Hindawi
Wireless Power Transfer
Volume 2021, Article ID 5513509, 12 pages
https://doi.org/10.1155/2021/5513509

Research Article

Optimal Wireless Information and Power Transfer Using Deep Q-Network

Yuan Xing,[1] **Haowen Pan**,[2] **Bin Xu**,[2] **Cristiano Tapparello**,[3] **Wei Shi**,[1] **Xuejun Liu**,[1] **Tianchi Zhao**,[4] **and Timothy Lu**[1]

[1]*Department of Engineering & Technology, University of Wisconsin-Stout, Menomonie, WI, USA*
[2]*Shanghai Legit Network Technology Co, Shanghai, China*
[3]*Department of Electrical and Computer Engineering, University of Rochester, Rochester, NY 14627, USA*
[4]*Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85721, USA*

Correspondence should be addressed to Yuan Xing; xingy@uwstout.edu

Received 3 February 2021; Revised 2 April 2021; Accepted 29 April 2021; Published 21 May 2021

Academic Editor: Arpan Desai

In this paper, a multiantenna wireless transmitter communicates with an information receiver while radiating RF energy to surrounding energy harvesters. The channel between the transceivers is known to the transmitter, but the channels between the transmitter and the energy harvesters are unknown to the transmitter. By designing its transmit covariance matrix, the transmitter fully charges the energy buffers of all energy harvesters in the shortest amount of time while maintaining the target information rate toward the receiver. At the beginning of each time slot, the transmitter determines the particular beam pattern to transmit with. Throughout the whole charging process, the transmitter does not estimate the energy harvesting channel vectors. Due to the high complexity of the system, we propose a novel deep Q-network algorithm to determine the optimal transmission strategy for complex systems. Simulation results show that deep Q-network is superior to the existing algorithms in terms of the time consumption to fulfill the wireless charging process.

1. Introduction

For a wireless transceiver pair with multiple antennas, optimizing the transmit covariance matrix can achieve high data-rate communication over the multiple-input multiple-output (MIMO) channel. Meanwhile, the radiated radio frequency (RF) energy can be acquired by the nearby RF energy harvesters to charge the electronic devices [1].

The problem of simultaneous wireless information and power transfer (SWIPT) has been widely discussed in recent years. SWIPT systems are divided into two categories: (1) the receiver splits the received signals for information decoding and energy harvesting [2, 3]; (2) separated and dedicated information decoders (ID) and RF energy harvesters (EH) exist in the systems [4]. For the second type of the system, different transmission strategies have ever been proposed to achieve good performance points in the rate-energy region [1, 2, 5]. For the multiple RF energy harvesters, which are in the vicinity of the wireless transmitter, the covariance matrix at the transmitter is designed to either maximize the net energy harvesting rate or fairly distribute the radiated RF energy at the harvesters [6, 7]. The achievable information rate of the wireless transmitter-receiver pair is beyond a minimum requirement for reliable communication. Most of the existing works assume the channel state information (CSI) is completely known. Given the complete CSI, the transmitter designs the transmit covariance matrix to achieve the maximum information rate while satisfying the RF energy harvesting requirement [4, 8].

However, in practice, it is difficult for the transmitter to obtain the channel state information to the nearby RF energy harvesters because the scattering distribution of the hardware-limited energy harvesters makes the channel estimation at the RF energy harvesters challenging [9, 10]. The analytic center cutting plane method (ACCPM) was proposed for the transmitter to approximate the channel

information with a few bits of feedback from the RF energy receiver iteratively [10]. Since this method is implemented by solving a convex optimization problem, the algorithm leads to high computational complexity. To reduce complexity, channel estimation based on Kalman filtering was proposed [11]. Nevertheless, the disadvantage of this approach is the slow convergence rate. In order to effectively deal with the CSI acquisition problem, in our paper, we will use the deep learning algorithm to solve the optimization problem in the SWIPT system only with partial channel information. The partial CSI is easy to acquire, which is already enough to achieve superior system performance using the deep Q-network. To the best of our knowledge, we are the first one to use the deep Q-network to optimize the SWIPT system performance and validate its superiority.

In our model, the transmitter intends to fully charge all surrounded energy harvesters' energy buffers in the shortest time while maintaining a target information rate toward the receiver. The communication link is defined as a strong line-of-sight (LOS) transmission, which is supposed to be invariant, but the energy harvesting channel conditions vary over time. Due to current hardware limitations, we assume that the estimation of the energy harvesting channel vectors is not able to be implemented under the fast varying channel conditions. As a result, the wireless charging problem can be modeled as a high complexity discrete-time stochastic control process with unknown system dynamics [12]. In [13], a similar problem has been explored. A multiarmed bandit algorithm is used to determine the optimal transmission strategy. In our paper, we apply a deep Q-network to solve the optimization problem and the simulation results demonstrate that the deep Q-network algorithm outperforms the multiarmed bandit algorithm. Historically, deep Q-network has a strongly proven record of attaining mastery over complex games with a very large number of system states, and unknown state transition probabilities [12]. More recently, a deep Q-network has been applied to deal with complex communication problems and has been shown to achieve good performance [14–16]. For this reason, we found deep Q-network fitting for our model. In our model, we consider the accumulated energy of the energy harvesters as the system states, while we define the action as the transmit power allocation. At the beginning of each time slot, each energy harvester sends feedback about the accumulated energy level to the wireless transmitter, and the transmitter collects all the information in order to generate the system state and inputs it into a well-trained deep Q-network. The deep Q-network outputs the Q values corresponding to all possible actions. The action with the maximum Q value is selected as the beam pattern to be used for the transmission during the current time slot.

Based on the traditional deep Q-network, the double deep Q-network and dueling deep Q-network algorithms are applied in order to reduce the observed overestimations [17] and improve the learning efficiency [18]. Henceforth, we apply dueling double deep Q-network to solve the varying channel multiple energy harvester wireless charging problem.

The novelties of this paper are summarized as follows:

(i) The simultaneous wireless information and power transfer problem is formulated as a Markov decision process (MDP) in an unknown varying channel condition for the first time.

(ii) The deep Q-network algorithm is applied to solve the proposed optimization problem for the first time. We demonstrate that, compared to the other existing algorithms, deep Q-network shows the superiority in efficient and stable wireless power transfer.

(iii) Multiple experimental scenarios are explored. By varying the number of transmission antennas and the number of energy harvesters in the system, the performance of both the deep Q-network and the other algorithms is compared and analyzed.

(iv) The evaluation for the algorithms is based on the real experimental data, which validate the effectiveness of the proposed deep Q-network in real-time simultaneous wireless information and power transfer systems.

The rest of the paper is organized as follows. In Section 2, we describe the simultaneous wireless information and power transfer system model. In Section 3, we model the optimization problem as a Markov decision process and present a deep Q-network algorithm to determine the optimal transmission strategy. In Section 4, we present our simulation results for different experimental environments. Section 5 concludes the paper.

2. System Model

As shown in Figure 1, an information transmitter communicates with its receiver while perceived by K nearby RF energy harvesters [8]. Both the transmitter and the receiver are equipped with M antennas, while each RF energy harvester is equipped with one receive antenna. The baseband received signal at the receiver can be represented as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{z}, \tag{1}$$

where $\mathbf{H} \in \mathbb{C}^{M\times M}$ denotes the normalized baseband equivalent channel from the information transmitter to its receiver, $\mathbf{x} \in \mathbb{C}^{M\times 1}$ represents the transmitted signal, and $\mathbf{z} \in \mathbb{C}^{M\times 1}$ is the zero-mean circularly symmetric complex Gaussian noise with $\mathbf{z} \sim \mathscr{CN}(\mathbf{0}, \rho^2\mathbf{I})$.

The transmit covariance matrix is denoted by $\mathbf{Q}$, i.e., $\mathbf{Q} = \mathrm{E}[\mathbf{x}\mathbf{x}^H]$. The covariance matrix is Hermitian positive semidefinite, i.e., $\mathbf{Q} \succcurlyeq 0$. The transmit power is restricted by the transmitter's power constraint P, i.e., $\mathrm{Tr}(\mathbf{Q}) \leq P$. For the information transmission, we assume that a Gaussian codebook with infinitely many code words is used for the symbols and the expectation of the transmit covariance matrix is taken over the entire codebook. Therefore, $\mathbf{x}$ is the zero-mean circularly symmetric complex Gaussian with $\mathbf{x} \sim \mathscr{CN}(\mathbf{0}, \mathbf{Q})$. With transmitter precoding and receiver filtering, the capacity of the MIMO channel is the sum of the capacities of the parallel noninterfering single-input single-output (SISO) channels (eigenmodes of channel $\mathbf{H}$) [19]. We

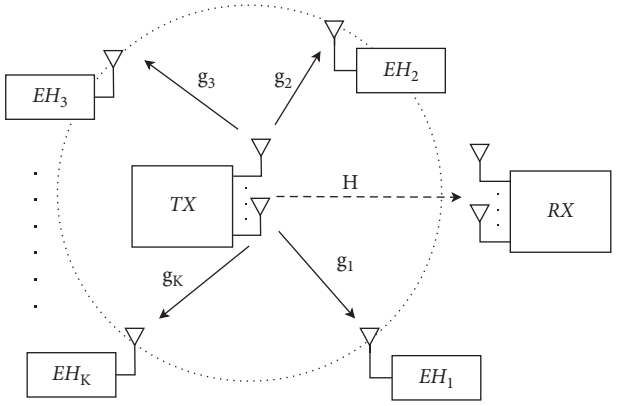

Figure 1: Wireless information transmitter and receiver surrounded by multiple RF energy harvesters.

convert the MIMO channel to M eigenchannels for information and energy transfer [20, 21]. A singular value decomposition (SVD) on $\mathbf{H}$ gives $\mathbf{H} = \mathbf{U}\Sigma\mathbf{V}^H$, where $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_M)$ contains the M singular values of $\mathbf{H}$. Since the MIMO channel is decomposed into M parallel SISO channels, the information rate can be given by

$$r = \sum_{m=1}^{M} \log\left(1 + \rho^{-2}\left|\sigma_m\right|^2 \widehat{q}_m\right), \tag{2}$$

where $\{\widehat{q}_m\}$ are the diagonal elements of $\widehat{\mathbf{Q}}$ with $\widehat{\mathbf{Q}} = \mathbf{V}^H\mathbf{Q}\mathbf{V}$.

The RF energy harvester received power specifies the harvested energy normalized by the baseband symbol period and scaled by the energy conversion efficiency. The received power at the ith energy harvester is

$$p_i = \mathbf{g}_i^H \mathbf{Q} \mathbf{g}_i, \tag{3}$$

where $\mathbf{g}_i \in \mathbb{C}^{M\times 1}$ is the channel vector from the transmitter to the ith energy harvester. With MIMO channel decomposition, the received power at energy harvester i is denoted as

$$p_i = \sum_{m=1}^{M} \left|\widehat{g}_{im}\right|^2 \widehat{q}_m, \tag{4}$$

where $\{\widehat{g}_{im}\}$ are the elements of vector $\widehat{\mathbf{g}}_i$ with $\widehat{\mathbf{g}}_i = \mathbf{V}^H\mathbf{g}_i$.

We define the simplified channel vector from the transmitter to the ith RF energy harvester as

$$\mathbf{c}_i = \left[\left|\widehat{g}_{i1}\right|^2, \left|\widehat{g}_{i2}\right|^2, \ldots, \left|\widehat{g}_{iM}\right|^2\right]^T, \tag{5}$$

for each $i \in \mathcal{K} = \{1, 2, \ldots, K\}$. The simplified channel vector contains no phase information. The K simplified channel vectors compose matrix $\mathbf{C} \in \mathbb{R}^{M\times K}$ as

$$\mathbf{C} = \left[\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K\right]. \tag{6}$$

In what follows, we assume that time is slotted, each time slot as a duration T, and that each energy harvester is equipped with an energy buffer of size $B_i \in [0, B_{\max}]$, $i \in \mathcal{K}$. Without loss of generality, we assume that, at $t = 0$, all harvesters' buffers are empty, which corresponds to system state $s_0 = [0, 0, \ldots, 0]$. At a generic time slot t, the transmitter transmits with one of the designed beam patterns. Each harvester i can harvest the specific amount of power p_i, and its energy buffer values increase to $B_i^{t+1} = B_i^t + p_i T$. Therefore, each state of the system includes the accumulated harvested energy information of all K harvesters, i.e.,

$$s_t = \left[B_1^t, B_2^t, \ldots, B_K^t\right], \tag{7}$$

where B_i^t denotes the ith energy harvester's accumulated energy up to time slot t.

Once all harvesters are fully charged, we assume that the system arrives at a final goal state denoted as $s_G = [B_{\max}, B_{\max}, \ldots, B_{\max}]$. We note that the energy buffer level $B_{\max}$ also accounts for situations in which $B_i > B_{\max}$.

3. Problem Formulation for Time-Varying Channel Conditions

In this section, we suppose that the communication link is characterized by strong LOS transmission, which results in an invariant channel matrix $\mathbf{H}$, while the energy harvesting channel vector $\mathbf{g}$ varies over time slots. We model the wireless charging problem as a Markov decision process (MDP) and show how to solve the optimization problem using reinforcement learning (RL). When the number of system states is very large, we apply a deep Q-network algorithm to acquire the optimal strategy at each particular system state.

3.1. Problem Formulation. In order to model our optimization problem as a RL problem, we define the beam pattern chosen in a particular time slot t as the action $\mathbf{a}^t$. The set of possible actions $\mathcal{A}$ is determined by equally generating L different beam patterns with power allocation vector $\widehat{\mathbf{q}} = [\widehat{q}_1, \ldots, \widehat{q}_m]$ that satisfies the power and information rate constraints, i.e., $\sum_{i=1}^{M} \widehat{q}_m = P$, $\sum_{i=1}^{M} \log(1 + \rho^{-2}|\sigma_m|^2 \widehat{q}_m) \geq R$. Each beam pattern corresponds to a particular power level p_i, which depends not only on the action $\mathbf{a}^t$ but also on the channel condition experienced by the harvester during time slot t.

Given the above, the simultaneous wireless information and energy transfer problem for a time-varying channel can be formulated as minimizing the time-consumption n to fully charge all the energy harvesters while maintaining the information rate between the information transceivers:

$$\mathscr{P}_1: \begin{aligned} &\underset{\{\mathbf{a}^t\}}{\text{minimize}} && n \\ &\text{subject to} && a_m^t \geq 0 \\ &&& \sum_{m=1}^{M} a_m^t \leq P \\ &&& \sum_{m=1}^{M} \log\left(1 + \rho^{-2}\left|\sigma_m\right|^2 a_m^t\right) \geq R \\ &&& \sum_{t=1}^{n}\sum_{m=1}^{M} \left|\widehat{g}_{im}^t\right|^2 a_m^t T \geq B_{\max}, \quad \forall i \in \mathcal{K} \end{aligned} \tag{8}$$

In general, the action selected at each time slot will be different to adapt to the current channel conditions and

current energy buffer state of the harvesters. Therefore, the evolution of our system can be described by a Markov chain, where the generic state s is identified by the current buffer levels of the harvester, i.e., $s = \{B_1, B_2, \ldots, B_K\}$. The set of all states is denoted by $\mathcal{S}$. Among all states, we are interested in the state in which all harvesters' buffer is empty, namely, $s_0 = \{0, \ldots, 0\}$, and the state s_G in which all the harvesters are fully charged, i.e., $s_G = \{B_{\max}, \ldots, B_{\max}\}$. If we suppose that we know all the channel coefficients at each time slot, problem $\mathcal{P}_1$ can be seen as a stochastic shortest path (SSP) problem from state s_0 to state s_G. At each time slot, the system is in a generic state s, the transmitter selects a beam pattern or action $\mathbf{a} \in \mathcal{A}$, and the system moves to a new state s'. The dynamics of the system is captured by transition probabilities $p_{s,s'}(\mathbf{a})$, $s, s' \in \mathcal{S}$, and $\mathbf{a} \in \mathcal{A}$, describing the probability that the harvesters' energy buffers reach the levels in s' after a transmission with beam pattern $\mathbf{a}$. We note that the goal state s_G is absorbing, i.e., $P_{s_G,s_G}(\mathbf{a}) = 1$, $\forall \mathbf{a} \in \mathcal{A}$.

Each transition also has an associated reward, $w(s, \mathbf{a}, s')$, that denotes the reward when the current state is $s \in \mathcal{S}$, action $\mathbf{a} \in \mathcal{A}$ is selected, and the system moves to state $s' \in \mathcal{S}$. Since we aim at reaching s_G in the fewest transmission time slots, we consider that the action entails a positive reward related to the difference between the current energy buffer level and the full energy buffer level of all harvesters. When the system reaches state s_G, we set the reward as 0. In this way, the system not only tries to fully charge all harvesters in the shortest time but will also uniformly charge all the harvesters. In detail, we define the reward function as

$$w(s, \mathbf{a}, s') = -\lambda\left(\mathrm{KB}_{\max} - \sum_{i=1}^{K} \min\left(s_i', B_{\max}\right)\right), \tag{9}$$

where

$$\lambda s_i' = \lambda s_i + \lambda \sum_{m=1}^{M} \left|\widehat{g}_{im}\right|^2 a_m T, \tag{10}$$

and λ denotes the unit price of the harvested energy.

It is noted that different reward functions can also be selected. As an example, it is also possible to set a constant negative reward (e.g., a unitary cost) for each transmission that the system does not reach the goal state and a big positive reward only for the states and actions that bring the system to the goal state s_G. This can be expressed as follows:

$$w(s, \mathbf{a}, s') = \begin{cases} +\infty, & s' = s_G, \\ -1, & \text{otherwise.} \end{cases} \tag{11}$$

We note that the reward formulation of equation (11) is actually equivalent to minimizing the number of time slots required to reach state s_G starting from state s_0.

Using the above formulation, the optimization problem $\mathcal{P} = (\mathcal{S}, \mathcal{A}, p, w, s_0, s_G)$ can then be seen as a stochastic shortest path search from state s_0 to state s_G on the Markov chain with states $\mathcal{S}$ and probabilities $\{p_{s,s'}(\mathbf{a})\}$, actions $\mathbf{a} \in \mathcal{A}$, and rewards $w(s, \mathbf{a}, s')$. Our objective is to find, for each possible state $s \in \mathcal{S}$, an optimal action $\mathbf{a}^*(s)$ so that the system will reach the goal state following the path with maximum average reward. A generic policy can be written as $\pi = \{\mathbf{a}(s): s \in \mathcal{S}\}$.

Different techniques can be applied to solve problem $\mathcal{P}_1$, as it represents a particular class of MDPs. In this paper, however, we assume that the channel conditions at each time slot are unknown, which corresponds to not knowing the transition probabilities $\{p_{s,s'}(\mathbf{a})\}$. Therefore, in the next section, we describe how to solve the above problem using reinforcement learning.

3.2. Optimal Power Allocation with Reinforcement Learning. Reinforcement learning is suitable for solving optimization problems in which the system dynamics follow a particular transition probability function, however, the probabilities $\{p_{s,s'}(\mathbf{a})\}$ are unknown. In what follows, we first show how to apply the Q-learning algorithm [22] to solve the optimization problem and then show how we can combine the reinforcement learning approach with a neural network to approximate the system model in case of large states and action sets, using deep Q-network [12].

3.2.1. Q-Learning Method. If the number of system states is small, we can depend on the traditional Q-learning method to find the optimal strategy at each system state, as defined in the previous section.

To this end, we define the cost function of action $\mathbf{a}$ on system state s as $\{p_{s,s'}(\mathbf{a})\}$, with $s \in \mathcal{S}, \mathbf{a} \in \mathcal{A}$. The algorithm initializes with $Q(s, \mathbf{a}) = 0$ and then updates the Q values using the following equation:

$$Q(s, \mathbf{a}) = (1 - \alpha(s, \mathbf{a}))Q(s, \mathbf{a}) + \alpha(s, \mathbf{a})[w(s, \mathbf{a}, s') + \gamma f(s', \mathbf{a})], \tag{12}$$

where

$$f(s', \mathbf{a}) = \min_{\mathbf{a} \in \mathcal{A}} Q(s', \mathbf{a}), \tag{13}$$

and $\alpha(s', \mathbf{a})$ denotes the learning rate. In each time slot, only one Q value is updated, and hence, all the other Q values remain the same.

At the beginning of the learning iterations, since the Q-table does not have enough information to choose the best action at each system state, the algorithm randomly explores new actions. Hence, we first define threshold $\varepsilon_c \in [0.5, 1]$, and we then randomly generate a probability $p \in [0, 1]$. In the case that $p \geq \varepsilon_c$, we choose the action $\mathbf{a}$ as

$$\mathbf{a} = \max_{\mathbf{a} \in \mathcal{A}} Q(s, \mathbf{a}). \tag{14}$$

On the contrary, if $p < \varepsilon_c$, we randomly select one action from the action set $\mathcal{A}$.

When Q^* converges, the optimal strategy at each state is determined as

$$\pi^*(s) = \arg\max_{\mathbf{a}\in\mathscr{A}} Q^*(s,\mathbf{a}), \tag{15}$$

which corresponds to finding the optimal beam pattern for each system state during the charging process.

3.2.2. Deep Q-Network. When considering a complex system with multiple harvesters, large energy buffers, and time-varying channel conditions, the number of system states dramatically increases. In order to learn the optimal transmit strategy at each system state, the Q-learning algorithm described before requires a Q-table with a large number of elements, making it very difficult for all the values in the Q-table to converge. Therefore, in what follows, we describe how to apply the deep Q-network (DQN) approach to find the optimal transmission policy.

The main idea of DQN is to train a neural network to find the Q function of a particular system state and action combination. When the system is in state s, and action $\mathbf{a}$ is selected, the Q function is denoted as $Q(s,\mathbf{a},\theta)$. θ denotes the parameters of the Q-network. The purpose of training the neural network is to make

$$Q(s,\mathbf{a},\theta) \approx Q^*(s,\mathbf{a}). \tag{16}$$

According to the DQN algorithm [17], two neural networks are used to solve the problem: the evaluation network and the target network, which are denoted as eval_net and target_net, respectively. Both the eval_net and the target_net are set up with several hidden layers. The input of the eval_net and the target_net are denoted as s and s', which describe the current system state s and the next system state s', respectively. The output of eval_net and target_net are denoted as $Q_e(s,\mathbf{a},\theta)$ and $Q_t(s,\mathbf{a},\theta)$, respectively. The evaluation network is continuously trained to update the value of θ; however, the target network only copies the weight parameters from the evaluation network intermittently (i.e., $\theta' = \theta$). In each neural network learning epoch, the loss function is defined as

$$\text{loss}(\theta) = E\left[\left(y - Q_e(s,\mathbf{a},\theta)\right)^2\right]. \tag{17}$$

where y represents the real Q value and is calculated as

$$y = w(s,\mathbf{a},s') + \varepsilon \max_{a'\in\mathscr{A}} Q_t(s',\mathbf{a}',\theta'), \tag{18}$$

where ε is the learning rate. As the loss function updates, the values are backpropagated to the neural network to update the weight of the eval_net.

In order to better train the neural network, we apply the experience reply method to remove the correlation between different training data. Each experience consists of the current system state s, the action $\mathbf{a}$, the next system state s', and the corresponding reward $w(s,\mathbf{a},s')$. The experience is denoted by the set $ep = \{s,\mathbf{a},w(s,\mathbf{a},s'),s'\}$. The algorithm records D experiences, and randomly select D_s (with $D_s < D$) experiences from D for training. After the training is finished, target_net clones all the weight parameters from the eval_net (i.e., $\theta' = \theta$).

The algorithm used for the DQN training process is presented in Algorithm 1. In the algorithm, we define in each training iteration, we generate D usable experiences ep and select D_s of all for training the eval_net. In total, we suppose there are U training iterations. We consider that, for both the eval_net and the target_net, there are N_l layers in the neural network. In the learning process, we use $\mathbf{C}$ to denote all energy harvesters' channel condition in a particular time slot.

3.2.3. Dueling Double Deep Q-Network. Since more harvesters and time-varying channel conditions incur more system states, even if we utilize the original DQN, it is hard to study the transmit rules for the transmitter. Therefore, we can apply dueling double DQN in order to deal with the overestimating problem during the training process and improve the learning efficiency of the neural network. Doubling DQN is a technique that strengthens the traditional DQN algorithm by preventing overestimating to happen [17]. In traditional DQN, as shown in equation (18), we utilize the target_net to predict the maximum Q value of the next state. However, the target_net is not updated at every training episode, which may lead to an increase in the training error and therefore complicate the training process. In doubling DQN, we utilize both the target_net and the eval_net to predict the Q value. The eval_net is used to determine the optimal action to be taken for the system state s' as follows:

$$y = w(s,\mathbf{a},s') + \varepsilon \max_{a'\in\mathscr{A}} Q_e\left(s', \arg\max_{\mathbf{a}\in\mathscr{A}} Q(s',\mathbf{a},\theta),\theta'\right). \tag{19}$$

It can be shown that, following this approach, the training error considerably decreases [17].

In traditional DQN, the neural network only has the Q value as the output. In order to speed up the convergence, we apply dueling DQN by setting up two output streams from the neural network. The first stream is represented by the output value $V(\mathbf{s},\theta,\beta)$ results of the neural network, which represents the Q value of each system state. The second stream is called advantage output $A(s',\mathbf{a},\theta,\alpha)$ and describes the advantage of applying each particular action to the current system state [18]. α and β are parameters that relate the two streams and the neural network output, which is denoted as

$$Q(s,\mathbf{a},\theta,\alpha,\beta) = V(s,\theta,\beta) + \left(A(s',\mathbf{a},\theta,\alpha) - \frac{1}{|A|}\sum_{a'} A(s',\mathbf{a},\theta,\alpha)\right). \tag{20}$$

Dueling DQN can efficiently eliminate the extra training freedom, which speeds up the training [18].

4. Simulation Results

We simulate a MIMO wireless communication system with nearby RF energy harvesters. The wireless transmitter has at

(1) Randomly generate the weight parameter θ for the eval_net. The target_net clones the weight parameters $\theta' = \theta$. $u = 1$. $s = s_0$. $\mathbf{C} = \mathbf{C}_t$. $t = 1$. $D = d = 1$.
(2) At the beginning of the time slot, randomly generate a probability $p \in [0, 1]$.
IF $D > 200$ and $p \geq \varepsilon_{ch}$:
we choose the action $\mathbf{a}$ as $\mathbf{a} = \max_{\mathbf{a} \in \mathscr{A}} Q(s, \mathbf{a})$
ELSEIF $p < \varepsilon_{ch}$:
Randomly choose the action from action set $\mathscr{A}$.
The transmitter transmits with the selected beam pattern.
(3) Throughout the whole time slot, the RF energy is accumulated in the harvesters' energy buffer, as $s_i' = s_i + \sum_{m=1}^{M} |\widehat{g}_{im}^t|^2 a_m T, \forall i \in \mathscr{K}$. At the end of each time slot, each harvester feedbacks the energy level to the transmitter and the system state is updated to s'.
(4) $ep(d) = \{s, \mathbf{a}, w(s, \mathbf{a}, s'), s'\}$. $d = d + 1$. If D reaches the maximum of experience pool, D remains constant, $d = 1$, otherwise, $D = d$. $s = s'$. $t = t + 1$. $\mathbf{C} = \mathbf{C}_t$.
(5) After experience pool accumulates enough data, from D experiences, randomly select D_s experiences to train the neural network eval_net. Backpropagation method is applied to minimize the loss function loss(θ). Clone the weight parameters from eval_net to target_net after several time intervals.
(6) **IF** $s' = s_G$:
$s = s_0$. $t = 1$. $\mathbf{C} = \mathbf{C}_t$. $u = u + 1$. If $u = U$, algorithm terminates; otherwise, go back to step 2.
IF $s' \neq s_G$:
go to step 3.

Algorithm 1: Deep Q-network algorithm training process.

most $M = 4$ antennas. The 4×4 communication MIMO channel matrix $\mathbf{H}$ is measured by two Wireless Open-Access Research Platform (WARP) v3 boards. Both WARP boards are mounted with the FMC-RF-2X245 dual-radio module, which is operated in 5.805 GHz frequency band. The Xilinx Virtex-6 FPGA operates as the central processing system and the WARPLab is used for rapid physical layer prototyping which is compiled by MATLAB [23]. We deploy two transceivers as line-of-sight transmission. The maximum transmitted power is $P = 12$W. $\rho^2 = -70$ dBm. The information rate requirement R is 53 bps/Hz. The average channel gain from the transmitter to the energy harvester is -30 dB. The energy conversion efficiency is 0.1. The duration of one time slot is defined as $T = 100$ ms.

DQN is trained to solve for the optimal transmit strategies for each system state. The simulation parameters used for DQN are presented in Table 1.

As described in Section 3.2, the exploration rate ε_c determines the probability that the network selects an action randomly or follows the values of the Q-table. Initially, we set $\varepsilon_c = 1$ because the experience pool has to accumulate reasonable amount of data to train the neural network. $\varepsilon_c = 1$ decreases with 0.001 at each training interval and finally stops at $\varepsilon_{ch} = 0.1$, since the experience pool has collected enough training data.

Refer to [24]. The dueling double DQN is used in our paper, which is shown in Figure 2. The software environment for simulation is TensorFlow 0.12.1 with Python 3.6 in Jupyter Notebook 5.6.0.

For the energy harvesters' channel, to show an example of the performance achievable by the proposed algorithm, we consider the Rician channel fading model [25]. We suppose within each time slot t, the channel is invariant and varies in different time slots [26]. At the end of each time slot, the energy harvester feedbacks the current energy level back to the transmitter. For the Rician fading channel model, the total gain of the signal is denoted as $\mathbf{g} = \mathbf{g}^s + \mathbf{g}^d$, where $\mathbf{g}^s$ is the invariant LOS component and $\mathbf{g}^d$ denotes a zero-mean Gaussian diffuse component. The channel between the transmit antenna m and the energy harvester i can be denoted as $g_{im} = g_{im}^s + g_{im}^d$. The magnitude of the faded envelope can be modeled using the Rice factor K^r such that $K_{im}^r = \rho_{im}^2 / 2\sigma_{im}^2$, where ρ_{im}^2 denotes the average power of the main LOS component between the transmit antenna m and energy harvester i and σ_{im}^2 denotes the variance of the scatter component. We can derive the magnitude of the main LOS component as $|g_{im}^s| = \sqrt{2K_{im}^r}\sigma_{im}$ since $1/2E[(|g_{im}^d|)^2] = \sigma_{im}^2$. The mean and the variance of g_{im} are denoted as $\mu_{g_{im}} = g_{im}^s$ and $\sigma_{g_{im}}^2 = \sigma_{im}^2$, respectively. In polar coordinates, $g_{im} = r_{im} e^{j\theta_{im}}$.

First, we explore the optimal deep Q-network structure under fading channels. We suppose the number of antennas is $M = 3$ and the number of energy harvesters is $K = 2$. The channel between each antenna of the transmitter and each harvester is individually Rician distributed. The action set $\mathscr{A}$ contains 13 actions satisfying the information rate requirement: $[2, 2, 8]^T$, $[2, 4, 6]^T$, $[2, 6, 4]^T$, $[2, 8, 2]^T$, $[4, 2, 6]^T$, $[4, 4, 4]^T$, $[4, 6, 2]^T$, $[4, 8, 0]^T$, $[6, 2, 4]^T$, $[6, 4, 2]^T$, $[6, 6, 0]^T$, $[8, 2, 2]^T$, and $[8, 4, 0]^T$.

The LOS amplitude components of all channel links are defined as $r_{im} = 0.5$, with $i = 1, 2$ and $m = 1, 2, 3$. The LOS phase components of all channel links are defined as $\theta_{11} = \pi/4$, $\theta_{12} = \pi/2$, $\theta_{13} = -\pi/4$, $\theta_{21} = -\pi/2$, $\theta_{22} = 0$, and $\theta_{23} = 3\pi/4$. The standard deviation of the g_{im} amplitude and phase is denoted as σ_{im} and $1/\sqrt{2K_{im}^r}$, respectively. We suppose $\sigma_{im} = 0.05, \forall i, m$. Hence, $1/\sqrt{2K_{im}^r} = (r_{im}/\sigma_{im})^{-1} = 0.1, \forall i, m$.

Using the fading channel model above, in Figure 3, we show how the structure of the neural network together with the learning rate can affect the performance of the DQN, for a fixed number of training episodes (i.e., 40000). The performance of DQN is measured by the average number of

Table 1: DQN simulation parameters.

Dueling Deep Q-network	Value
Number of hidden layers (N_L)	4
Number of nodes of each hidden layer	100
Learning rate (ε)	≤0.1
Mini-batch size	10
Learning frequency	5
Training starting step	200
Experience pool	≥20000
Initial exploration rate (ε_c)	1
Final exploration rate (ε_c)	0.1
Exploration interval	0.001
target_net weight replacement interval	≥100
Discount factor	0.9
Training episodes	≥40000

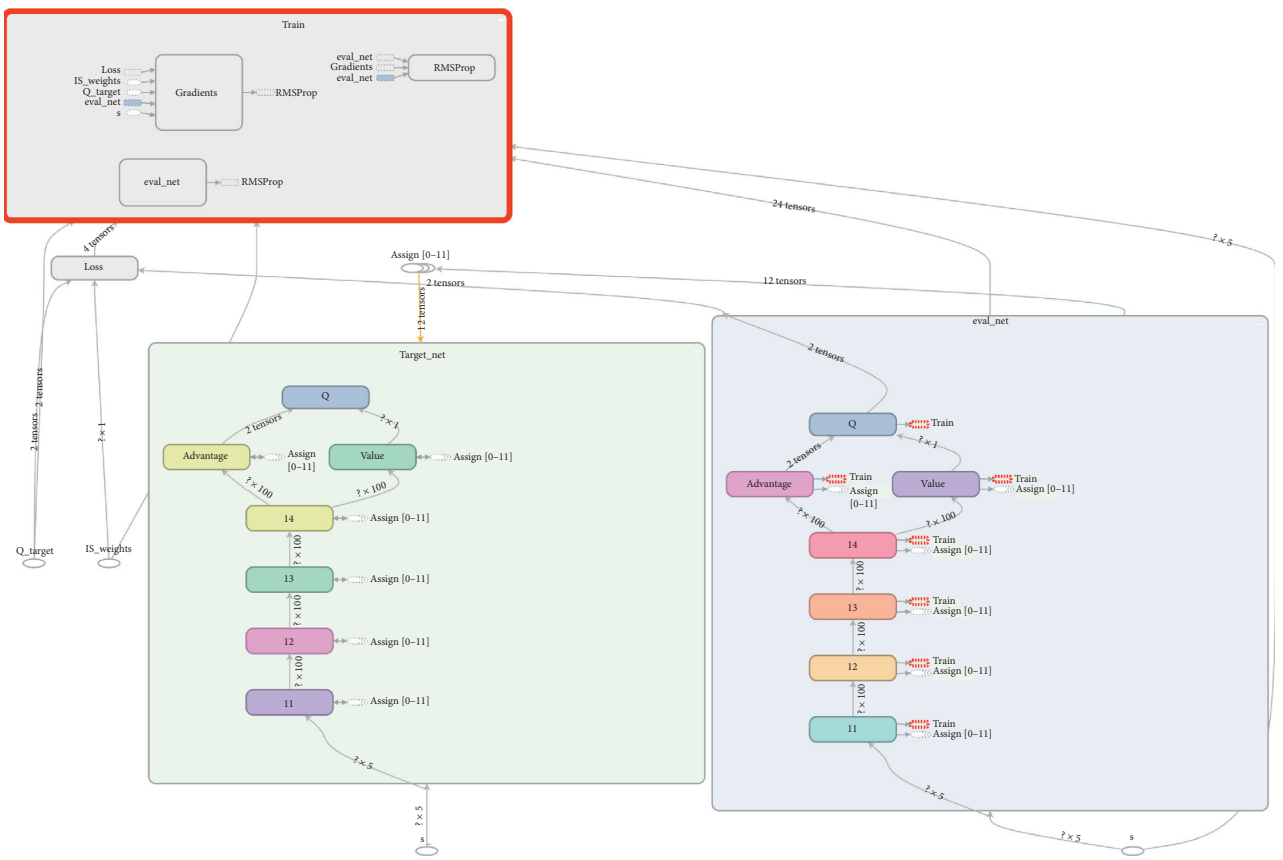

Figure 2: Dueling double deep Q-network structure.

time slots required to fully charge two harvesters. The average time-consumption is obtained over 1000 testing data. Figure 3 shows that if the deep Q-network has multiple hidden layers, a smaller learning rate is necessary to achieve better performance. When the learning rate is 0.1, the DQN with 4 hidden layers performs worse than a neural network with 2 or 3 hidden layers. On the other side, when the learning rate decreases, we can see that the neural network with 4 hidden layers and a learning rate of 0.00005 achieves the best overall performance. We do not see a monotonic decrease in the average number of time slots due to the stochastic nature of the channel that causes some fluctuations in the DQN optimization. After an initial improvement, decreasing the learning rate results in a slight increase in the average number of charging steps for all three neural network structures. This is due to the fixed number of training episodes. As a result, for all the simulations presented in this section, we consider a DQN algorithm using a 4 hidden layer deep neural network, with 100 nodes in each layer and a learning rate of 0.00005.

In Figure 4, we can observe that the size of the experience pool also affects the performance of DQN (40000 training episodes). To eliminate the correlation between the training data, we select part of the experience pool for training. In our

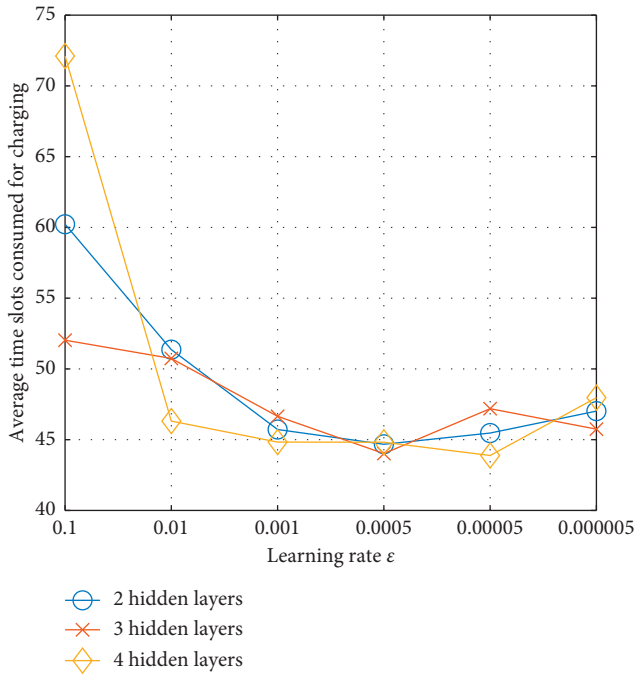

FIGURE 3: Deep Q-network performance on different learning rates and number of hidden layers for the neural network.

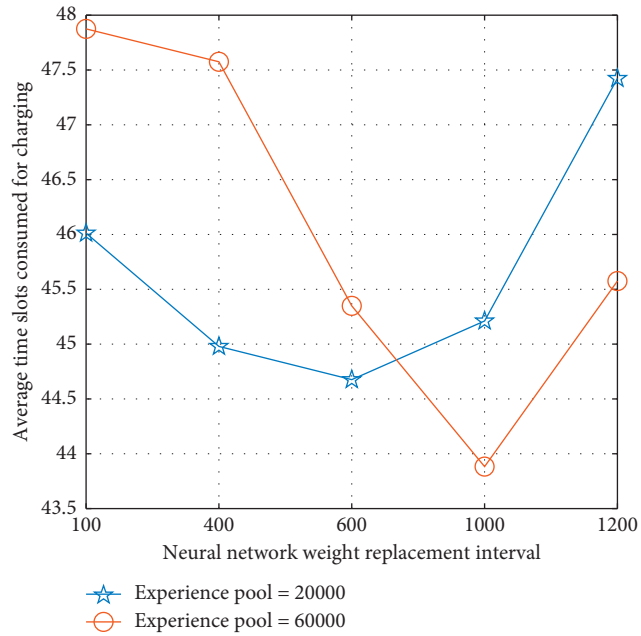

FIGURE 4: Deep Q-network performance for different values of neural network replacement iteration interval and experience pool.

simulation, this parameter, called mini-batch, is set to 10. Larger experience pool contains more training data; hence, selecting the mini-batch from it for training can eliminate the correlation between the training data. However, we need to balance the size of the experience pool and the target_net weight replacement interval. If the experience pool is large but the replacement iteration interval is small, even if we address the correlation problem between the training data, the neural network does not have enough training episodes to reduce the training error before the weight of the target_net is replaced. From Figure 4, we can observe that a large number of replacement iteration intervals may not be the best choice too. Therefore, we determine that, for our problem, DQN achieves the best performance when the size of the experience pool and the neural network replacement iteration interval are 60000 and 1000, respectively.

Figure 5 shows the impact of the reward function (see Section 3.1) on the DQN performance. In this figure, we consider the following reward functions: Reward_1: $w(s, \mathbf{a}, s') = 0$ if $s' = s_G$ and $w(s, \mathbf{a}, s') = -\lambda(\text{KB}_{\max} - \sum_{i=1}^{K} \min(s_i', B_{\max}))$ otherwise; Reward_2: $w(s, \mathbf{a}, s') = 10$ if $s' = s_G$ and $w(s, \mathbf{a}, s') = -1$ otherwise; Reward_3: $w(s, \mathbf{a}, s') = 1$ if $s' = s_G$ and $w(s, \mathbf{a}, s') = -1$ otherwise. Here, $K = 2$ and $\lambda = 0.25$. All three reward functions are designed to minimize the number of time slots required to fully charge all the harvesters. However, from Figure 5, we can observe that the best performance can be obtained using Reward_1. In this case, the energy level accumulated by each harvester increases uniformly, which results in the DQN to converge faster to the optimal policy. Both Reward_2 and Reward_3, instead, do not penalize states that unevenly charge the harvesters and therefore require more iterations to converge to the optimal solution (not shown in the figure) due to the large number of system states to explore. Therefore, in the following simulations, we use the reward function Reward_1 in both Figures 5 and 6, we average 40000 training steps every 100 steps in order to better show the convergence of the algorithm.

Figure 6 shows that when each energy harvester in the system is equipped with a larger energy buffer, the number of system states increases, and therefore, DQN requires more training period to converge to the steady transmit strategy for each system state. We can observe that when $B_{\max} = 1.6$ mJ, the system only needs less than 5000 training episodes to converge to the optimal strategy, and when $B_{\max} = 3.2$ mJ, the system needs around 12000 training episodes to converge to the optimal policy. However, for $B_{\max} = 4.8$ mJ, the system needs as many as 20000 training episodes to converge to the optimal strategy.

In the following simulations, we explore the impact of the channel model on optimization problem $\mathscr{P}_1$. For the Rician fading channel model, we consider $K_{im}^r \geq 10$ and to be the same for all i, m. In this way, we can approximate the Rician distribution as a Gaussian distribution. We fix $r_{im} = 0.5$, $\forall i, m$, but allow the standard deviation of both the amplitude and the phase of the channel to change to evaluate the performance on the system under different channel conditions. Since $r_{im} = 0.5$ and $\sqrt{2K_{im}^r} = r_{im}/\sigma_{im}$, $1/\sqrt{2K_{im}^r} = 2\sigma_{im}$. We define $\sigma_{im} \leq 0.1$ to guarantee $K_{im}^r \geq 10$.

In Figure 7, we express the standard deviation $\sigma_{\text{amp}} = \sigma_{im}$, $\forall i, m$ of the phase and amplitude of the channel, and we compare the performance attained by the optimal policy with the performance of different other algorithms. The multiarmed bandit (MAB) algorithm is also implemented to compare with the DQN. In MAB, each bandit arm represents a particular transmission pattern. The upper confidence bound (UCB) algorithm [27] is implemented to

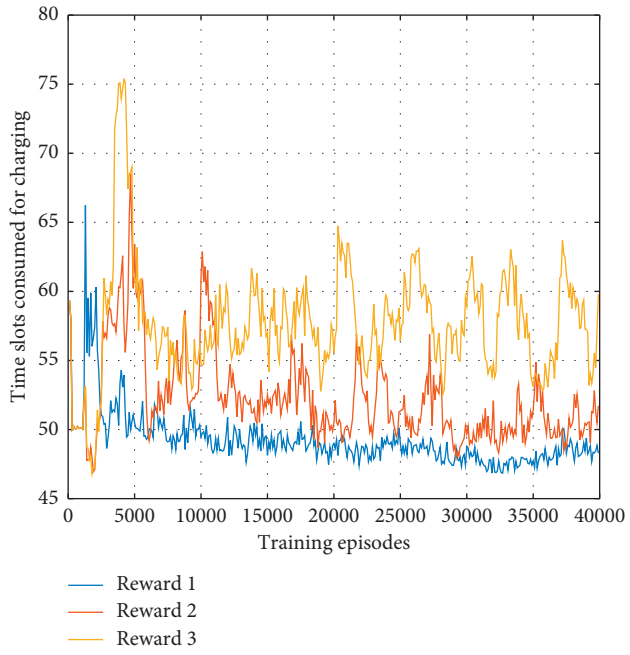

Figure 5: The deep Q-network performance for different reward functions.

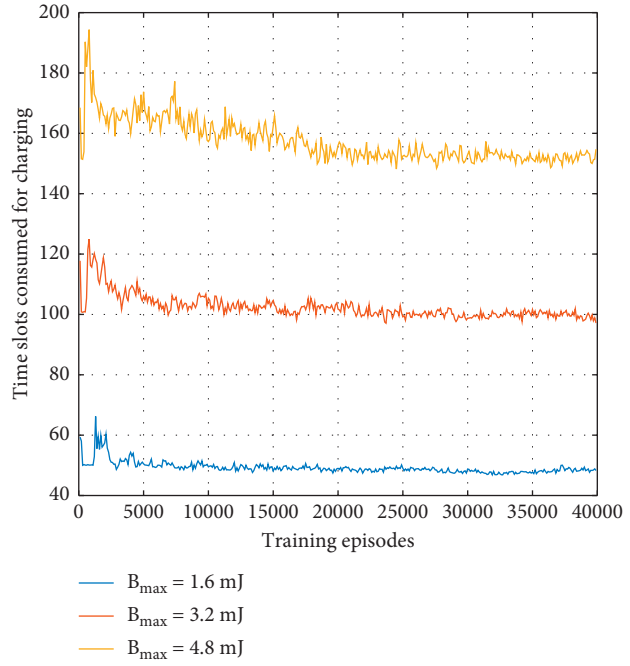

Figure 6: The deep Q-network performance for different energy buffer size B_{max}.

maximize the reward $w(s, \mathbf{a}, s')$ and determine the optimal action. Once the action is selected from the action space $\mathcal{A}$, it will be used for transmission continuously. The myopic algorithm is another machine learning algorithm that can be compared with DQN. Myopic solution has the same structure as the DQN; however, the reward discount is

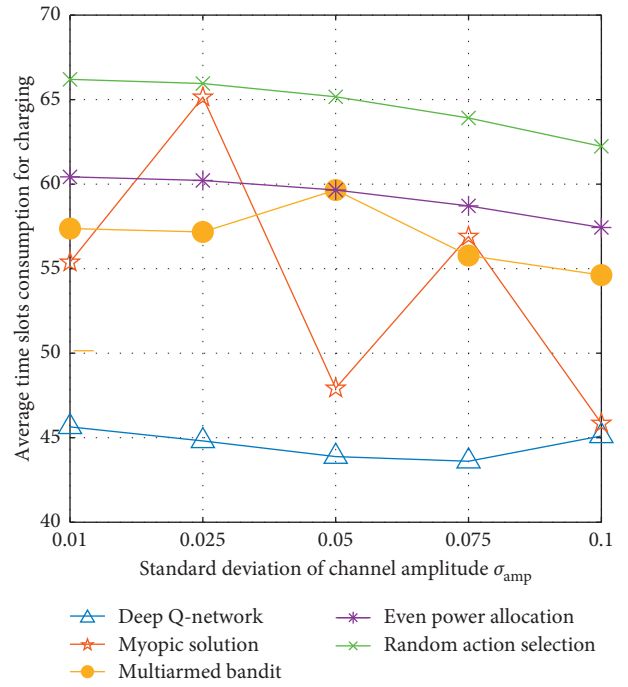

Figure 7: The comparison of average time consumption between DQN and other algorithms (myopic solution, multiarmed bandit, even power allocation, and random action selection) in the Rician fading channel model. The number of transmit antennas is $M = 3$. The number of energy harvesters is $N = 2$.

defined as $\gamma = 0$. As a result, the optimal strategy is determined only according to the current observation instead of considering the future consequence. Myopic solution has been widely used to solve the complex optimization in wireless communication problem and achieve good system performance [28]. Besides two machine learning algorithms, another two heuristic algorithms are also used for system performance comparison. For even power allocation, the transmit power P is evenly allocated on parallel channel for transmission. The random action selection is also applied for performance comparison. The random action selection has the worst performance while DQN performs best. Compared to the optimal existing algorithm multiarmed bandit algorithm, the DQN can consume 20% fewer time slots to complete charging. In some channel conditions, the myopic solution can achieve a similar performance as the DQN. However, the myopic solution cannot perform stably. For example, as the standard deviation of the channel amplitude is $\sigma_{amp} = 0.025$, DQN can outperform myopic solution by 45%. The instability can be explained as the myopic solution makes the decision only on the current system state and the current reward, which does not consider the future consequence. Hence, the training effects cannot be guaranteed. Overall, the DQN has superiority in both the charging time consumption and performing stability corresponding to different channel conditions.

To better explain the performance of the optimal policy, in Figure 8, we plot the action selected by DQN at a particular system state when $\sigma_{amp} = 0.05$. When $\sigma_{amp} = 0.05$,

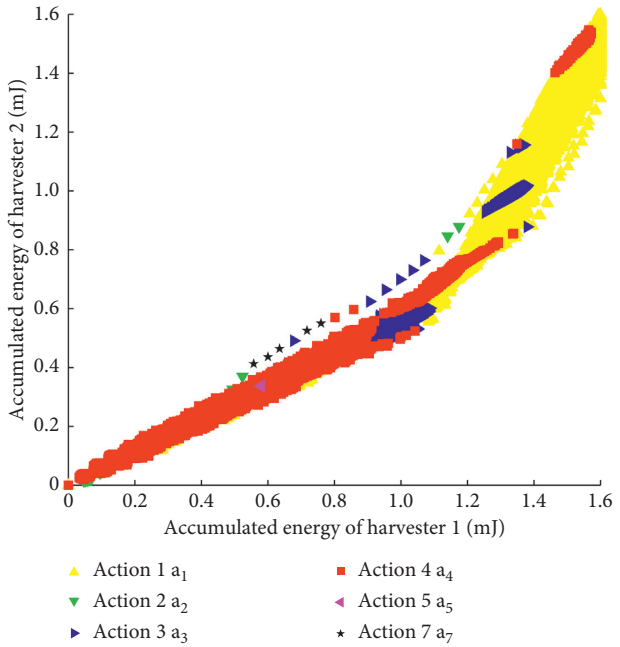

FIGURE 8: The action selection process of two harvesters scenario when $\sigma_{\text{amp}} = 0.05$.

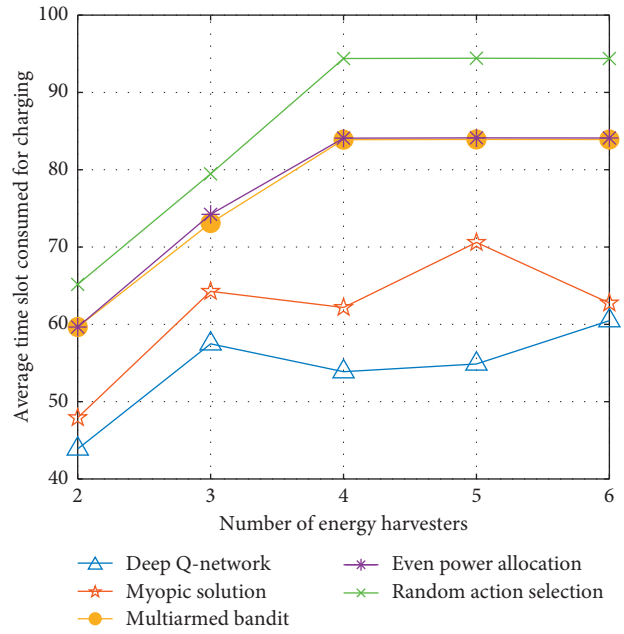

FIGURE 9: The comparison of average time consumption between DQN and other algorithms (myopic solution, multiarmed bandit, even power allocation, and random action selection) when the number of energy harvesters is $N = 2, 3, 4, 5, 6$. The number of transmit antennas is $M = 3$. The standard deviation of the channel amplitude is $\sigma_{\text{amp}} = 0.05$.

the optimal action selected by multiarmed bandit is the third action $\mathbf{a}_3 = [2, 6, 4]^T$, which can finish charging both harvesters in around 60 time slots. Meanwhile, the optimal policy determined by DQN can finish charging in around 43 time slots. To this end, Figure 8 shows that the charging process can actually be divided into two parts: before harvester 1 accumulates 1.2 mJ energy and harvester 2 accumulates 0.8 mJ energy, mostly action 4 $\mathbf{a}_4 = [2, 8, 2]^T$ is selected. After that, mostly action 1 $\mathbf{a}_1 = [2, 2, 8]^T$ is selected. As defined above, both the amplitude and the phase of the channel are Gaussian distributed with zero standard deviation, $\widehat{\mathbf{g}}_1^0 = [0.05, 0.59, 0.11]^T$ and $\widehat{\mathbf{g}}_2^0 = [0.04, 0.19, 0.51]^T$. So when both the amplitude and the phase of the channel change, the simplified channel state information will be distributed around $\widehat{\mathbf{g}}_1^0$ and $\widehat{\mathbf{g}}_2^0$. As a result, it can be shown that a policy that selects either action 1 or action 4 with different probabilities can have better performance than the policy that only selects action 3. Henceforth, the DQN can consume 40% fewer time slots to fully charge two energy harvesters.

In Figure 9, the performance of the DQN is compared with the other four algorithms by varying the number of energy harvesters in the system. In general, as the number of energy harvesters increases, all four algorithms consume more time slots to complete the wireless charging process. Compared to the random action selection, DQN can consume at least 58% less time slots to complete the charging. The performance of the multiarmed bandit and the even power allocation is very similar, which can be explained as the optimal action determined by the multiarmed bandit algorithm is close to the even power allocation strategy. Compared with two fixed action selection strategies (multiarmed bandit and even power allocation), DQN can reduce the time consumption by up to 72% (when the number of energy harvesters is $N = 3$). The myopic solution is still not the optimal strategy. From the figure, we can observe that the myopic solution outperforms two fixed action selection algorithms. Even though in some conditions ($N = 6$), the performance difference between DQN and myopic solution is very small, the myopic solution consumes more than 15% of the time slot than DQN in average. Overall, the DQN is the optimal algorithm which consumes fewest time slots to fully charge all the energy harvesters regardless of the number of energy harvesters.

In Figure 10, the number of transmit antennas is increased from $M = 3$ to $M = 4$. The number of energy harvesters varies from $N = 2$ to $N = 6$. Though the number of antennas increases, the channel conditions between the transmitter and the energy harvesters become more complicated; DQN still outperforms all the other four algorithms. Compared with myopic solution, multiarmed bandit, even action selection, and random action selection, DQN can consume up to 27%, 54%, 55%, and 76% fewer time slots to fulfill the wireless charging, respectively. As the number of energy harvesters increases, the superiority of the DQN becomes more obvious compared to two fixed action selection algorithms, which can be explained as it is more inefficient to select one fixed action to deal with a more complicated varying channel environment. Even though in some conditions, the performance of the myopic solution and DQN is similar, the myopic solution is not stable in dealing with different energy harvesters conditions. The

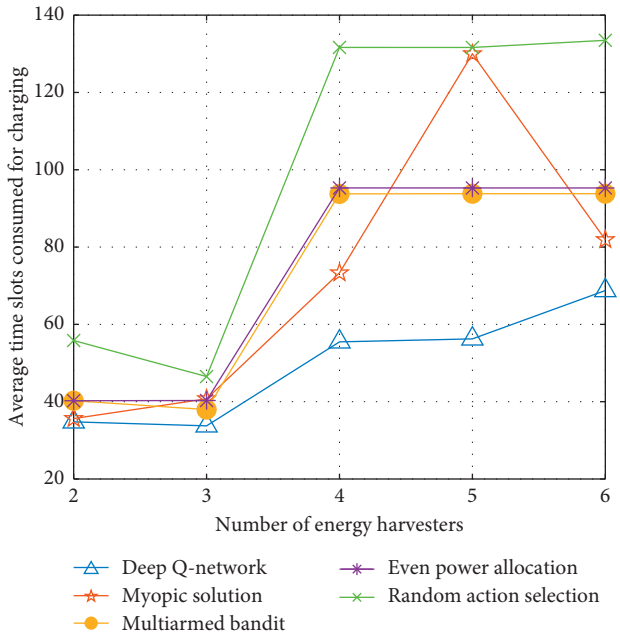

FIGURE 10: The comparison of average time consumption between DQN and other algorithms (myopic solution, multiarmed bandit, even power allocation, and random action selection) when the number of energy harvesters is $N = 2, 3, 4, 5, 6$. The number of transmit antennas is $M = 4$. The standard deviation of the channel amplitude is $\sigma_{\text{amp}} = 0.05$.

results from both Figures 9 and 10 demonstrate the superiority of the DQN in optimizing the time consumption for wireless power transfer.

5. Conclusions

In this paper, we design the optimal wireless power transfer system for multiple RF energy harvesters. Deep learning methods are used to enable the wireless transmitter to fully charge the energy buffers of all energy harvesters in the shortest time while meeting the information rate requirement of the communication system.

As the channel conditions between the transmitter and the energy harvesters are time-varying and unknown, we model the problem as a Markov decision process. Due to the large number of system states in the model and the difficulty of training, we adapt a deep Q-network approach to find the best transmit strategy for each system state. In the simulation section, multiple experimental environments are explored. The measured real-time data are used to run the simulation. Deep Q-network is compared with the other four existing algorithms. The simulation results validate that the deep Q-network is superior to all the other algorithms in terms of the time consumption for fulfilling wireless power transfer.

Data Availability

The simulation data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] R. Zhang and C. K. Ho, "MIMO broadcasting for simultaneous wireless information and power transfer," *IEEE Transactions on Wireless Communications*, vol. 12, no. 5, pp. 1989–2001, 2013.

[2] S. Lee, L. Liu, and R. Zhang, "Collaborative wireless energy and information transfer in interference channel," *IEEE Transactions on Wireless Communications*, vol. 14, no. 1, pp. 545–557, 2015.

[3] Z. Zong, H. Feng, F. R. Yu, N. Zhao, T. Yang, and B. Hu, "Optimal transceiver design for SWIPT in K-User MIMO interference channels," *IEEE Transactions on Wireless Communications*, vol. 15, no. 1, pp. 430–445, 2016.

[4] Y. Xing, Y. Qian, and L. Dong, "Deep learning for optimized wireless transmission to multiple rf energy harvesters," in *Proceedings of the 2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, Chicago, IL, USA, August 2018.

[5] J. Park and B. Clerckx, "Joint wireless information and energy transfer in a two-user MIMO interference channel," *IEEE Transactions on Wireless Communications*, vol. 12, no. 8, pp. 4210–4221, 2013.

[6] W. Wu, X. Zhang, S. Wang, and B. Wang, "Max-min fair wireless energy transfer for multiple-input multiple-output wiretap channels," *IET Communications*, vol. 10, no. 7, pp. 739–744, 2016.

[7] A. Thudugalage, S. Atapattu, and J. Evans, "Beamformer design for wireless energy transfer with fairness," in *Proceedings of the 2016 IEEE International Conference on Communications (ICC)*, pp. 1–6, Kuala Lumpur, Malaysia, May 2016.

[8] Y. Xing and L. Dong, "Passive radio-frequency energy harvesting through wireless information transmission," in *Proceedings of the 2017 13th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pp. 73–80, Ottawa, ON, Canada, June 2017.

[9] P.-V. Mekikis, A. Antonopoulos, E. Kartsakli, A. S. Lalos, L. Alonso, and C. Verikoukis, "Information exchange in randomly deployed dense wsns with wireless energy harvesting capabilities," *IEEE Transactions on Wireless Communications*, vol. 15, no. 4, pp. 3008–3018, 2016.

[10] J. Xu and R. Zhang, "A general design framework for mimo wireless energy transfer with limited feedback," *IEEE Transactions on Signal Processing*, vol. 64, no. 10, pp. 2475–2488, 2016.

[11] K. W. Choi, D. I. Kim, and M. Y. Chung, "Received power-based channel estimation for energy beamforming in multiple-antenna RF energy transfer system," *IEEE Transactions on Signal Processing*, vol. 65, no. 6, pp. 1461–1476, 2017.

[12] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Playing atari with deep reinforcement learning," 2013, https://arxiv.org/abs/1312.5602.

[13] Y. Xing, Y. Qian, and L. Dong, "A multi-armed bandit approach to wireless information and power transfer," *IEEE Communications Letters*, vol. 24, no. 4, pp. 886–889, 2020.

[14] Y. He, Z. Zhang, F. R. Yu et al., "Deep reinforcement learning-based optimization for cache-enabled opportunistic interference alignment wireless networks," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 11, pp. 10 433–510 445, 2017.

[15] J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," 2016, https://arxiv.org/abs/1605.06676.

[16] Z. Xu, Y. Wang, J. Tang, J. Wang, and M. C. Gursoy, "A deep reinforcement learning based framework for power-efficient resource allocation in cloud rans," in *Proceedings of the 2017 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, Paris, France, May 2017.

[17] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," 2016, https://arxiv.org/abs/1509.06461.

[18] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas, "Dueling network architectures for deep reinforcement learning," 2015, https://arxiv.org/abs/1511.06581.

[19] E. Biglieri, R. Calderbank, A. Constantinides, A. Goldsmith, A. Paulraj, and H. V. Poor, *MIMO Wireless Communications*, Cambridge University Press, Cambridge, UK, 2007.

[20] S. Timotheou, I. Krikidis, S. Karachontzitis, and K. Berberidis, "Spatial domain simultaneous information and power transfer for mimo channels," *IEEE Transactions on Wireless Communications*, vol. 14, no. 8, pp. 4115–4128, 2015.

[21] D. Mishra and G. C. Alexandropoulos, "Jointly optimal spatial channel assignment and power allocation for mimo swipt systems," *IEEE Wireless Communications Letters*, vol. 7, no. 2, pp. 214–217, 2018.

[22] A. G. Barto, S. J. Bradtke, and S. P. Singh, "Learning to act using real-time dynamic programming," *Artificial intelligence*, vol. 72, no. 1-2, pp. 81–138, 1995.

[23] L. Dong and Y. Liu, "Parallel sub-channel transmission for cognitive radios with multiple antennas," *Wireless Personal Communications*, vol. 79, no. 3, pp. 2069–2087, 2014.

[24] Y. Xing, H. Pan, B. Xu, T. Zhao, C. Tapparello, and Y. Qian, "Multiuser data dissemination in OFDMA system based on deep q-network," in *Proceedings of the IEEE IEMTRONIC (International IOT, Electronics and Mechatronics Conference)*, Toronto, Canada, April 2021.

[25] J. Cavers, *Mobile Channel Characteristics*, Springer Science & Business Media, Berlin, Germany, 2006.

[26] T.-Q. Wu and H.-C. Yang, "On the performance of overlaid wireless sensor transmission with rf energy harvesting," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 8, pp. 1693–1705, 2015.

[27] A. Slivkins, "Introduction to multi-armed bandits," 2019, https://arxiv.org/abs/1904.07272.

[28] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access in wireless networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 2, pp. 257–265, 2018.