,f}^w(t). \quad (6)$$

III. Problem Formulation and Decomposition

Based on the above derivations, system energy consumption for one offloading duration t is calculated as

$$E_t = \sum_{u\in\mathcal{U}} \left(\sum_{f\in\mathcal{F}} b_{u,f}(t) \cdot E_{u,f}(t) \right). \tag{7}$$

Then, our concerned optimization problem is formulated below, which aims to minimize total system energy consumption across τ offloading durations

$$\begin{aligned}
&\min_{\{\boldsymbol{b}(t),\boldsymbol{x}(t)\}} \sum_{t=0}^{\tau-1} E_t\{\boldsymbol{b}(t),\boldsymbol{x}(t)\}\\
&(a1)\ \sum_{f=1}^{F} b_{u,f}(t) = 1, u\in\mathcal{U}, t\in\{0,1,\ldots,\tau-1\}\\
&(a2)\ \sum_{u\in\mathcal{U}} (1-x_u(t))\cdot b_{u,f}(t)\cdot D_u(t) \le C_f,\\
&\quad t\in\{0,1,\ldots,\tau-1\}\\
&(a3)\ b_{u,f}(t)\ \in\{0,1\}, u\in\mathcal{U}, f\in\mathcal{F}, t\in\{0,1,\ldots,\tau-1\}\\
&(a4)\ x_u(t)\ \in\{0,1\}, u\in\mathcal{U}, t\in\{0,1,\ldots,\tau-1\}
\end{aligned} \tag{8}$$

in which $\{\boldsymbol{b}(t)\} = \{b_{1,f}(t),\ldots,b_{U,f}(t)\}$, $\{\boldsymbol{x}(t)\} = \{x_1(t),\ldots,x_U(t)\}$, and C_f is the computing capability of each F-AP f in CPU cycles. Constraint $(a1)$ means that each device can only select one F-AP for computation offloading and constraint $(a2)$ indicates that each F-AP can only execute limited number of CPU cycles in each offloading duration.

Considering the large solution space induced by simultaneous optimization of $\{\boldsymbol{b}(t)\}$ and $\{\boldsymbol{x}(t)\}$, problem (8) is first studied under prefixed $\{\boldsymbol{b}(t)\}$ and equivalently transformed into a perduration local energy consumption minimization problem at each F-AP f with its request forwarding decisions as optimization variables, i.e., problem (9) in the following. If problem (9) is solved optimally, a mapping ϕ from $\boldsymbol{b}(t)$ to $\boldsymbol{x}(t)$ can be derived, which must hold when the objective value of problem (8) is optimal. By substituting $\boldsymbol{x}(t)$ with $\phi(\boldsymbol{b}(t))$, problem (8) is equivalently transformed to problem (10), whose solution space is greatly reduced, hence decreasing computation time for the deigned algorithms.

A. F-APs' Offloading Request Forwarding Subproblem

Under prefixed F-AP selection of devices in each offloading duration, problem (8) can be reduced to a local subproblem at each F-AP, aiming to optimize decisions on offloading request forwarding in each duration. The subproblem is as follows:

$$\begin{aligned}
&\min_{\{x_u(t)|u\in\mathcal{U}_f\}} \sum_{u\in\mathcal{U}_f} E_{u,f}(t)\\
&(b1)\ \sum_{u\in\mathcal{U}_f} (1-x_u(t))\cdot D_u(t) \le C_f\\
&(b2)\ x_u(t)\in\{0,1\}, u\in\mathcal{U}_f(t).
\end{aligned} \tag{9}$$

B. F-AP Selection Subproblem

Under the F-APs' joint request forwarding policy ϕ derived by solving the abovementioned subproblem, which is a mapping from $\{\mathcal{U}_1(t),\ldots,\mathcal{U}_f(t),\ldots,\mathcal{U}_F(t)\}$ or equivalently $\boldsymbol{b}(t)$ to $\boldsymbol{x}(t)$, problem (8) is transformed into the following F-AP selection subproblem described as:

$$\begin{aligned}
&\min_{\{\boldsymbol{b}(t)\}} \sum_{t=0}^{\tau-1} E_t(\boldsymbol{b}(t), \phi(\boldsymbol{b}(t)))\\
&(c1)\ \sum_{f=1}^{F} b_{u,f}(t) = 1, u\in\mathcal{U}, t\in\{0,1,\ldots,\tau-1\}\\
&(c2)\ b_{u,f}(t)\in\{0,1\}, u\in\mathcal{U}, f\in\mathcal{F}, t\in\{0,1,\ldots,\tau-1\}.
\end{aligned} \tag{10}$$

Algorithm 1: Low-Complexity Greedy Algorithm for F-APs' Offloading Request Forwarding.

```
$\forall u\in\mathcal{U}_f(t), x_u(t)=1$;
Sort device indexes in $\mathcal{U}_f(t)$ according to the value of $\alpha_u$
 in a descending order, and re-number the device indexes
 as $u_1, u_2, \ldots, u_{|\mathcal{U}_f(t)|}$;
**For** $i = 1 : |\mathcal{U}_f(t)|$:
let $x_u(t) = 0$;
  **If** $\sum_{l=1}^{i}(1-x_u(t))D_{u_l}(t) > C_f$ do:
    let $x_u(t) = 1$;
  **End If**
**End For**
```

IV. Algorithm Design

In this section, subproblem (9) is handled first by a low complexity greedy algorithm, and then a multi-agent DRL Based algorithm is proposed to select F-APs for devices to perform computation offloading, taking the dynamics of channel states, and device task requests into account.

A. Greedy Algorithm for F-APs' Decisions on Request Forwarding

In this subsection, a low-complexity greedy algorithm is designed to optimize F-APs' decisions on which offloading requests are forwarded to the cloud. For offloading duration t, the objective of F-APs' offloading request forwarding can be reformulated as

$$\begin{aligned}
&\sum_{u\in\mathcal{U}_f(t)} E_{u,f}(t)\\
&= \sum_{u\in\mathcal{U}_f(t)} \alpha_u(t)x_u(t) + \sum_{u\in\mathcal{U}_f(t)} E_{u,f}^{ex}(t)\\
&+ \sum_{u\in\mathcal{U}_f(t)} E_{u,f}^{w}(t)
\end{aligned} \tag{11}$$

in which $\alpha_u(t) = E_u^{fr}(t) + E_{u,cloud}^{ex}(t) - E_{u,f}^{ex}(t)$. It is found that the coefficient α^u of $x_u(t)$ is actually a constant, based on which a low-complexity greedy algorithm is proposed in **Algorithm 1**.

B. Algorithm Design for F-AP Selection

1) Genetic Algorithm Based Design: Generally, integer programming like subproblem (10) can be solved by heuristic methods, e.g., genetic algorithm (GA) that has been widely adopted in the wireless domain and achieves good performance. In this part, a GA-based F-AP selection scheme for computation offloading is developed shown in **Algorithm 2**, which performs online optimization on a perduration basis. Here, **Algorithm 2** appears as a performance baseline to provide an approximated upper performance bound for the multi-agent DRL algorithm to be proposed. To help readers understand the application of GA in computation offloading problems, the details of the algorithm are illustrated in the following.

Genetic algorithm adopts the idea of survival of the fittest as its evolution principle. At the start of the evolution process, a set of solutions is initialized and are further optimized subsequently through genetic operations, i.e., selection, crossover, and mutation, until reaching a satisfactory solution [30]. Particularly, the crossover and mutation operations can maintain population diversity and extend the searching region. To utilize GA for F-AP selection optimization, the following items should be emphasized.

1) *Individual and fitness function:* In GA, an individual is defined by a gene sequence that represents a solution to the concerned problem. In our considered scenario, problem (10) is decoupled over offloading durations and a solution to F-AP selection for current duration is encoded into a sequence consisting of 0 and 1. In addition, in order to evaluate how appropriate the individual is, the fitness function is as follows:

$$\text{Fitness function} = \begin{cases} c_{\min} + g, & \text{if } g + c_{\min} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

where g is the objective function defined as the reciprocal of total energy consumption E_t, while $c_{\min}$ can be an input value or the minimum value of g in all current generations or the nearest L generations.

2) *Initialization and individual selection:* A 0-1 sequence is randomly generated for each individual as its initial genes. Based on the fitness value of individuals, individuals with high fitness value are chosen by means of roulette [31], which will then participate the following crossover and mutation operations.

3) *Crossover and mutation:* Crossover and mutation are applied iteratively to increase the diversity and produce better solutions to the problem. Crossover is a step of switching two individuals' genes and generating two new individuals, it applies the crossover probability p_c to control whether single-point crossover, multipoint crossover, etc, is used to produce new individuals [32]. For the mutation operation, an individual is chosen first. Then, for each element of genes, it mutates with the mutation probability p_m can be mutated from 0 to 1 or 1 to 0.

For the genetic algorithm, it contains point mutation, one point crossover and roulette wheel selection. Denote G as the number of generations, K as the population size, and M as the size of the individuals. Then, the complexity of GA is $\mathcal{O}(G(KM + K + K))$ and is further simplified to $\mathcal{O}(GKM)$.

Algorithm 2: GA Based F-AP Selection.

```
1:  Input: The size of the population $K$, the generation of
    GA $G$, and the parameters of GA $p_m$ and $p_c$.
2:  Initialization: Initialize the genes of $K$ individuals
    and set the number of generation $g = 0$. Set $I_{best}$ as a
    randomly selected individual.
3:    For g=0:G-1
4:      Evaluate the population, i.e., calculate the fitness
        value of each individual.
5:      Find the best individual $I^g_{best}$ in the current
        generation, if $I^g_{best}$ outperforms $I_{best}$,
        $I_{best} = I^g_{best}$.
6:      Select $Q$ individuals based on the principles of
        roulette wheel for the following crossover and
        mutation operations. If $I_{best}$ is not included in
        these individuals, replace the worst individual with
        $I_{best}$.
7:      Choose two individuals randomly and conduct the
        crossover operation with the crossover probability
        $p_c$.
8:      For all the individuals, the mutation operation with
        probability $p_m$ is performed.
9:    $g = g + 1$
10:   For end
 Output: The best individual and its fitness value.
```

2) Multi-agent DRL Based Design: Considering the potential high online optimization latency caused by genetic algorithm based F-AP selection, it is essential to develop more advanced approaches that can make quick response to dynamic IIoT environments. As an emerging tool for wireless network control and optimization, DRL has attracted a lot of attention recently. By training DRL models offline and then making decisions online based on trained models, decision making time can be greatly shortened owing to the avoidance of iterative optimization procedures. In addition, compared to traditional Q learning, one advantage of DRL is that it can directly learn control policies from high dimensional raw network data [33]. Hence, we further present a DRL-based F-AP selection design, and it features a multi-agent setting to overcome the curse of dimensionality of the action space in the single-agent setting whose number of actions is F^U. Specifically, a DRL model is created for each IIoT device to selection a proper F-AP for it.

In order to apply DRL, the state space, action space, and reward function are defined first.

1) *State space:* For each IIoT device, the input state of its corresponding DRL model in offloading duration t is denoted as $\mathcal{S}_u(t) \triangleq \{\mathbf{b}(t), \mathbf{h}_u(t), O_u(t), D_u(t)\}$, where $\boldsymbol{b}(t)$ represents the networkwise F-AP selection state at the beginning of offloading duration t and $\boldsymbol{h}_u(t) = \{h_{u,1}(t), h_{u,2}(t), \ldots, h_{u,F}(t)\}$ denotes the channel gain vector between device u and each F-AP.

2) *Action space:* For the DRL model associating with device u, each of its actions represents an F-AP selection with fth action meaning device u offloads its task to F-AP f. The action taken by DRL agent u in offloading duration t is denoted as $a_u(t)$.

3) *Reward function:* The reward in offloading duration t $r_u(t)$ is taken as the negative of total system energy consumption, i.e, E_t defined in (7).

The training process of the proposed multi-agent DRL based F-AP selection optimization design is shown in **Algorithm 3**, where each decision step corresponds to an offloading duration. At the beginning of duration t, each DRL agent u takes $\mathcal{S}_u(t)$ as the input of its deep Q network to output the Q-values $Q(\mathcal{S}_u(t), a_u(t), \theta_u(t))$ corresponding with each action, where $\theta_u(t)$ represents the DQN weights. The action is further selected based on ε-greedy scheme. After that, each device u offloads its requested task to the desired F-AP according to the action selection result, and each F-AP further forwards some received requests to the cloud according to Algorithm 1. At the start of duration $t+1$, observed state is updated from $\mathcal{S}_u(t)$ to $\mathcal{S}_u(t+1)$ and DRL agent u stores the interaction experience composed of $\mathcal{S}_u(t)$, $\mathcal{S}_u(t+1)$, E_t, and $a_u(t)$ into a replay memory. Every multiple interactions with the environment, a minibatch of experiences are randomly sampled from the replay memory to update DQN's parameters, aiming to minimize the mean-squared-error between the target Q value and the predicted Q value. Specifically, the loss function is given by

$$L(\theta_u) = \mathbb{E}\left\{\left(y_u - Q\left(\mathcal{S}_u, a_u, \theta_u\right)\right)^2\right\} \tag{13}$$

where $y_u = r_u + \gamma \max_{a_u} \hat{Q}(\mathcal{S}_u, a_u, \hat{\theta}_u)$ and $\hat{Q}$ is the Q value output by the target DQN whose weights will be reset as the DQN's weights every larger period.

3) Practical Implementation: The proposed multi-agent DRL based scheme is trained offline and then makes F-AP selection decisions online. After the training completes at the cloud, all the U DRL models are sent to each F-AP. Initially, each IIoT device accesses the nearest F-AP and sends its offloading request that contains the input date size and computing burden information. By information exchange with other F-APs via interface like X2, each F-AP can acquire the state information necessary for the DRL agent of each accessed IIoT device. For device u accessing F-AP f, if the decision outcome is F-AP f', device u will be handed over from F-AP f to F-AP f'. After all the devices finish F-AP selection, subsequent data uploading phase and task execution phase will start, and similar process will repeat later on.

4) Complexity Analysis: For the multi-agent DRL-based F-AP selection algorithm, the training process runs offline and is performed in the cloud that is with sufficient computation resource. Hence, we mainly pay attention to the complexity for online decision making. In multi-agent DRL, a DRL agent with a DQN is created for each device and these DQNs can run in parallel across F-APs. Hence, the total complexity relies on the complexity of a single DQN incurred by calculating the output based on the input and finding the action with the maximal Q value.

Considering that the input state includes the network wise F-AP selection state, local channel gain, the data volume to be uploaded, and the number of CPU cycles needed for completing the device's task, the number of input neurons of the DQN for each device is $(UF + F + 2)$ with the F-AP selection state being represented by a 0-1 vector. Moreover, there are F neurons in the output layer and H neurons in the hidden layer. Denote the number of hidden layers as L. Then, for the DQN related to a device, the number of multiplication operations is $(UF + F + 2) \times H + (L-1) \times H \times H + H \times F = \mathcal{O}(H(UF + F + 2 + (L-1)H + F))$ and can be simplified into $\mathcal{O}(H(UF + (L-1)H))$. In addition, the complexity of applying the activation function is $\mathcal{O}(L \times H)$. Therefore, the complexity for computing the DQN output is $\mathcal{O}(H(UF + LH - H + L))$ and can be simplified into $\mathcal{O}(H(UF + LH))$. Since the operation for selecting the action with the maximal Q value is with complexity $\mathcal{O}(F)$, the total complexity is $\mathcal{O}(H(UF + LH))$.

Algorithm 3: Multi-Agent DRL Based F-AP Selection.

```
1:  For device $u = 1, 2, \ldots, U$:
2:    Create a DRL agent for device $u$ and initialize the
      weight parameter $\theta_u$ and $\hat{\theta}_u$ for the DQN and target
      DQN, respectively.
3:  End For
4:  For epoch $e = 0, 1, \ldots, E-1$:
5:    Initialize observed state $\mathcal{S}_u(0)$ for each DRL agent $u$;
6:    For decision step $t = 0, 1, \ldots, \tau - 1$:
7:      For DRL agent $u = 1, 2, \ldots, U$:
8:        Generate a random number $x$ between 0 and 1;
9:          If $x \le \varepsilon$ or the replay memory has not
          accumulated enough interaction samples:
10:           Agent $u$ randomly selects an action;
11:         Else
12:           Agent $u$ selects the action $a_u(t)$ satisfying
          $a_u(t) = \text{argmax}_{a_u} Q(\mathcal{S}_u(t), a_u, \theta_u(t))$
13:         If end
14:     For end
15:         For each agent $u$, its observed state transits
          from $\mathcal{S}_u(t)$ to $\mathcal{S}_u(t+1)$ according to the F-AP
          selection of all the agents and the dynamics of
          its own channel states and task requests.
16:         Each agent $u$ stores the reward $E_t$ together
          with $\mathcal{S}_u(t)$, $\mathcal{S}_u(t+1)$ and $a_u(t)$ as an
          interaction sample into its replay memory.
17:         When the capacity of the replay memory is
          full, the earliest sample is abandoned.
18:         Every $n$ decision steps, each agent randomly
          fetches a mini-batch of the interaction samples
          from its replay memory and performs gradient
          descent with regard to the loss function in (13).
19:         Every $N > n$ decision steps, let $\hat{\theta}_u = \theta_u$,
          $\forall u \in U$.
20:   End For
21: End For
```

V. Simulation Results and Analysis

In simulation, since the model training time for a large scale network can be too long and our hardware resource is limited, the number of F-APs and the number of IIoT devices are set

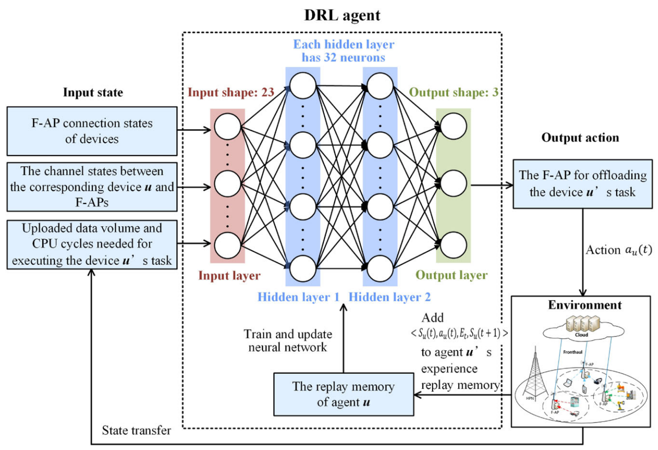

DRL agent
Each hidden layer has 32 neurons
Input state
Input shape: 23
Output shape: 3
F-AP connection states of devices
The channel states between the corresponding device u and F-APs
Uploaded data volume and CPU cycles needed for executing the device u' s task
Output action
The F-AP for offloading the device u' s task
Action $a_u(t)$
Input layer
Output layer
Hidden layer 1
Hidden layer 2
Environment
Cloud
Fronthaul
F-AP
HPN
Add $<S_u(t), a_u(t), E_t, S_u(t+1)>$ to agent u' s experience replay memory
Train and update neural network
The replay memory of agent u
State transfer

Fig. 2. Illustration of the DRL model for agent u.

to 3 and 6, respectively. The total number of tasks potentially requested by each device is 5. By referencing [28], the CPU frequency of 3 F-APs are set to $23 \times 10^8, 25 \times 10^8, 21 \times 10^8$ Hz, respectively. The CPU frequency of the cloud center is 6×10^9 Hz. In addition, other parameters like computing constant factors, data volume, and CPU cycles are set as per [29].

As mentioned previously, channel states change dynamically over time. Specifically, several numbers are randomly generated for each F-AP-device link as the possible values of the corresponding channel gain, which are then normalized with the noise power with power spectral density -174 dbm/hz [34]. After each offloading duration, channel states and the task request of each device will vary based on their transition probability matrices, which are in shape of 4×4 and 5×5, respectively. The time lengths of data uploading phase and task execution phase is 0.04 s and 1 s, respectively. Based on the CPU frequency, the computation capability of F-APs and the cloud can be further calculated.

The DRL model for device u is illustrated in Fig. 2. Based on the simulation parameter setting, there are 6 DRL agents. For each agent, its DQN is a dense neural network constructed by one input layer, two hidden layers, and one output layer, and the input size and the output size are 23 and 3, respectively. There are 32 neurons in each hidden layer, and ReLu is utilized as the activation function. By comparing the simulation performance under different learning rates 0.1, 0.001, and 0.0001, it is found that 0.0001 is the most appropriate value. Moreover, since the accumulated system energy consumption is considered for each epoch with limited durations, γ is set to 1. To achieve a stable each epoch with limited durations, gamma is set to 1. To achieve a stable training, it is better to slow down the frequency of updating the target DQN. Therefore, we set 900 as the number of steps to update target DQN, which outperforms other settings like 90. By referencing [35] and [36], the capacity of replay memory and initial steps are set as 3000 and 1000. All other parameters in the simulation are listed in Table II.

TABLE II
SIMULATION PARAMETERS

Parameter	Value	Parameter	Value
The learning rate of Adam optimizer	0.0001	Normalized channel gain	$[1,5]*10^3$
The capacity of replay memory	3000	Uploaded data volume for task execution	$[1,5]*10^5$ bits
The number of steps to update target DQN	900	The number of CPU cycles needed for completing task	$[21,45]*10^8$
The number of steps to update DQN	3	Channel bandwidth	$20 * 10^6$ Hz
Computing constant factor of cloud	10^{-30}	Computing constant factor of F-APs	$1.5*10^{-30}$
Discounted factor	1	Energy consumption for per-bit fronthaul transmission	10^{-8}
Noise	10^{-13} W	The initial steps to populate replay memory by random action selection	1000
The number of DRL agents	6	The shape of the input layer	23
The number of hidden layers	2	The number of neurons in each hidden layer	32

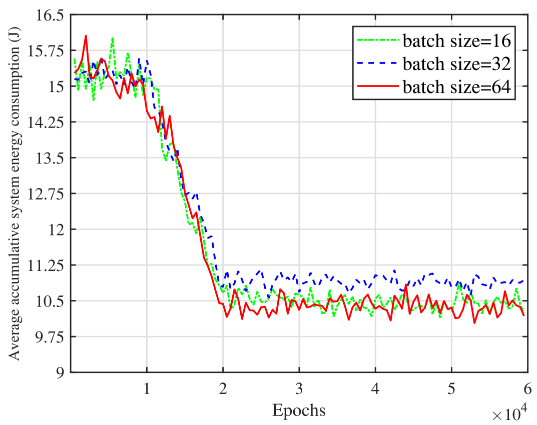


Fig. 3. System energy consumption under different batch sizes.

A. Impacts of Minibatch Size on System Performance

In this section, we investigate the impacts of batch size for gradient descent training on system performance, which is shown in Fig. 3. The DRL model corresponding to each IIoT device is trained with 60 000 epochs and each epoch contains 30 offloading durations. The accumulative performance in the figure is the total energy consumption in a training epoch and the initial

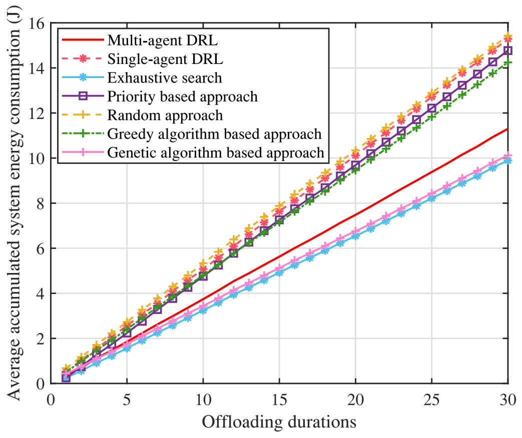


Fig. 4. Performance comparison with benchmark schemes.

TABLE III
ONLINE DECISION TIME FOR ONE OFFLOADING DURATION

Scheme Name	Consumed Time
Multi-agent DRL	0.00971s
Exhaustive search	0.34294s
Priority based approach	0.00064s
Random based approach	0.00055s
Greedy algorithm based approach	0.00053s
Genetic algorithm based approach	3.42052s

environment states of all the epochs are the same. As shown in Fig. 3, the curves first descend and then fluctuate slightly. This is because the Q value estimation is gradually improved with continuous training by minimizing the loss function in (13) and the performance will become stable once the model has been fully trained. In addition, it can be seen that our proposal is not sensitive to the batch size setting. In the following, the used DRL models are all trained with batch size 64 that has the best performance.

B. Performance Comparison With Other Baselines

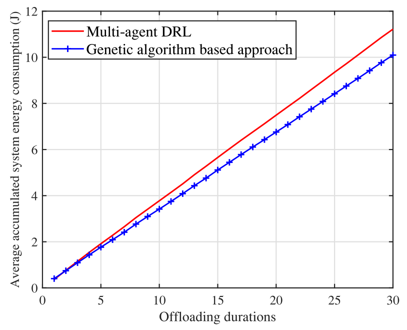
To verify the superiority of multi-agent DRL based F-AP selection, the following schemes are adopted for comparison in our simulation.

1) *Exhaustive search*: In each offloading duration, the cloud collects global information and then calculates the system energy consumption for all the 729 network wise F-AP selection solutions. Devices are notified by the HPN with the solution that has the lowest system power consumption.
2) *Priority-based F-AP selection*: In each offloading duration, each device accesses an F-AP based on both channel state and the computing capability of F-APs.
3) *Random-based F-AP selection*: In each offloading duration, each device randomly chooses an F-AP for task offloading.
4) *Greedy algorithm based F-AP selection*: In each offloading duration, the cloud collects global information and then greedily selects the F-AP that each device accesses.
5) *Genetic algorithm based F-AP selection*: In each offloading duration, the cloud collects global information and then uses the GA-based approach proposed in Section IV-B with the number of iterations being 100.
6) *Single-agent DRL-based F-AP selection*: In this baseline, each output action represents an F-AP selection for all the devices. The same as multi-agent DRL, the single-agent DRL model is trained also with 60 000 epochs.

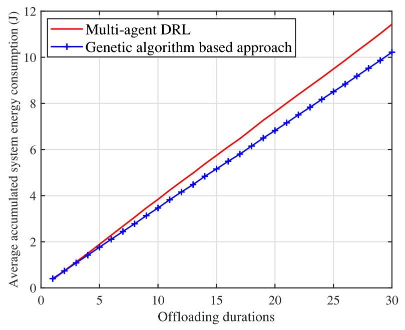
The comparison results with various schemes is shown in Fig. 4, and the time needed for optimizing networkwise F-AP selection by all the schemes for one offloading duration is listed in Table III. In Fig. 4, as the number of offloading durations increases, the curve under each scheme goes higher. Because the vertical axis variable is the accumulative energy consumption. Compared with multi-agent DRL, exhaustive search, and genetic algorithm can improve performance by 12.42% and 10.4%, respectively. Generally, exhaustive search scheme finds the optimal solution so it always has the minimum energy consumption, while genetic algorithm based F-AP selection leads to the near-optimal solution. Therefore, these two schemes both perform better than the multi-agent DRL algorithm. However, as shown in Table III, the DRL algorithm costs much less time for F-AP selection in a duration. On the other hand, relative to priority-based selection, random selection, and greedy algorithm, our proposal can improve performance by 23.49%, 26.78%, and 20.69%, respectively. The three schemes all achieve significantly higher energy consumption, since they are simply designed and not well optimized. In addition, compared to the single DRL agent setting, our proposed multi-agent DRL method achieves much less energy consumption. This is because the action space of the single DRL agent is quite large and it is hard to fully explore the action space, leading to poor Q value estimation. Overall, the proposed multi-agent DRL approach well balances the system performance and time complexity.

Note that in previous results, the initial states of each training epoch for DRL models in training and testing phases are the same. Hence, we intend to further investigate the generalization capability of our proposal. In the context of supervised learning, generalization refers to the adaptive ability to new samples. Good generalization ability means for data that does not have the same pattern as that of the learning set, the trained model can still provide appropriate output. In this article, we interpret the generalization of trained DRL models as the ability to make proper decisions in an environment with different initial states. Specifically, during training phase, assume that all devices will connect to F-AP ♯1 at the beginning of each training epoch. When model training completes, three environments with different initial connection states will be considered for the testing phase as follows.

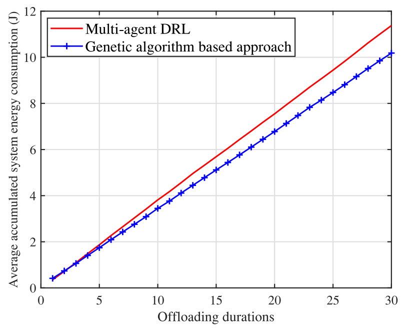
1) *Initial connection state 1*: All the devices connect to F-AP ♯2 initially.



(a) Performance comparison under state 1



(b) Performance comparison under state 2



(c) Performance comparison under state 3

Fig. 5. Performance comparison between the proposal and genetic algorithm based approach under various initial states of the learning environment.

2) *Initial connection state 2*: Initially, device ♯1 and ♯3 connect to F-AP ♯1, device ♯2 and ♯5 connect to F-AP ♯2, and device ♯4 and ♯6 connect to F-AP ♯3.
3) *Initial connection state 3*: Initially, device ♯1, ♯3, and ♯4 connect to F-AP ♯1 and device ♯2, ♯5, and ♯6 connect to F-AP ♯2.

As shown in Fig. 5, compared with multi-agent DRL, genetic algorithm can improve performance by 10.02%, 10.62%, and 10.48%, respectively. As acknowledged before, the performance ratio improved by GA is 10.4% when training and testing phase apply the same initial states. Therefore, we can conclude that the proposed multi-agent DRL based F-AP selection approach can well fit into the considered different environments and, hence, has a good generalization capability.

VI. Conclusion

In this article, a DRL-based F-AP selection approach was proposed for computation offloading scenarios in IIoT, aiming to minimize long-term system energy consumption. Specifically, to overcome the scalability issue faced by traditional DRL approaches, a multi-agent setting was adopted, which creates a DRL model for each IIoT device. With the trained DRL models, each F-AP can optimize the F-AP selection for each device's task offloading within a short time based on dynamic network and device states. After F-AP selection, a low-complexity greedy algorithm was then applied at each F-AP to decide, which offloading requests were further forwarded to the cloud to meet local computing capability constraints. Via simulation, the impact of batch size on system energy consumption showed, and the effectiveness of the proposal was demonstrated by comparing with various baselines. Meanwhile, the generalization ability was verified as well.

References

[1] Q. Li, N. Zhang, M. Cheffena, and X. Shen, "Channel-based delay control in delay-constrained industrial WSNs," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 696–711, Jan. 2020.
[2] W. Liu, P. Popovski, Y. Li, and B. Vucetic, "Wireless networked control systems with coding-free data transmission for industrial IoT," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 1788–1801, Mar. 2020.
[3] M. Aazam, K. A. Harras, and S. Zeadally, "Fog computing for 5G tactile industrial internet of things: QoE-aware resource allocation model," *IEEE Trans. Ind. Informat.*, vol. 15, no. 5, pp. 3085–3092, May 2019.
[4] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial internet of things: Challenges, opportunities, and directions," *IEEE Trans. Ind. Informat.*, vol. 14, no. 11, pp. 4724–4734, Nov. 2018.
[5] H. Harb and A. Makhoul, "Energy-efficient sensor data collection approach for industrial process monitoring," *IEEE Trans. Ind. Informat.*, vol. 14, no. 2, pp. 661–672, Feb. 2018.
[6] M. Aazam, S. Zeadally, and K. A. Harras, "Deploying fog computing in industrial internet of things and industry 4.0," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4674–4682, Oct. 2018.
[7] S. Li *et al.*, "Joint admission control and resource allocation in edge computing for internet of things," *IEEE Netw.*, vol. 32, no. 1, pp. 72–79, Jan.-Feb. 2018.
[8] Y. Chen, N. Zhang, Y. Zhang, X. Chen, W. Wu, and X. S. Shen, "Energy efficient dynamic offloading in mobile edge computing for internet of things," *IEEE Trans. Cloud Comput.*, to be published, doi: 10.1109/TCC.2019.2898657.
[9] C. Pradhan, A. Li, C. She, Y. Li, and B. Vucetic, "Computation offloading for IoT in C-RAN: Optimization and deep learning," *IEEE Trans. Commun.*, vol. 68, no. 7, pp. 4565–4579, Jul. 2020.
[10] Z. Ning, X. Wang, J. J. P. C. Rodrigues, and F. Xia, "Joint computation offloading, power allocation, and channel assignment for 5G-enabled traffic management systems," *IEEE Trans. Ind. Informat.*, vol. 15, no. 5, pp. 3058–3067, May 2019.
[11] X. Xu, X. Zhang, H. Gao, Y. Xue, L. Qi, and W. Dou, "BeCome: Blockchain-enabled computation offloading for IoT in mobile edge computing," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4187–4195, Jun. 2020.
[12] O. Munoz, A. Pascual-Iserte, and J. Vidal, "Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading," *IEEE Trans. Veh. Technol.*, vol. 64, no. 10, pp. 4738–4755, Oct. 2015.
[13] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, Dec. 2016.
[14] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Trans. Commun*, vol. 65, no. 8, pp. 3571–3584, Aug. 2017.
[15] S. Jošilo and G. Dán, "Selfish decentralized computation offloading for mobile cloud computing in dense wireless networks," *IEEE Trans. Mobile Comput.*, vol. 18, no. 1, pp. 207–220, Jan. 2019.
[16] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.
[17] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang, "Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 66, no. 8, pp. 7432–7445, Aug. 2017.
[18] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, Jan. 2019.

[19] Y. Yang, Y. Wang, K. Liu, N. Zhang, S. Gu, and Q. Zhang, “Deep reinforcement learning based online network selection in CRNs with multiple primary networks,” *IEEE Trans. Indus. Informat.*, to be published, doi: 10.1109/TII.2020.2971735.

[20] Y. Sun, Y. Wang, J. Jiao, S. Wu, and Q. Zhang, “Deep learning-based long-term power allocation scheme for NOMA downlink system in S-IoT,” *IEEE Access*, vol. 7, pp. 86288–86296, 2019.

[21] Y. Sun, M. Peng, and S. Mao, “A game-theoretic approach to cache and radio resource management in fog radio access networks,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 10, pp. 10145–10159, Oct. 2019.

[22] L. Huang, S. Bi, and Y. J. Zhang, “Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks,” *IEEE Trans. Mobile Comput.*, to be published, doi: 10.1109/TMC.2019.2928811.

[23] M. Min, L. Xiao, Y. Chen, P. Cheng, D. Wu, and W. Zhuang, “Learning-based computation offloading for IoT devices with energy harvesting,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1930–1941, Feb. 2019.

[24] Y. Liu, H. Yu, S. Xie, and Y. Zhang, “Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 11158–11168, Nov. 2019.

[25] G. D. Celik and E. Modiano, “Scheduling in networks with time-varying channels and reconfiguration delay,” *IEEE/ACM Trans. Netw.*, vol. 23, no. 1, pp. 99–113, Feb. 2015.

[26] H. Khazaei, J. Misic, and V. B. Misic, “Performance analysis of cloud computing centers using m/g/m/m+r queuing systems,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 5, pp. 936–943, May 2012.

[27] A. Bonadio, F. Chiti, and R. Fantacci, “Performance analysis of an edge computing SaaS system for mobile users,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 2049–2057, Feb. 2020.

[28] C. You, K. Huang, H. Chae, and B.-H. Kim, “Energy-efficient resource allocation for mobile-edge computation offloading,” *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[29] T. Dang and M. Peng, “Joint radio communication, caching, and computing design for mobile virtual reality delivery in fog radio access networks,” *IEEE J. Sel. Areas Commun.*, vol. 37, no. 7, pp. 1594–1607, Jul. 2019.

[30] F. Guo, H. Zhang, H. Ji, X. Li, and V. C. M. Leung, “An efficient computation offloading management scheme in the densely deployed small cell networks with mobile edge computing,” *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2651–2664, Dec. 2018.

[31] H. Wei and X.-S. Tang, “A genetic-algorithm-based explicit description of object contour and its ability to facilitate recognition,” *IEEE Trans. Cybern.*, vol. 45, no. 11, pp. 2558–2571, Nov. 2015.

[32] Y. Guo, Z. Mi, Y. Yang, and M. S. Obaidat, “An energy sensitive computation offloading strategy in cloud robotic network based on GA,” *IEEE Syst. J.*, vol. 13, no. 3, pp. 3513–3523, Sep. 2019.

[33] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, “Application of machine learning in wireless networks: Key techniques and open issues,” *IEEE Commun. Surveys Tut.*, vol. 21, no. 4, pp. 3072–3108, Oct.–Dec. 2019.

[34] Y. Zou, J. Zhu, and R. Zhang, “Exploiting network cooperation in green wireless communications,” *IEEE Trans. Commun.*, vol. 61, no. 3, pp. 999–1010, Sep.Mar. 2013.

[35] Y. Sun, M. Peng, and S. Mao, “Deep reinforcement learning-based mode selection and resource management for green fog radio access networks,” *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1960–1971, Apr. 2019.

[36] Y. S. Nasir and D. Guo, “Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks,” *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2239–2250, Oct. 2019.

Yijing Ren received the bachelor’s degree in Internet of Things in 2018 from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, where she is currently working toward the master’s degree in information and communication engineering with the School of Information and Communication Engineering.

Her current research interests include the optimization of computation offloading in industrial Internet of Things, fog computing, multi-agent deep reinforcement learning, and the joint optimization of radio and caching resource.

Ms. Ren’s undergraduate thesis “Reinforcement Learning Based Resource Allocation in Fog Radio Access Networks” was granted the excellent undergraduate thesis award.

Yaohua Sun received the bachelor’s degree (hons.) in telecommunications engineering (with management) and the Ph.D. degree in communication engineering from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2014 and 2019, respectively.

He is currently an Assistant Professor with the State Key Laboratory of Networking and Switching Technology, BUPT. His current research interests include Internet of Things, edge computing, resource management, (deep) reinforcement learning, network slicing, and fog radio access networks.

Dr. Sun was the recipient of the National Scholarship in 2011 and 2017, and he has been reviewers for IEEE Transactions on Communications, IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE SYSTEMS JOURNAL, *Journal on Selected Areas in Communications*, *IEEE Communications Magazine*, *IEEE Wireless Communications Magazine*, IEEE WIRELESS COMMUNICATIONS LETTERS, IEEE COMMUNICATIONS LETTERS, and IEEE INTERNET OF THINGS JOURNAL.

Mugen Peng (Fellow, IEEE) received the Ph.D. degree in communication and information systems from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2005.

Afterward, he joined BUPT, where he has been a Full Professor with the School of Information and Communication Engineering since 2012. During 2014, he was also an Academic Visiting Fellow at Princeton University, USA. He has authored or coauthored more than 100 refereed IEEE journal papers and more than 300 conference proceeding papers. His main research interests include wireless communication theory, self-organization networking, heterogeneous networking, cloud communication, and Internet of Things.

Prof. Peng was a recipient of the 2018 Heinrich Hertz Prize Paper Award and the Best Paper Award in the JCN 2016 and IEEE Wireless Communications and Networking Conference 2015. He is currently or have been on the Editorial/Associate Editorial Board for the *IEEE Communications Magazine*, IEEE ACCESS, IEEE INTERNET OF THINGS JOURNAL, *IET Communications*, and *China Communications*.