Deep Q-Learning based Dynamic Resource Allocation for Self-Powered Ultra-Dense Networks

Han Li, Hui Gao, Tiejun Lv and Yueming Lu
Key Laboratory of Trustworthy Distributed Computing and Service (BUPT), Ministry of Education
Beijing University of Posts and Telecommunications, Beijing, China 100876
{lihan, huigao, lvtiejun, ymlu}@bupt.edu.cn

***Abstract*—Though enhancing the capacity and coverage of cellular networks to meet the explosive increasing of traffic demands, Ultra-Dense Network (UDN) suffers from great power consumption and greenhouse gas emission. Turning small base stations (SBS) off dynamically according to the network operating conditions is a promising solution to enhance energy efficiency (EE). Inspired by the success of Deep Q-learning Network(DQN) on solving complicated real-time control problems and the usage of energy-harvesting(EH) in UDNs, without any prior knowledge about energy arrival, data arrival and channel state information, we present a DQN-based framework for dynamic resource allocation in EH-UDN. Specially, joint optimization problem is formulated, considering both EE and quality of service (QoS) of the network. Then according to the application scenario, the action space, state space and reward function of the proposed DQN-based framework are defined and formulated. We evaluate the performance of the proposed framework by comparing it with the classic Q-learning framework via simulation. Numerical results show that our proposed scheme can enhance EE while taking good control of the QoS.**

I. Introduction

Ultra-Dense Network (UDN), which shortens the distances among access points and users, is a promising solution to meet the explosive increasing of traffic demands [1]. Though enhancing the capacity and coverage of cellular networks, UDN suffers from the increasing of total power consumption because of the densely deployment of base stations (BSs). As a result, energy harvesting UDN (EH-UDN) powered by green energy such as solar has been studied in recent years, which bears the potential to reduce the greenhouse emissions. For instance, the feasibility of EH small cell network (EH-SCN) is analyzed in [2], revealing into impact of BS density on the network utility. Aiming to optimize the green energy empowered mobile networks, strategies have been suggested by [3], such as EH awared user association, BS sleeping and so on.

Among all these promising technologies in [3], BS sleeping can enhance the EE of UDN while ensuring sufficient network capacity by optimizing BS ON/OFF scheduling according to the environment. Therefore, several researches focusing on optimizing the ON/OFF scheduling have been proposed [4]–[6].The traffic statistical information is adopted in [4] to optimize the BS sleeping scheduling. By using such statistical information of traffics, BSs can get previsions about the future traffic loads. Then BSs can decide the ON/OFF operation mode to enhance EE. Note that [4] does not take EH into consideration. However, adopting EH into networks, which can reduce greenhouse emission, is of great research significance. Thus, focusing on a new on-grid scenario, i.e., BSs are powered by both power grid and the harvested energy, an energy-aware traffic off-loading scheme is proposed in [5], [7]. User association, ON-OFF states of BSs, and power control are jointly optimized in this scheme according to the statistical information of energy arrival and traffic loads. Also assuming the prior knowledge on the statistical information of traffic and energy arrival, a dynamic programming algorithm is adopted in [6] to minimize the grid power consumption and blocking probability in the on-grid UDNs. Note that [5], [6] consider the on-grid scenario due to the consideration of instability of EH. However, with the development of photovoltaic (PV) solar panels and power allocation technologies, such instability can be smoothed so the off-grid SBS, i.e., BSs are powered solely by the harvested energy, can be achieved. Compared with on-grid ones, off-grid ones are more flexible and affordable, thus showing great potentials in practical applications.

It is noted that optimizing ON/OFF strategy to enhance EE while taking good control of the QoS is of great practical significance. Also, most of the existing works either assume statistical or complete information about EH and traffic to perform ON/OFF scheduling for EH BSs [4]–[6]. However, [2], [4] have shown that the real data and energy flows are so complex that finding the proper distributions to model them is none-trivial, thus degrading the performance of these statistic model based schemes. Therefore, in this paper we focus on solutions of off-grid scenario considering both EE and QoS of UDN in the presence of complete uncertainly of EH process and traffic coming. [8]

In consideration of the uncertainty of EH process and traffic conditions, reinforcement learning (RL) [9] has shown to be a viable tool to tackle the real-time dynamic resource allocation problems, and the advantages are two-ford. First, instead of optimizing the current reward, RL takes long-term rewards into consideration, which is very important for the time-variant dynamic systems, such as the considered EH-UDNs. Second, RL can achieve near-optimal performance by the interactive environment learning, even without the needs of prior information of the traffic, channel state information (CSI) and

This work is partially supported by the National Nature Science Foundation of China (NSFC) (No. 61671072), and the Joint Research Grant of NSFC for Overseas Chinese Scholars (No. 61629101).

978-1-5386-4328-0/18/$31.00 ©2018 IEEE

etc. Therefore, RL framework is suitable for the considered scenario, where no information about EH or traffic is assumed as prior. As an emerging paradigm, there are a few researches on adopting Q-learning into wireless communications. For example, [10] adopts Q-learning in optimizing SBSs ON/OFF policy to reduce the power consumption while meeting the traffic demands. It is noted that traditional RL algorithms such as Q-learning is based on the Q-value table $Q(s, a)$. Each element of this table represents the total reward for each state-action pair. The Q-table will become very large when the states grow, thus the RL algorithms will be hardly converge and lead to poor performance. The aforementioned challenges should be tackled when applying RL to resource allocation in the considered EH-UDN.

Recently, Deep Reinforcement Learning (DRL), which is the combination of RL and Deep learning, has shown to achieve better performance than conventional Q-Learning with acceptable complexity [11]. As DQN shows great potential in handling dynamic systems, we introduce DQN into EH-UDN to enhance EE while maintaining QoS. Specially, we focus on developing self-organizing and online algorithm for optimizing ON/OFF policies of EH-UDN. In summary, the main contributions of our work are three-fold:

1) Differing from the problem formulated in [5] where the prior knowledge about the user arrival and CSI are required, we formulate SBS ON/OFF problem in EH-UDN with off-grid mode into a dynamic and real-time optimization problem, and the prior knowledge about the energy arrival, user arrival and CSI are bypassed, which is practical and of interests.
2) In contrast to the Q-learning based scheme adopted in [10], we introduce DQN to solve the proposed dynamic optimization problem. Specifically, DQN learns a near-optimal policy to decide the ON/OFF states of SBSs to enhance the network EE while maintaining QoS according to the environment. Then the action space, state space and reward function are carefully defined to enhance EE. We apply a Deep Neural Network (DNN) to approximate the EE of the network after every action.
3) Extensive simulations are conducted to obtain useful insights. It is shown that the proposed DQN-based framework achieves higher EE and takes better control of QoS compared to the Q-learning algorithm adopted in [10]. Also, DQN-based scheme shows great potential in handling complex problem.

The rest of this paper is organized as follows: Section II describes our system model. Section III presents the problem formulation and the DQN-based framework. In section IV we compare our framework with the Q-learning framework. Finally, conclusions are provided in Section VI.

II. System Model

We consider the downlink of an EH-UDN. N small base stations (SBSs) powered exclusively by EH sources are deployed randomly in this area. A cloud computing agent is deployed to collect information. The agent decides SBS ON/OFF state to enhance EE. The SBS set is defined as $\mathcal{N} = \{1, 2, j, \ldots, N\}, |\mathcal{N}| = N$. Meanwhile, users arrive randomly and the user set at time instant t is defined as $I(t)$. The operation time line is divided into T time instants. The agent collects information and makes decisions at the beginning of each time instant. Also, as shown in [8], energy is harvested at the beginning of each time instant $t, t = 1, 2, 3, \ldots, T$. To handle the EH instability, each SBS is equipped with a battery to store the harvested energy. Note that SBSs can harvest energy regardless the state. However, the energy will be stored into batteries instead of using directly. That means, the energy harvested by SBS j at time instant t, namely $E_j^t \in \mathbb{R}^+, j \in \mathcal{N}$, can not be used before the time instant $(t+1)$, the harvested energy is stored in the battery which have capacity of B_{max}. Then the battery level of SBS j at time instant t is then given by

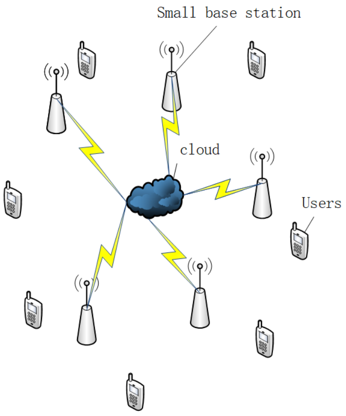

Figure 1. System model

$$B_j^{(t)} = \min(B_j^{(t-1)} - P_j^{tx}\rho_j^{(t)} + E_j^{(t-1)}, B_{max}) \qquad (1)$$

where $\rho_j^{(t)}$ shows the state of the SBS j at time instant t, e.g., when the SBS j is on, $\rho_j^{(t)}$ equals to one and the SBS provides services to the associated users, otherwise $\rho_j^{(t)}$ is equal to zero and the SBS will turn to sleep. P_j^{tx} is the transmission power of the SBS j .

A. Network performance

User i at location x and time instant t will choose the SBS with the minimum distance to associate. Then the user association rule is given by

$$j^*(i,t) = arg \min_j \left\| x_{ij}^{(t)} \right\| \qquad (2)$$

where $\left\| x_{ij}^{(t)} \right\|$ is the distance between SBS j and user i at time instant t. Then the set of users associated with the SBS j at time instant t can be defined by $I_j(t) = \{i \mid j^*(i,t) = j, i \in I(t)\}$,$I(t) = I_1(t) \cup I_2(t) \cdots \cup I_N(t)$. Then the traffic load of SBS j at time instant t is then given by

$$L_j(t) = \frac{|I_j(t)|}{|I_{max}|} \qquad (3)$$

where $|I_{max}|$ is the predefined maximum number of users to a SBS, which is a constant value. Assume that all SBSs use the same frequency to transmit, users associated to SBS j will suffer interference from all the other SBSs. interference. When user i is associated to the SBS j , the signal to interference plus noise ratio (SINR) of user i at time instant t is

$$\mathrm{SINR}_{i,j}(t) = \frac{P^{tx} h_{ij}^{(t)} \rho_i^{(t)} \left\| x_{ij} \right\|}{\sum\limits_{k \in \mathcal{N}, k \neq j} P^{tx} h_{ik}^{(t)} \rho_k^{(t)} \left\| x_{ik} \right\|^{-\alpha} + \sigma^2 W} \tag{4}$$

where $h_{ij}^{(t)} \sim exp(1)$ is the channel fading power between SBS j and user i. $\left\| x_{ij} \right\|^{-\alpha}$ denotes the path loss between SBS j and user i, where α is the path loss exponent. P^{tx} is the fixed transmission power of SBS. Meanwhile, σ^2 is the Gaussian noise and W is the transmission bandwidth. When user i is associated to SBS j at time instant t, the throughput achieved at time instant t with time duration τ is given by

$$R_{ij}^{(t)} = \tau W \log_2(1 + \mathrm{SINR}_{i,j}(t)) \tag{5}$$

where τ is time interval between two instants.

Then the total throughput achieved by the SBS j at time instant t is calculated by the sum of throughputs of users associated to SBS j, and is given by

$$R_j^{(t)} = \sum_{i \in I_j} R_{ij}^{(t)} \tag{6}$$

The achievable data rate of user i served by SBS j at time instant t is defined as the maximum bits that user i served by SBS j can receive per time interval, and is given by

$$c_{ij}(t) = BR_{ij}^t, \tag{7}$$

According to [8], traffic delay of user i can be defined as the time required for the SBS to transmit the data needed by user i. Then, the delay of SBS j can be defined as the sum of traffic delay of served users. If the UE's data requirement is M bits, the total traffic delay of SBS j at time instant t is

$$D_j(t) = \sum_{i \in I_j} \frac{M}{c_{ij}(t)} \tag{8}$$

Then we can get the total delay at the time instant t as

$$D(t) = \sum_{j \in \mathcal{N}} D_j(t) \tag{9}$$

B. Energy Consumption

The power consumption of a SBS can be divided into two parts, namely the power consumption of serving users and the power consumption of switching states. Taking traffic loads into consideration, the SBS j's power consumption of serving users at time instant t is given by

$$P_j^{(t)} = \left(L_j(t) P_j^{op} + P_j^{tx} \right) \rho_j^{(t)} \tag{10}$$

where P_j^{op} is the predefined maximum operation consumption of SBS j. Also, the power consumption of switching states of SBS j is $P_{cj}^{(t)} = \beta_j(t) P_c$, where P_c is a constant switching power consumption and $\beta_j(t)$ is:

$$\beta_j(t) = \begin{cases} 1 & \rho_j^{(t)} \neq \rho_j^{(t-1)} \\ 0 & otherwise \end{cases} \tag{11}$$

That means the power consumption of switching states exists only if the SBS changes the ON/OFF state.

Then the total power consumption at time instant t is :

$$P_{total}^{(t)} = \sum_{j=1}^{N} (P_j^{(t)} + P_{cj}^{(t)}) \tag{12}$$

As the energy consumption and throughput are modeled, according to [12], the EE at time instant t is defined as

$$EE(t) = \frac{\sum_j R_j^t}{P_{total}^{(t)}} \tag{13}$$

III. Problem Formulation

A. Optimization problem formulation

In this section, we formulate the joint optimization problem for EE as well as traffic delay. Due to lacking of prior knowledge about energy arrival, user arrival and CSI, the optimization problem can hardly be solved. Thus a DQN-based framework is proposed to solve this problem.

According to (9), (13), the utility function $G(t)$ is formulated as

$$G(t) = \zeta EE(t) - (1 - \zeta) D_j(t) \tag{14}$$

Taking both EE and QoS into consideration, the parameter ζ is introduced to take control of the trade-off between EE and delay.

Considering the equations above, the joint optimization problem can be formulated as follows:

$$\max_{\rho} \quad \textstyle\sum_{t=1}^{T} G(t) \tag{15}$$

$$s.t. \quad B_j^t \leqslant B_{max}, \forall j, \forall t \tag{16}$$

$$B_j^{t+1} = \min(B_j^t - \tau P_j^{tx} \rho_j^{(t)} + E_j^{t-1}, B_{max}), \quad \forall j, \forall t \tag{17}$$

$$\rho_j^{(t)} \in \{0, 1\}, \forall j, \forall t \tag{18}$$

$$|I_j(t)| \leqslant |I_{max}|, \forall j, \forall t \tag{19}$$

$$I_1(t) \cup I_2(t) \cdots \cup I_N(t) = I(t), \forall t \tag{20}$$

As the scenario is time-correlated, actions in time instant t may influence future utility function. Therefore, we choose the sum of utility functions of T time instants as the optimization goal to avoid greedy choices which may degrade the total performance. The meaning of optimization target and these constraints are explained below.

- The optimization target ρ is a matrix where the element of ρ at row j and column t is $[\rho]_{jt} = \rho_j^{(t)}$.
- (16) means the energy stored in batteries are limited.
- (17) shows how energy stored into batteries transfer between nearby time instants.

- (18) means that at time instant t, SBS j can only choose to turn on or off. '1' stands for on and '0' stands for off.
- (19) means each SBS can only serve limited users.
- (20) means each SBS serves a set of users and the union of the user sets at time instant t is the total user set at time instant t. In other words, the total user number in the scenario is time varying and the users of different SBSs are correlated.

This optimization problem may have an optimal solution only if the complete state information is known. Meanwhile, the optimization target is all actions in a period of time duration, which is rather complicated. Thus the optimization problem is transferred as follows

$$\max_{\rho^{(t)}} \sum_{k=t}^{T} \gamma^{k-t+1} G(k) \tag{21}$$
$$s.t. (16), (17), (18), (19), (20)$$

The optimization target $\rho^{(t)}$ is a vector where the jth element of $\rho^{(t)}$ is $[\rho]_j = \rho_j^{(t)}$. By $\rho^{(t)}$ instead of ρ, the global optimization problem is transferred to a real-time optimization problem, which is more realistic. Furthermore, since the transferred framework is real-time, the utility function at time instant t is the most important when optimizing the target at time instant t. To this end, the utility function of future time instants should also be considered. Thus, the discount factor γ is introduced to measure the importance of the future utility function. By doing so, the greedy choices can be avoided.

It is observed that the optimization problem is NP-hard. Nonetheless, the optimization problem still needs to know the complete information about the future time instants to reach the optimal solution, which is also not practical. Even if the method of solving the NP-hard problem can be found, the lacking of prior knowledge about energy arrival, user arrival and CSI may paralyze the method or degrade its achievable performance. RL-based algorithm, however, can get near-optimal solution just by interacting with environments and does not need the aforementioned prior knowledge. In this paper, we then turn (21) into a DQN-based framework for more practical applications.

B. DQN framework formulation

Briefly, there are three elements in the RL framework: action a, state s and reward r. For the scenario considered in this paper, we define state space, action space and reward of the DQN-based framework at time instant t as follows, namely

- State space: The decision is made by the cloud computing agent. The agent should know all information about SBSs to decide action. Thus the state of the agent at time instant t is given by
$$s_t = [E_1^{(t)}, B_1^{(t)}, L_1^{(t)}, R_1^{(t)}, D_1^{(t)}, \ldots, E_N^{(t)}, B_N^{(t)}, L_N^{(t)}, R_N^{(t)}, D_N^{(t)}] \tag{22}$$
that means the agent will know all the SBSs' harvested energy, battery levels, traffic loads, throughputs and delays.
- Action space: The SBSs only have two modes, i.e., on or sleep. So the action is
$$a_t = [\rho_1^{(t)}, \rho_j^{(t)}, \ldots, \rho_N^{(t)}] \tag{23}$$
If $a_t = [1, 0, 0, \ldots, 0]$, it means at time instant t, the agent chooses to turn the first SBS on and turn off the other SBSs to enhance the total rewards.
- Reward function: The reward needs to take the objective of the framework into consideration. Our goal is to maximize$\sum_{k=t}^{T} \gamma^{k-t+1} G(k)$, so the reward function is then given by
$$r_t = G(t \mid s = s_t, a = a_t) \tag{24}$$

In each time instant t, the agent will be at a state s_t and decide to make an action $a_t = \pi(s_t)$ according to the policy π. Then the agent will send control signals to the nearby SBSs to turn some SBSs ON/OFF and get the reward $G(t \mid s = s_t, a = a_t)$. RL uses value function $Q(s, a)$ which is the expected cumulative future (always with discounts) reward when the RL agents at state s and choose action a. $Q(s, a)$ is given by

$$Q(s_t, a_t) = \mathbb{E}\left[\sum_{k=t}^{T} \gamma^{k-t+1} G(t \mid s = s_t, a = a_t)\right] \tag{25}$$

When γ equals to 1, $Q(s, a)$ will be equal to the sum of the rewards and when γ equals to zero, $Q(s, a)$ only takes current reward into consideration. According to [13], $Q(s_t, a_t)$ can be rewritten as

$$Q(s_t, a_t) = G(t \mid s = s_t, a = a_t) + \gamma Q(s_{t+1}, a_{t+1}) \tag{26}$$

A neural network, e.g., a deep neural network (DNN) is used to approximate the $Q(s_t, a_t)$, so the Q-value at time instant t can be rewritten as $Q(s_t, a_t \mid \theta^Q)$ where θ^Q stands for the parameters of the network. After the approximation, the optimal policy $\pi^*(s)$ will be given by

$$\pi^*(s) = \arg\max_{a'} Q^*(s_t, a'_t \mid \theta^Q) \tag{27}$$

where $Q^*(s, a)$ is the optimal Q-value via DNN approximation. DQN will choose the approximated action $a_{t+1} = \pi^*(s_{t+1})$. Then the approximated $\tilde{Q}(s_t, a_t)$ can be formulated as

$$\tilde{Q}(s_t, a_t \mid \theta^Q) = G(t \mid s = s_t, a = a_t) + \gamma Q(s_{t+1}, \pi * (s_{t+1}) \mid \theta^Q) \tag{28}$$

Then θ^Q, which is the parameter of the DNN, is updated by minimizing the loss:

$$L = \frac{1}{T} \sum_{t=0}^{T} (\tilde{Q}(s_t, a_t \mid \theta^Q) - Q(s_t, a_t \mid \theta^Q))^2 \tag{29}$$

Specially, at time instant t, the agent will choose action a_t according to (27), get a reward r_t and turn to next state s_{t+1}. Then vector (s_t, a_t, r_t, s_{t+1}) will be stored into experience memory which can speed the learning procedure [11]. Then

Algorithm 1 DQN based dynamic resource allocation algorithm

```
• Initialize replay memory $\mathcal{D}$ to capacity $N$
• Initialize parameter of the DNN $\theta^Q$ with random weights.
• for episode=1,M do
    – Initialize the UDN scenario, receive initial observation state $s_1$
    – for t=1, T, do
        ∗ with probability $\varepsilon$ select a random action $a_t$
        ∗ otherwise $a_t = \pi(s_t)$
        ∗ do $a_t$ , observe $r_t$ and $s_{t+1}$
        ∗ Store transition $(s_{t,}a_t, r_t, s_{t+1})$ in $\mathcal{D}$
        ∗ if the replay memory $\mathcal{D}$ is full, do
            · Sample a random batch of $K$ transitions $(s_{i,}a_i, r_i, s_{i+1})$ from $\mathcal{D}$
            · Set $\tilde{Q}(s_i, a_i \mid \theta^Q) = r_t + \gamma \max_{a'} Q'(s_{t+1}, a'_{t+1} \mid \theta^Q)$
            · Update the parameter of DNN $\theta^Q$ by minimizing the loss: $L = \frac{1}{N}\sum_i(\tilde{Q}(s_i, a_i \mid \theta^Q) - Q(s_i, a_i \mid \theta^Q)^2$
            · Update the policy $\pi(s_t) = \arg\max_{a'} Q(s_t, a'_t \mid \theta^Q)$
    – end for
• end for
```

the parameter θ^Q and policy π will be updated according to the random batch of K data fetched from experience memory. Note that there are some tricks that can guarantee the performance, in this paper we just simply show the process. More details are shown in [11].

We conclude the DQN-based dynamic resource allocation problem into Algorithm 1.

IV. Simulation Results and Analysis

Simulation settings and results of the proposed DQN-based dynamic resource allocation framework are presented in this section. The DNN adopted in the model contains two fully connected hidden layers. The first layer contains 64 neurons and second contains 32 neurons. The activation function of the input layer, the first hidden layer and the second hidden layer are tanh. The activation function of the output layer is sigmoid. We implemented it using Tensorflow 1.0. The common settings we used are summarized in Table I.

Table I
SIMULATION SETTINGS

Parameter	Value
Channel bandwidth B	20MHz
maximum associated number $\lvert I_{max} \rvert$	20
Size of replay memory $\mathcal{D}$	10000
Battery Capacity	$100J$
Data Requests	$100bits$
Transceiverower P^{tx}	3W
Operation power P^{op}	10W
Switching power P_c	1W
Background noise σ^2	-174dBm/Hz
Path Loss α	2
Distance	Uniformly distributed in [0,50]m
Time interval	1s
Batch size K	200
Episode number	200

We compared the proposed DQN-based framework with a classic Q-learning framework using in [10]. First, we compare the performance of the proposed framework in the scenario that the number of SBS is 5. In each time instant, users come according to a poisson distribution $\Gamma(\lambda(t))$ where $\lambda(t) = sin(t) + 1$ and the locations of the users obey uniform distribution in the serving area. Also, users leave according to a negative exponential distribution $exp(1)$. Meanwhile, the energy arrival is discretized and also satisfies a possion distribution where $\lambda(t) = cos(t) + 1$. Each discretized energy stands for 1 Joule. Specially, if energy harvested by SBS j at time instant t is 2, $2J$ will be stored into the battery of SBS j . Note that though we assume energy arrival and user arrival both satisfy specific time-related distributions for simplicity, the DQN-based framework actually gets no prior knowledge about energy arrival and so on. Since the scenario contains lots of instability, we set 200 time instants for one episode and the total EE and delay will be recorded and sumed to reduce the instability. The corresponding results are presented in Fig.2. From this figure we can make the following observations. Note that for simplicity, we call the two methods, Q-learning and DQN respectively in the following.

1) DQN performs worse than Q-learning in the first 50 episodes. In fact, in the first 50 episodes DQN just chooses actions randomly and stores the feedbacks into replay memory. After 50 episodes, DQN starts to learn from the experience. We can see that the performance of DQN is unstable but the instability is reducing as episodes go on. After 100 episodes, DQN tends to be stable and performs better then Q-learning for over 20% of performance.
2) Q-learning can learn from feedbacks in the beginning, so it performs better than DQN in the first 50 episodes. As we can see, the performance of Q-learning continues to increase until the 50th episode and then performs quite stable. As we can see, in this scenario, Q-learning achieves convergence as fast as DQN but performs worse then DQN.

As Q-learning can converge quickly, Q-learning can be used at the beginning. Simultaneously, DQN can be pretrained according to the data from Q-learning or other algorithms until the performance of DQN is better than Q-learning. Afterwards, DQN can be used instead of Q-learning.

Then we compare the performances of the proposed framework in scenarios of different SBS numbers as shown in Fig.3 and Fig.4. In this simulation, the total number of episodes will be set as 100. From the figures we can note that the EE and delay performances all decrease and as the number of SBSs increases, the performance gaps between DQN and Q-learning become larger. Such phenomenons can be explained as the density of SBSs is higher than what is necessary. Therefore, as

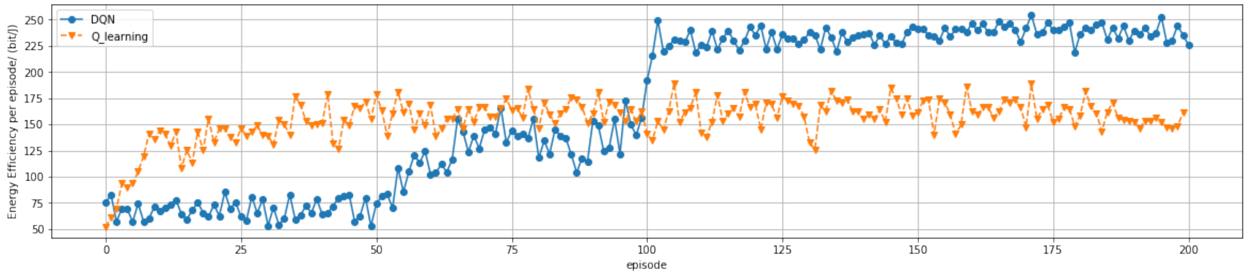

Figure 2. the optimization process in 5 SBSs scenario

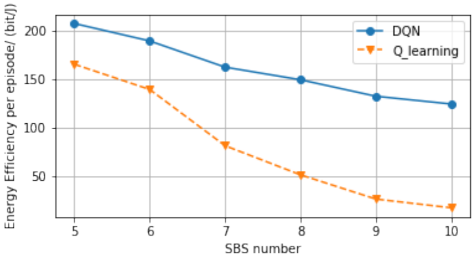

Figure 3. The total energy efficiency VS. the SBS numbers

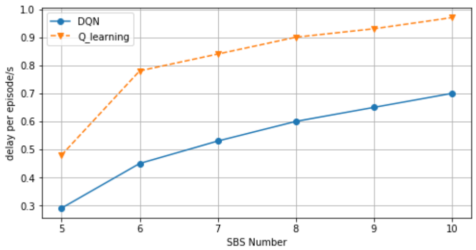

Figure 4. The total delay VS. the SBS numbers

the SBS number increases, the power consumption increases while the number of serving users are not changed. Consequently, the EE of the EH-UDN is decreasing. On the other hand, such dense deployment leads to severe interference, so the total delay will increase. As the scenarios become more complex, Q-learning can no longer perform as good as DQN and show degrading in performances.

V. CONCLUSIONS

In this paper, a novel DQN-based framework is presented for EE in EH-UDN. Specially, we formally formulate the SBS ON/OFF problem into a dynamic and real-time optimization problem. The problem is NP-hard and the prior knowledge about the energy arrival, user arrival and CSI are bypassed. Then the action space, state space and reward function are carefully defined so as to transfer the optimization problem into the DQN-based framework. We have evaluated the proposed DQN-based framework by comparing it with a well-used RL framework called Q-learning via simulation. Simulation results show that the proposed DQN-based framework can enhance EE significantly.

REFERENCES

[1] I. Hwang, B. Song, and S. S. Soliman, “A holistic view on hyper-dense heterogeneous and small cell networks,” *IEEE Communications Magazine*, vol. 51, no. 6, pp. 20–27, June 2013.

[2] Y. Mao, Y. Luo, J. Zhang, and K. B. Letaief, “Energy harvesting small cell networks: feasibility, deployment, and operation,” *IEEE Communications Magazine*, vol. 53, no. 6, pp. 94–101, June 2015.

[3] T. Han and N. Ansari, “Powering mobile networks with green energy,” *IEEE Wireless Communications*, vol. 21, no. 1, pp. 90–96, February 2014.

[4] O. Blume, H. Eckhardt, S. Klein, E. Kuehn, and W. M. Wajda, “Energy savings in mobile networks based on adaptation to traffic statistics,” *Bell Labs Technical Journal*, vol. 15, no. 2, pp. 77–94, Sept 2010.

[5] S. Zhang, N. Zhang, S. Zhou, J. Gong, Z. Niu, and X. Shen, “Energy-aware traffic offloading for green heterogeneous networks,” *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 5, pp. 1116–1129, May 2016.

[6] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[7] H. Zhang, S. Huang, C. Jiang, K. Long, V. C. Leung, and H. V. Poor, “Energy efficient user association and power allocation in millimeter-wave-based ultra dense networks with energy harvesting base stations,” *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 9, pp. 1936–1947, 2017.

[8] G. Lee, W. Saad, M. Bennis, A. Mehbodniya, and F. Adachi, “Online ski rental for scheduling self-powered, energy harvesting small base stations,” in *2016 IEEE International Conference on Communications (ICC)*. IEEE, 2016, pp. 1–6.

[9] R. S. Sutton and A. G. Barto, “Reinforcement learning: An introduction,” *IEEE Transactions on Neural Networks*, vol. 9, no. 5, pp. 1054–1054, 1998.

[10] M. Miozzo, L. Giupponi, M. Rossi, and P. Dini, “Distributed q-learning for energy harvesting heterogeneous networks,” in *2015 IEEE International Conference on Communication Workshop (ICCW)*, June 2015, pp. 2006–2011.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing atari with deep reinforcement learning,” *arXiv preprint arXiv:1312.5602*, 2013.

[12] F. Han, Z. Safar, and K. J. R. Liu, “Energy-efficient base-station cooperative operation with guaranteed qos,” *IEEE Transactions on Communications*, vol. 61, no. 8, pp. 3505–3517, August 2013.

[13] E. Ghadimi, F. D. Calabrese, G. Peters, and P. Soldati, “A reinforcement learning approach to power control and rate adaptation in cellular networks,” in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–7.