EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Dynamic spectrum access and sharing through actor-critic deep reinforcement learning

Liang Dong[1*], Yuchen Qian[2] and Yuan Xing[3]

*Correspondence: liang_dong@baylor.edu

[1] Department of Electrical and Computer Engineering, Baylor University, Waco, TX 76798, USA
[2] Facebook, Inc., Menlo Park, CA, USA
[3] Engineering and Technology Department, University of Wisconsin-Stout, Menomonie, WI 54751, USA

Abstract

When primary users of the spectrum use frequency channels intermittently, secondary users can selectively transmit without interfering with the primary users. The secondary users adjust the transmission power allocation on the frequency channels to maximize their information rate while reducing channel conflicts with the primary users. In this paper, the secondary users do not know the spectrum usage by the primary users or the channel gains of the secondary users. Based on the conflict warnings from the primary users and the signal-to-interference-plus-noise ratio measurement at the receiver, the secondary users adapt and improve spectrum utilization through deep reinforcement learning. The secondary users adopt the actor-critic deep deterministic policy gradient algorithm to overcome the challenges of large state space and large action space in reinforcement learning with continuous-valued actions. In addition, multiple secondary users implement multi-agent deep reinforcement learning under certain coordination. Numerical results show that the secondary users can successfully adapt to the spectrum environment and learn effective transmission policies.

Keywords: Dynamic spectrum access, Spectrum sharing, Power allocation, Reinforcement learning, Deep deterministic policy gradient, Multi-agent cooperation

1 Introduction

Dynamic spectrum access and sharing can alleviate the spectrum congestion problem of wireless communications. The primary users are incumbent spectrum users who share underutilized spectrum with the secondary users. Traditionally, the secondary users perform spectrum sensing and transmit in spectrum holes to avoid interference to the primary users. However, when the primary users quickly switch to different frequency channels, the secondary users may obtain outdated spectrum usage information through spectrum sensing. Alternatively, the secondary users can predict and adapt to the spectrum usage of the primary users [1–5]. We assume that when a primary user experiences excessive interference from the secondary users on a specific frequency channel, it will issue some indication or warning. Over time, the secondary users are alerted to these channel conflicts and extrapolate future spectrum availability. They will gradually adapt to the primary user's spectrum usage and transmit accordingly to avoid or reduce interference to the primary users. At the same time, the secondary users allocate transmission power over available frequency channels to increase their information rates.

In this paper, the secondary users predict and adapt to the spectrum usage of the primary users through reinforcement learning. With reinforcement learning, a secondary user is an agent who can observe the state of the communication network, take transmission actions, get rewards, and adjust its transmission strategy. The secondary users learn the preferred transmission policies through interaction with the communication network. They test different frequency channels by transmitting on those channels. If the interference to the primary user on a frequency channel exceeds a threshold, the primary user will issue warnings about this specific channel. The secondary users receive and store these warnings to guide their future decisions. By designing rewards related to the ultimate goal, the secondary users can refine transmission policies that specify the frequency channels to use and the transmission power levels on those channels.

The reinforcement learning method is a practical solution for secondary user transmission, as it circumvents the problematic requirements of training labels in supervised learning. There are other challenges. First, the state space of reinforcement learning is large for spectrum access and sharing applications. For this, we employ deep reinforcement learning with the deep *Q*-networks [6]. Second, since agent actions are continuous-valued transmission power levels on the frequency channels, the action space of reinforcement learning is large. For this, we implement deep deterministic policy gradient algorithms [7–10]. These algorithms exploit an actor-critic architecture that leverages policy-based and value-based learning methods to deliver superior performance. We design the deep neural networks in the actor-critic framework to learn the temporal and spectral correlations of channel occupancy.

To deal with multiple secondary users coexisting in a communication network, we apply multi-agent deep reinforcement learning for dynamic spectrum access and sharing. Each secondary user is an autonomous agent who can interact with the communication network and make strategic decisions. Together, the agents want to avoid or reduce interference to the primary users while maximizing the sum information rate of the secondary users. Multiple agents can coordinate with each other. In the secondary user network, the agents can (1) exchange the information of rewards, (2) exchange the information of rewards and state observations, or (3) exchange the information of rewards, state observations, and actions. More coordination leads to better spectrum utilization at the cost of slightly higher communication overhead. Numerical results show that the secondary users can successfully access and share the spectrum with the primary users through actor-critic deep reinforcement learning.

In contrast to related research work, in this work, the secondary users apply deep reinforcement learning to the problem of joint channel selection and power control. The channel state information (CSI) is entirely unknown to the secondary users. Learning from the channel conflict warnings and the received signal-to-interference-plus-noise ratio (SINR), the secondary users select and multiplex frequency channels for transmission. The secondary users allocate continuous-valued transmission power on the channels. By designing the reward function, the secondary users can adapt and maximize the sum information rate while limiting interference to the primary users. The neural networks of the actor-critic learning framework are trained online in a distributed fashion. The rest of the paper is organized as follows. We review recent research on the application of deep reinforcement learning to spectrum access and sharing in Sect. 2. In Sect. 3, we present the system model and problem formulation. In Sect. 4, we propose the dynamic spectrum access and sharing method based on reinforcement learning. The actor-critic deep reinforcement learning algorithms are elaborated in Sect. 5. In Sect. 6, we discuss multi-agent deep reinforcement learning for multiple secondary users to manage spectrum and interference. Experimental methods are described in Sect. 7. The results are given in Sect. 8, which show improved spectrum utilization and validate the method. Finally, we draw the conclusion in Sect. 9.

2 Related work

Deep reinforcement learning has been used for multi-channel access and power control in wireless communications. For the channel selection problem, Wang et al. [11] implemented a deep *Q*-network (DQN) on a single user for dynamic multi-channel access. The channel accessibility was binary, and the transition of the state of multiple correlated channels followed an unknown joint Markov model. The user action was to select one channel to sense and transmit a packet. The reward was also binary, indicating whether the transmission was successful, and the user learned to select the channel and maximize the expected number of successful transmissions. Naparstek and Cohen [12] used a dueling DQN for distributed dynamic spectrum access of multiple users. A long short-term memory (LSTM) layer was added to the DQN that maintained an internal state and aggregated observations over time. Game theoretical analysis was developed to model and analyze multi-user dynamics. Network training was done offline by a centralized unit. The user action was to select the channel to transmit. The reward was a network utility that relied on the binary indicator of successful packet transmission. Chang et al. [13, 14] used a special type of recurrent neural network (RNN), namely reservoir computing or echo state network, in deep reinforcement learning for distributed dynamic spectrum access of multiple users. The RNN exploited the temporal correlations. Each primary user broadcasted warning signals to the secondary users if a signal collision occurred. The secondary user observed the imperfect spectrum sensing outcome, and the action was to select at most one channel to access. The reward function was discrete. Zhong et al. [15] used the actor-critic reinforcement learning framework for multi-channel access for both single-user and multi-user cases. The multiple channels were correlated and each channel had a binary state. Modeling the channel selection as a partially observable Markov decision process, the user selected channels to access by learning the channel switching patterns as well as other users' action patterns. Huang et al. [16, 17] used a double DQN for spectrum access for device-to-device (D2D)-enabled cellular networks. Double DQN avoided overestimation of the *Q*-values. The D2D pairs observed the discrete environment state, including channel availability and a distance vector. There was a virtual controller that acted as a centralized learning agent. The action set was discrete, specifying which D2D nodes to transmit, and the D2D pairs' objective was to maximize the sum throughput or maximize a fairness objective function while avoiding interference to the cellular users. Doshi et al. [18] and Guo et al. [19] implemented multi-agent deep reinforcement learning in the medium access control layer for channel access. Through centralized learning and decentralized execution, each user made binary transmission decisions to maximize the long-term average rate [18] or improve network performance in terms of throughput, delay, and jitter [19].

For the power control problem, Li et al. [20] used deep reinforcement learning for the case of one primary user and one secondary user. The secondary user learned the power control policy and shared the spectrum with the primary user. A set of sensor nodes were deployed to collect received signal strength information to assist the secondary user. The action taken by the secondary user was the transmission power from a pre-specified set of power levels. The reward was binary and was obtained when both the primary and secondary receivers had SINR exceeding a threshold. Nasir and Guo [21] developed deep reinforcement learning for a distributively executed dynamic power allocation for multiple users. The transmitter collected delayed CSI and information of the interferer and interfered sets from several neighbors and adapted its transmission power accordingly. A centralized network trainer was used. Each user took action from a discrete power level set to maximize a weighted sum-rate utility. Zhang et al. [22] used deep reinforcement learning for power control of the secondary user to share the spectrum with the primary user. The learning was based on the asynchronous advantage actor critic (A3C) and distributed proximal policy optimization (DPPO) methods. A set of spatially distributed wireless sensors obtained the received signal strength and sent the information to the secondary user. Multiple secondary users learned power control policy simultaneously. The transmission power level was discrete. The objective was to achieve spectrum sharing by guaranteeing the received SINR of the primary and secondary users. Meng et al. [23] considered the dynamic downlink power control problem to maximize the sum rate in multi-user wireless cellular networks. It was centralized dynamic power allocation in a multi-user cellular network. Multi-agent deep reinforcement learning was used with centralized training and distributed execution. Implemented algorithms included policy-based REINFORCE, value-based deep Q-learning, and actor-critic deep deterministic policy gradient (DDPG) algorithms.

For the joint channel selection and power control problem, Ye et al. [24] used deep reinforcement learning for decentralized resource allocation in unicast and broadcast vehicular communications. Each vehicle-to-vehicle (V2V) link was an agent that observed the state that included the instantaneous CSI of the V2V links and the vehicle-to-infrastructure (V2I) links, previous interference information, and previous sub-channel selections of the neighbors. The action was to select the sub-channel and power level for transmission, and there were three selectable levels of transmission power. The objective was to increase the capacity of the V2I and V2V links while meeting the latency constraints of the V2V communications. Liang et al. [25] developed multi-agent reinforcement learning with a fingerprint-based DQN for spectrum and power allocation in vehicular networks. With inaccurate CSI, the agents cooperated and made distributed spectrum access decisions to improve the sum capacity of V2I links and the payload delivery rate of V2V links. It was centralized learning and distributed execution. The agent action was to select spectrum sub-bands and discrete transmission power levels. The selectable power levels were limited to four levels, thus a low-dimensional discrete action space. Xu et al. [26] used multi-agent reinforcement learning with DDPG for resource allocation in vehicular communications. The resource allocation problem was formulated as a decentralized discrete-time finite-state Markov decision process. The state of each V2V communication included the local channel information and interference channel information. The V2V agent learned both frequency spectrum allocation and transmission power control to maximize the sum rate of V2I communications and guarantee the latency and reliability of V2V communications. The transmission power level was from a continuous action space, but each V2V pair could not multiplex more than one uplink spectrum channel at a time. Li and Guo [27] used multi-agent actor-critic and neighbor-agent actor-critic frameworks of deep reinforcement learning for resource allocation in D2D underlay communications. The training was centralized with shared global historical states, actions, and policies or neighbor users' historical information. The execution was distributed. Each D2D pair's action was to select the resource block and the discrete-valued transmission power. The objective was to improve the sum rate of the D2D links while ensuring the communication quality of the cellular users. Tan et al. [28] used distributed deep reinforcement learning for joint channel selection and power control in overlay D2D networks. The state included the received signal power, the received interference plus noise, and the outdated CSI. Each D2D pair learned the correlation patterns by analyzing local information and partial outdated non-local information. It then inferred global network information to optimize channel selection and transmission power. Each D2D pair chose at most one channel to transmit at a time, and the selectable transmission power was discretized into multiple levels. Song et al. [29] used deep Q-learning with echo state networks for spectrum management that coordinated interference between secondary users and suppressed interference for primary users. The secondary users had interference information feedback from the primary users but no CSI. The transmission power had discretized levels. The user action was the three-level (increase, decrease, no change) adjustment of the power level. Yang et al. [30] used a distributed coordinated multi-agent dueling DQN for resource management in two-tier heterogeneous networks. Each cell selected its joint device association, spectrum allocation, and power allocation based on locally observed information including current CSI. The objective was to maximize the network capacity while guaranteeing the quality-of-service (QoS) requirements of mobile devices.

3 System model and problem formulation

In a wireless communication network, the primary users of the spectrum are the licensed users of N frequency channels. For dynamic spectrum access, when a frequency channel is idle with no communication usage by the primary users, the secondary users of the spectrum are allowed to access the channel. For dynamic spectrum sharing, the secondary users can transmit on a frequency channel as long as the interference at the primary user receiver is below a threshold. The secondary users are the major players of dynamic spectrum access and sharing. They intend to adapt to the channel usage pattern of the primary users to avoid channel conflicts with the licensed spectrum users.

3.1 Channel usage

The primary users use the frequency channels of the spectrum according to a channel usage and switching pattern. For ease of processing, we divide the time into equal intervals. During each time slot, the state of channel usage is represented by a vector $\mathbf{x} = [x_1, x_2, \ldots, x_N]^T$. In this vector, x_i is the state of the ith frequency channel, where

$x_i = 1$ indicates that the channel is available and $x_i = -1$ indicates that the channel is occupied. In the next time slot, the channel usage may switch to another state.

The secondary users can transmit on these frequency channels. If the channel is currently occupied by the primary user and the secondary user's transmission causes excessive interference to the primary user receiver, a warning is issued to indicate channel conflict. The warning signals of channel conflict can be sent through auxiliary channels. The secondary users receive the conflict warnings and store the history of channel conflicts in their experience databases. Through this process, the secondary users can observe the state of channel usage. It should be noted that the observation is partial. If no secondary users are transmitting on a particular frequency channel, the current channel state is ambiguous to the secondary users. In addition, if the interference to the primary user receiver is tolerable, no warning will be issued.

3.2 Transmission problem of secondary users

There are K secondary users of the spectrum who want to transmit on the N frequency channels. Suppose that the nth frequency channel ($n \in \mathcal{N} = \{1, 2, \ldots, N\}$) is free from primary usage. If multiple secondary users transmit on the same channel, they interfere with each other. The kth secondary user ($k \in \mathcal{K} = \{1, 2, \ldots, K\}$) transmits on the nth channel. Its receiver can measure the received SINR on the nth channel as

$$\mathrm{SINR}_{k,n} = \frac{|h_{kk,n}|^2 p_{k,n}}{\sum_{l \in \mathcal{K} \setminus k} |h_{kl,n}|^2 p_{l,n} + W N_0} \tag{1}$$

where $p_{k,n}$ and $p_{l,n}$ are the transmission power of the kth secondary user and the interfering secondary users on the nth frequency channel, respectively. $|h_{kk,n}|$ and $|h_{kl,n}|$ are the channel gains of the nth frequency channel between the kth transmitter and the kth receiver and between the lth transmitter and the kth receiver, respectively. The channel gains $\{|h|\}$ are of the secondary users and assumed time-invariant but *unknown*. W is the bandwidth of each frequency channel and N_0 is the single-sided noise spectral density.

The kth secondary user may transmit on multiple frequency channels. Therefore, its information rate in bits per second per hertz (bit/s/Hz) is given by

$$\begin{aligned} q_k &= \sum_{n=1}^{N} \log_2(1 + \mathrm{SINR}_{k,n}) \\ &= \sum_{n=1}^{N} \log_2 \left(1 + \frac{|h_{kk,n}|^2 p_{k,n}}{\sum_{l \in \mathcal{K} \setminus k} |h_{kl,n}|^2 p_{l,n} + W N_0} \right). \end{aligned} \tag{2}$$

The kth secondary user adjusts the transmission power $\{p_{k,n}\}_{n \in \mathcal{N}}$ to maximize its information rate. Each secondary user has a transmission power limit P such that $\sum_{n \in \mathcal{N}} p_{k,n} \le P$. If the kth secondary user does not transmit on the n'th frequency channel, it sets $p_{k,n'} = 0$. Therefore, the transmission design includes both frequency channel selection and transmission power allocation.

If the kth secondary user transmits on the mth frequency channel ($m \in \mathcal{N}$) that conflicts with any primary user, the primary user will issue a warning about the mth channel. After receiving the warning signal, the secondary user invalidates the communication effort on the mth channel by setting $\text{SINR}_{k,m} = 0$ at the receiver. This effectively eliminates the contribution to the information rate q_k that is attributed to the transmission on the mth channel.

The transmission problem of the kth secondary user can be formulated as

$$\begin{aligned} &\underset{\{p_{k,n}\}_{n\in\mathcal{N}}}{\text{maximize}} && q_k \\ &\text{subject to} && \textstyle\sum_{n\in\mathcal{N}} p_{k,n} \le P \\ &&& \text{SINR}_{k,m} = 0, \quad m \in \mathcal{M} \end{aligned} \tag{3}$$

where $\mathcal{M}$ ($\mathcal{M} \subseteq \mathcal{N}$) is the set of indexes of frequency channels that are occupied by the primary users.

In the case of spectrum sharing, when the primary user experiences excessive channel interference from the secondary users, it will issue a warning about that channel. Otherwise, the kth secondary user receives no warning and considers the transmission successful on that channel. It measures the received SINR and calculates the information rate. In this case, the SINR on the nth frequency channel is modified as

$$\text{SINR}_{k,n} = \frac{|h_{kk,n}|^2 p_{k,n}}{\sum_{l\in\mathcal{K}\setminus k} |h_{kl,n}|^2 p_{l,n} + I_n + WN_0} \tag{4}$$

where I_n is the interference from the primary user's transmission on the nth frequency channel. The transmission problem of the kth secondary user is modified as

$$\begin{aligned} &\underset{\{p_{k,n}\}_{n\in\mathcal{N}}}{\text{maximize}} && q_k \\ &\text{subject to} && \textstyle\sum_{n\in\mathcal{N}} p_{k,n} \le P \\ &&& \text{SINR}_{k,m} = 0, \quad m \in \mathcal{M} \ \& \ p_m^R > \Gamma \end{aligned} \tag{5}$$

where p_m^R is the interference power on the mth frequency channel received by the primary user and Γ is the threshold. When the interference power exceeds the threshold, a warning signal of the mth channel is issued.

Problems (3) and (5) cannot be solved by optimization methods because

1 The secondary user does not know the channel usage and switching pattern of the primary users. That is, set $\mathcal{M}$ and how it changes are unknown.
2 The secondary user does not know the channel gains $\{|h|\}$ that are in the expressions of $\{\text{SINR}_{k,n}\}$ and in the information rate q_k as revealed in (2).
3 The secondary user does not know the interference from the primary user's transmission I_n that are in the expressions of $\{\text{SINR}_{k,n}\}$ in (4).

The secondary users dynamically adjust transmission power to maximize the information rate. At the same time, the secondary users adapt to avoid channel conflicts with the primary users. In the case of dynamic spectrum access, we want $p_m^R = 0, m \in \mathcal{M}$. In the case of dynamic spectrum sharing, we want $p_m^R \le \Gamma, m \in \mathcal{M}$. To achieve this goal, we develop a spectrum access and sharing method that is based on deep reinforcement learning for the secondary users.

4 Dynamic spectrum access and sharing through reinforcement learning

The secondary users apply reinforcement learning algorithms to adapt to the wireless communication environment. Each secondary user is an agent. In the following, we study the kth secondary user as a specific agent and omit the subscript k for clarity. In reinforcement learning, the agent observes the system state at each time slot, takes action, and receives a reward.

4.1 State and observation

The agent obtains knowledge about the state of channel usage through observation. If any secondary user transmits on a frequency channel that conflicts with the primary user, the agent receives the warning and marks the channel as occupied with "-1". If the agent transmits and receives the warning of the channel, it sets the received SINR to zero when calculating the information rate. The warning may be caused by other secondary user's transmission on the channel. If there is no warning of the channel, and the agent successfully transmits on the channel, the agent observes that the channel is available and marks it with "1". However, if there is no warning of the channel and the agent does not transmit on that channel, the agent is ambiguous about the channel usage and leaves it as "0." Therefore, the observation of the primary user's usage of a frequency channel is ternary from $\{-1, 0, 1\}$. The agent's observation of the state of channel usage is partial.

To capture the pattern of channel usage and switching, the agent expands the observation space to include what was observed in the previous T_0 time slots. At time slot t, the agent has the observation of the system state across T_0 consecutive time slots. It is given by

$$o_t = \left\{\hat{\mathbf{x}}_{t-T_0}, \hat{\mathbf{x}}_{t-T_0+1}, \ldots, \hat{\mathbf{x}}_{t-1}\right\} \tag{6}$$

where $\hat{\mathbf{x}}$ is the observation of the state vector, whose elements are ternary from $\{-1, 0, 1\}$.

The system state is the channel occupancy by the primary users. It is independent of the communication behavior of the secondary users. However, the agent's observation of the system state can be affected by other secondary users' transmission.

4.2 Action

The variables of optimization problems (3) and (5) are the secondary user's transmission power on the frequency channels. Likewise, the action of the agent at time slot t is the transmission power as

$$a_t = \{p_{n,t}\}_{n \in \mathcal{N}} \tag{7}$$

where $p_{n,t}$ is the agent's transmission power on the nth frequency channel. The action is continuous-valued. It satisfies the condition that $\sum_{n=1}^{N} p_{n,t} \leq P$, where P is the transmission power limit. If the agent does not transmit on the n'th frequency channel, it is simply expressed as $p_{n',t} = 0$.

The agent tends to transmit at full power on available frequency channels to maximize the information rate, that is, $\sum_{n=1}^{N} p_{n,t} = P$. An exception is when the agent expects no available channel in the current time slot. In this case, the agent does not transmit resulting $\sum_{n=1}^{N} p_{n,t} = 0$.

4.3 Reward

Reward design is crucial for reinforcement learning to make the learning algorithm a suitable replacement for optimization problems (3) and (5). At time slot t, the agent takes action a_t and gets reward r_t that is given by

$$r_t = q_t - \beta \sum_{m \in \mathcal{M}_t} p_{m,t} \tag{8}$$

where q_t is the information rate defined in (2) for the agent in time slot t. The larger the information rate, the greater the reward. The second term in (8) is the penalty due to conflict with the primary users. $\mathcal{M}_t$ is the set of indexes of frequency channels on which the secondary transmission causes conflicts with the primary user's current usage. The penalty is effective when the agent receives the channel warning while transmitting on that channel (nonzero $p_{m,t}$). The more transmission power the agent uses on the conflicting channels, the greater the penalty. β is a scalar that strikes a balance between maximizing the information rate and minimizing the channel conflict with the primary users.

Here, we circumvent the problem that involves the received interference power p_m^R at the primary users as in (5). It is impractical for the agent (secondary user) to acquire p_m^R. The only clue it has is the primary users' warning when $p_m^R > \Gamma$. In the reward design, we transfer the problem of received interference power at the primary users to the penalty on the transmission power $p_{m,t}$ at the secondary user.

High rewards encourage the agent to allocate transmission power on frequency channels that maximize the information rate. At the same time, the penalty term gives the agent an incentive to avoid channel conflicts with the primary users. With a large value of β, the agent strives to transmit on channels that do not conflict with the primary users. In practice, β is a hyperparameter that can be tuned empirically.

The goal of reinforcement learning is to find the optimal policy. Following this policy, the agent will take the best action when observing the system state. With the roll-out of the "state, action, reward, next_state" trajectory, the optimal policy will maximize the cumulative discounted reward, which is given by

$$R_t = \sum_{\tau=t}^{T} \gamma^{(\tau-t)} r_\tau, \quad 0 \leq \gamma \leq 1 \tag{9}$$

where γ is the discount rate and T is the time horizon.

In dynamic spectrum access and sharing, the secondary user's actions do not affect which frequency channels the primary users will occupy in the future. It may affect the way the secondary user probes the frequency channels by transmitting on specific channels. The impact of the current action on future rewards is limited. Therefore, in the algorithm implementation, we use a moderate γ, e.g., $\gamma = 0.5$.

5 Actor-critic deep reinforcement learning for dynamic spectrum access and sharing

When the secondary user applies reinforcement learning for dynamic spectrum access and sharing, the state space of the reinforcement learning is large. Therefore, we use deep reinforcement learning algorithms with deep Q-networks. In addition, the secondary user's action is to adjust the levels of transmission power. The continuous-valued actions lead to a large action space of the reinforcement learning. This prompts us to adopt deep deterministic policy gradient (DDPG) [7, 8] and twin delayed deep deterministic policy gradient (TD3) [10] algorithms for dynamic spectrum access and sharing.

5.1 Actor-critic deep deterministic policy gradient algorithm

The DDPG algorithm is an actor-critic algorithm that integrates the deep Q-network with the deterministic policy gradient algorithm. It can concurrently learn a Q-function and a policy in a high-dimensional and continuous action space [23, 26, 31]. The algorithm maintains an actor function to deterministically map state observations to actions and maintains a critic function to estimate the Q-values of action-state pairs. The actor-critic architecture takes advantage of value-based and policy-based learning methods.

The critic estimates the action-value function, i.e., the Q-function. The Q-function describes the expected cumulative discounted reward after taking an action a_t on observation o_t following policy π. That is

$$Q^{\pi}(o_t, a_t) = \mathbb{E}_{\pi}[R_t \mid o_t, a_t] \tag{10}$$

where $\mathbb{E}_{\pi}$ is the expectation with the states taken in the state space (equivalently, the observations taken in the observation space) and the actions thereafter following policy π. In reinforcement learning, the Q-function is expressed recursively according to the Bellman equation

$$Q^{\pi}(o_t, a_t) = \mathbb{E}[r(o_t, a_t) + \gamma \mathbb{E}_{\pi}[Q^{\pi}(o_{t+1}, a_{t+1})]]. \tag{11}$$

The Q-function $Q^{\pi}(o_t, a_t)$ can be approximated by a Q-network parameterized with θ^{Q}. That is

$$Q(o_t, a_t \mid \theta^{Q}) = Q^{\pi}(o_t, a_t). \tag{12}$$

A deep neural network implements the Q-network. Its details are discussed in Sect. 5.3.

The actor updates the policy distribution in the direction suggested by the critic with policy gradients. Policy gradient methods are used in model-free reinforcement learning, because the state transitions are unknown to the agent. When the policy is deterministic, the actor directly maps observation o_t to action a_t. This can be approximated by a policy network parameterized with θ^{μ}. That is

$$a_t = \mu(o_t \mid \theta^{\mu}). \tag{13}$$

A deep neural network implements the policy network. Its details are discussed in Sect. 5.3.

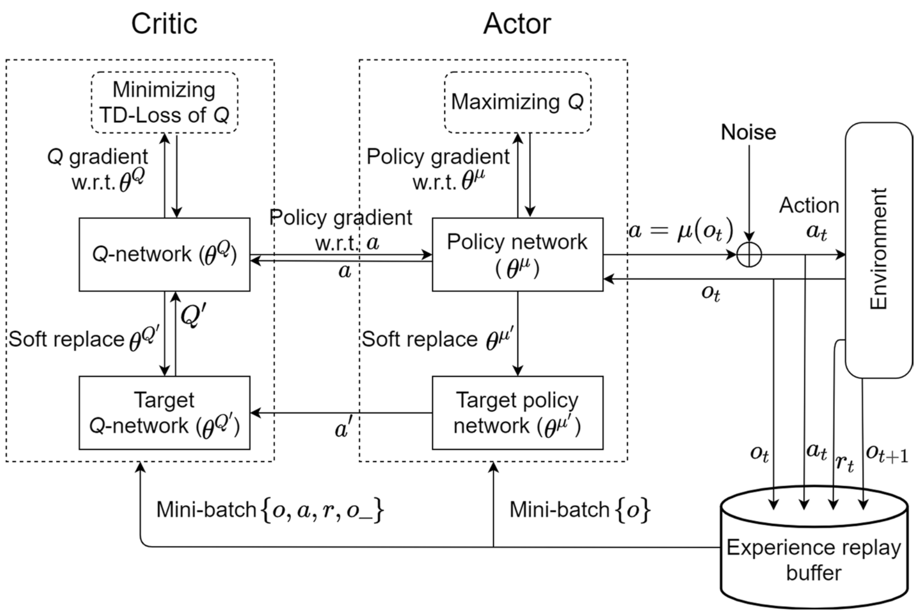
Critic
Actor
Minimizing TD-Loss of Q
Q gradient w.r.t. θ^Q
Q-network (θ^Q)
Soft replace $\theta^{Q'}$
Q'
Target Q-network ($\theta^{Q'}$)
Maximizing Q
Policy gradient w.r.t. θ^μ
Policy network (θ^μ)
Soft replace $\theta^{\mu'}$
Target policy network ($\theta^{\mu'}$)
Policy gradient w.r.t. a
a
a'
Noise
$a = \mu(o_t)$
Action a_t
o_t
Environment
o_t a_t r_t o_{t+1}
Mini-batch $\{o, a, r, o\_\}$
Mini-batch $\{o\}$
Experience replay buffer

Fig. 1 Actor-critic architecture of deep deterministic policy gradient algorithm

The DDPG algorithm also maintains a target Q-network (Q') and a target policy network (μ'). They are the time-delayed copies of the Q-network and the policy network, respectively. The target Q-network is parameterized with $\theta^{Q'}$, and the target policy network is parameterized with $\theta^{\mu'}$. Figure 1 shows the actor-critic architecture of the algorithm that includes the four networks, i.e., a Q-network, a deterministic policy network, a target Q-network, and a target policy network.

In the neural-network implementation, the updated Q-value ($\tilde{Q}$) according to the Bellman equation (11) is given by

$$\tilde{Q}_t = r(o_t, a_t) + \gamma Q'(o_{t+1}, \mu'(o_{t+1} \mid \theta^{\mu'}) \mid \theta^{Q'}). \tag{14}$$

The Q-network (Q) is updated by minimizing the temporal-difference (TD) loss as

$$\min \quad L = \sum \left[Q(o_t, a_t \mid \theta^Q) - \tilde{Q}_t \right]^2 \tag{15}$$

where $\sum$ denotes the average sum over a mini-batch.

With the actor's policy function μ, the objective is to maximize the expected return, i.e., the expected cumulative discounted reward, as

$$\max \quad J = \mathbb{E}\left[Q(o_t, a)|_{a=\mu(o_t)} \right]. \tag{16}$$

Because the action space is continuous, the function $Q(o_t, a)$ is differentiable with respect to the action argument a. As the policy function is differentiable, we can apply the chain rule and take the derivative of the objective function with respect to the policy parameter. Therefore, the policy network (μ) is updated by using the sampled policy gradient as

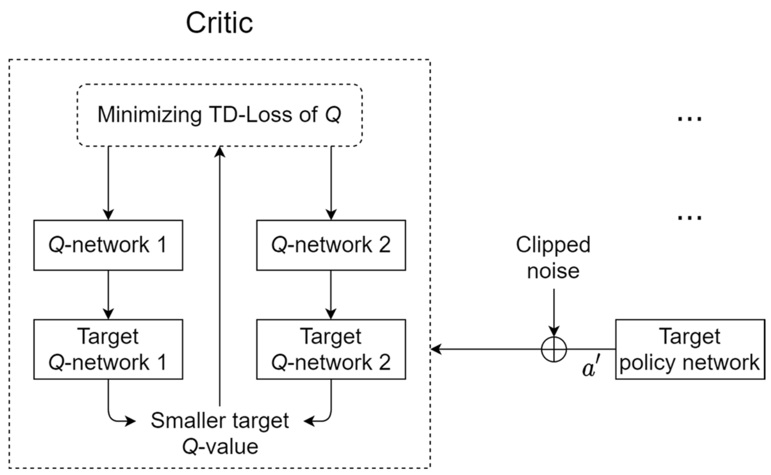

Fig. 2 Part of structure of twin delayed deep deterministic policy gradient algorithm

$$\nabla_{\theta^{\mu}} J \approx \sum \nabla_a Q(o_t, a \mid \theta^Q)|_{a=\mu(o_t)} \nabla_{\theta^{\mu}} \mu(o_t \mid \theta^{\mu}). \tag{17}$$

The two target networks slowly track the learned parameters of the *Q*-network and the policy network. The updates follow the soft-replacement rules of Polyak averaging as

$$\begin{aligned} \theta^{Q'} &\leftarrow \rho\theta^{Q} + (1-\rho)\theta^{Q'} \\ \theta^{\mu'} &\leftarrow \rho\theta^{\mu} + (1-\rho)\theta^{\mu'} \end{aligned} \tag{18}$$

where $\rho (0 < \rho \ll 1)$ controls the update rate to be slow to improve the stability of learning.

5.2 Twin delayed deep deterministic policy gradient algorithm

The DDPG algorithm can be fragile when the learned *Q*-function starts to overestimate the *Q*-value. The algorithm will exploit the errors in the *Q*-function, which can trigger policy breaking. The TD3 algorithm extends the DDPG algorithm to address this issue.

The TD3 algorithm uses two separate *Q*-networks in the critic (shown in Fig. 2). It learns two *Q*-functions and uses the smaller of the two *Q*-values to form the target in the Bellman error loss function. The low value provides a more stable approximation, thus improving the algorithm stability.

The TD3 algorithm delays the update of the policy network. In other words, it updates the policy network less frequently than the *Q*-network. With the interaction between the actor (policy network) and the critic (*Q*-network), errors may accumulate, leading to a decline in learning performance. The delayed update of the policy network allows the *Q*-network to become more stable with smaller errors before the value estimate is used to update the policy network. The value estimate with lower variance can help produce a better policy.

Deterministic policy algorithms tend to generate target *Q*-values with high variance when updating the critic. This is due to overfitting the spikes in the value estimate. The TD3 algorithm handles this by adding random noise to the target action in the actor. The range of noise is clipped to keep the target action close to the ideal one. By adding noise

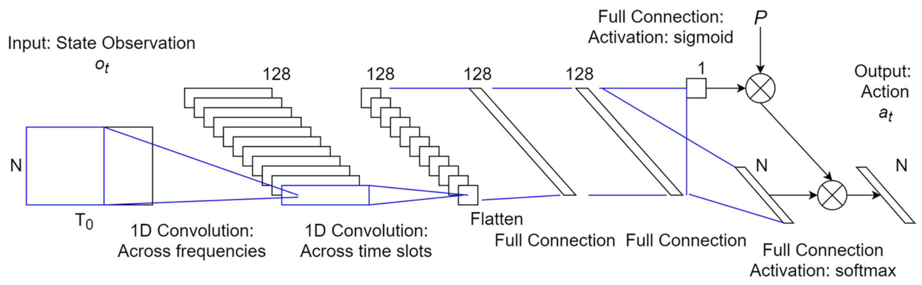
Input: State Observation
o_t
N
T_0
128
128
128
128
1
1D Convolution: Across frequencies
1D Convolution: Across time slots
Flatten
Full Connection
Full Connection
Full Connection: Activation: sigmoid
P
N
Full Connection Activation: softmax
Output: Action a_t
N

Fig. 3 Neural-network architecture of the policy network

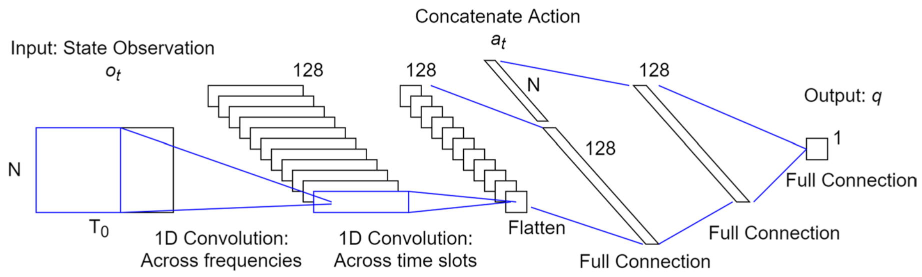

Fig. 4 Neural-network architecture of the *Q*-network

to the target action, the *Q*-value is smoothed out with changes in the action. In return, the policy is less likely to be affected by the *Q*-function errors.

5.3 Deep neural networks for actor and critic

Deep neural networks are used as function approximators in the actor and the critic. Figure 3 shows the network architecture for the policy network and the target policy network of the actor. The state observation o_t with dimension $N \times T_0$ is the input of the network. The input passes through a one-dimensional (1D) convolutional layer. The convolution filter covers the entire frequency span and part of the time span, i.e., the kernel size is less than T_0. There are 128 such filters, so the output depth of this layer is 128. The layer is followed by another 1D convolutional layer. The convolution filter covers the entire time span. These layers excavate and exploit the correlation of the input data in both frequency and time. In the time domain, an LSTM layer could be an alternative [32]. However, the LSTM layer has higher sequential operation complexity. Using the 1D convolutional layer in the time domain instead of the LSTM results in similar learning performance with less complexity.

Next, the output of the layer is flattened and consecutively passes through two fully connected layers, each with 128 nodes. The activation functions of these layers are rectified linear unit (ReLU). After that, the data pass through two fully connected layers in parallel. The first one has N output nodes and the activation function is softmax. It gives the distribution of the transmission power on the N frequency channels. The second one has one output node and the activation function is sigmoid. It handles the situation where the agent expects no available channel and decides not to transmit at all. The network output is the product of these two layer-outputs, multiplied by the transmission power limit P. This ensures that the agent's total transmission power on the frequency channels never exceeds the power limit P. The policy network is trained by maximizing the expected cumulative discounted reward, i.e., the Q-function. The parameters of the target policy network are updated by Polyak averaging the policy-network parameters.

Figure 4 shows the network architecture for the Q-network and the target Q-network of the critic. Similar to the policy network, the state observation o_t is input to the Q-network that consecutively passes through two 1D convolutional layers. The output of the layer is flattened and passes through a fully connected layer with 128 nodes. At this layer, the data are concatenated with the action a_t (transmission power on N channels) that comes from the actor. The concatenated data go through another fully connected layer with 128 nodes. The activation functions of these fully connected layers are ReLU. Finally, the data pass through a fully connected layer with one output node. No activation function is used. It gives the Q-value. The Q-network is trained by minimizing the temporal-difference loss of the Q-value. For the training, the critic needs mini-batches of the tuples (observation, action, reward, next observation). The parameters of the target Q-network are updated by Polyak averaging the Q-network parameters.

5.4 Neural network training

The neural networks of the DDPG and the TD3 algorithms are trained to produce a deterministic policy. The algorithms are off-policy algorithms. At training time, we add noise to the action to perform exploration. With a continuous action space, the exploration is given by

$$a_t = \mu(o_t \mid \theta^{\mu}) + w \tag{19}$$

where w is the noise sampled from a noise process. At the beginning of training, the variance of the noise is large so that the agent can try very different actions to establish a useful learning experience. When the neural network learns better and better during training, the noise variance will gradually decrease.

In dynamic spectrum access and sharing, the action is the transmission power allocated on the frequency channels $a_t = \{p_{n,t}\}_{n\in\mathcal{N}}$. The agent explores new actions by adding uncorrelated zero-mean Gaussian noise $\{w_n\}_{n\in\mathcal{N}}$ to the power levels. That is

$$p_{n,t} = \max[\hat{p}_{n,t} + w_n, 0], \quad \forall n \in \mathcal{N} \tag{20}$$

where $\hat{p}_{n,t}$ is the power level from the policy network. The operation $\max[\cdot, 0]$ is to ensure that the transmission power is non-negative. If the power levels obtained through exploration exceed the transmission power limit P, they will be trimmed down according to

$$\frac{P}{\sum_{n\in\mathcal{N}} p_{n,t}} p_{n,t} \Rightarrow p_{n,t}, \quad \forall n \in \mathcal{N}. \tag{21}$$

To improve exploration, in the beginning, the agent can take actions that are randomly sampled from a uniform distribution over valid actions. During regular training, the agent gradually reduces the variance of the noise w_n that is added to $\hat{p}_{n,t}$. With endless training as in reinforcement learning, exploration eventually fades away and the new action follows the deterministic policy.

Experience replay is applied in the learning algorithms. The algorithms use a replay buffer to store the experience tuples (observation, action, reward, next observation). The replay buffer is large enough to contain a wide range of previous experiences. When the replay buffer is full, the oldest tuples are pushed out. When training the neural networks, we randomly extract mini-batches of experience from the replay buffer to update the network parameters. Experience replay breaks the correlation in successive updates, thus stabilizing learning.

6 Multi-agent deep reinforcement learning for multiple secondary users

6.1 Cooperative tasks of secondary users for dynamic spectrum access and sharing

When K secondary users access the spectrum simultaneously, we encounter a multi-agent learning problem. The agents are the secondary users who observe the system state of channel usage by receiving conflict warnings from the primary users. Under the premise of avoiding transmission conflicts with the primary users, the agents selectively transmit on the available frequency channels so that they do not interfere with each other excessively.

At time slot t, the kth agent ($k \in \mathcal{K}$) adjusts the transmission power $\{p_{k,n}\}_{n\in\mathcal{N}}$ to boost the information rate q_k given in (2). Multiple agents adjust the transmission power in a distributed manner, but the goal is to maximize the sum information rate of the secondary users. The transmission problem of the K agents can be formulated as

$$\begin{array}{rl} \underset{\{p_{k,n}\}_{k\in\mathcal{K},n\in\mathcal{N}}}{\text{maximize}} & \sum_{k\in\mathcal{K}} q_k \\ \text{subject to} & \sum_{n\in\mathcal{N}} p_{k,n} \le P, \quad \forall k \\ & \text{SINR}_{k,m} = 0, \quad m \in \mathcal{M}, \quad \forall k \end{array} \tag{22}$$

where $\mathcal{M}(\mathcal{M} \subseteq \mathcal{N})$ is the set of indexes of frequency channels that are currently in conflict with the primary users.

In a cooperative game, the agents have the same reward function and they learn to maximize the common discounted return. The agents use deep reinforcement learning to deal with the transmission problem (22). Each agent processes its own learning, but the agents share a unified reward to encourage cooperation. The unified reward combines the individual rewards of the agents. It is given by

$$r_t = \sum_{k\in\mathcal{K}} r_{k,t} = \sum_{k\in\mathcal{K}} q_{k,t} - \beta \sum_{k\in\mathcal{K}} \sum_{m\in\mathcal{M}_t} p_{k,m,t} \tag{23}$$

where $q_{k,t}$ is the information rate of the kth agent at time slot t, which depends on the received SINRs on the available frequency channels. Set $\mathcal{M}_t$ indicates the conflicting frequency channels at time slot t. The kth agent transmits on the mth frequency channel with power $p_{k,m,t}$.

When the K agents transmit at time slot t with power $\{p_{k,n,t}\}_{k\in\mathcal{K},n\in\mathcal{N}}$, they measure the received SINRs and listen to warnings from the primary users. These warnings can be triggered by any agent's transmission on the channels that are in conflict with the primary users. Each agent communicates its detected set $\mathcal{M}_{k,t}$ to others. By adopting the union of sets $\{\mathcal{M}_{k,t}\}_{k\in\mathcal{K}}$ reported by the agents, i.e., $\mathcal{M}_t = \bigcup_{k\in\mathcal{K}} \mathcal{M}_{k,t}$, the agents can get a more accurate set $\mathcal{M}_t$ for the unified reward. To some extent, this enhances the detection of primary spectrum usage through secondary user cooperation [33, 34]. Different agents may probe different frequency channels. Multiple agents monitor warnings together. The kth agent sets the received SINR to zero on conflicting channels and calculates the information rate $q_{k,t}$ according to (2) and the individual reward $r_{k,t}$ according to (8). Then, the agent exchanges the value of the individual reward with others. In the end, each agent obtains the unified reward $r_t = \sum_{k\in\mathcal{K}} r_{k,t}$.

Multiple agents interact with the communication environment and make strategic decisions that adapt to the spectrum and other agents' activities. Each agent uses an actor-critic deep reinforcement learning mechanism to find a policy that tends to maximize the common cumulative discounted reward. According to this policy, the agent specifies the frequency channels it uses and the transmission power levels on these channels. Algorithm 1 summarizes the distributed learning algorithm with multi-agent cooperation. In Algorithm 1, we describe the learning process of the kth agent.

Algorithm 1 : The kth agent's learning algorithm for dynamic spectrum access

```
1. Initialize the experience replay buffer.
2. Randomly initialize parameters $\theta^Q$ of the $Q$-network $Q(o_{k,t}, a_{k,t} \mid \theta^Q)$ and parameters $\theta^\mu$ of the policy network $a_t = \mu(o_{k,t} \mid \theta^\mu)$.
3. Initialize the target networks $Q'$ and $\mu'$ with parameters $\theta^{Q'} \leftarrow \theta^Q$ and $\theta^{\mu'} \leftarrow \theta^\mu$.
4. Start at time slot $t = 0$. Obtain observation $o_{k,1}$ of the system state over $T_0$ consecutive time slots.
5. for $t = T_0$ to forever do
6.   if Replay buffer is full then
7.     Select action $a_{k,t} = \mu(o_{k,t} \mid \theta^\mu) + w_t$, where $w_t$ is the exploration noise. Execute action $a_{k,t}$. Slightly reduce the noise variance.
8.   else
9.     Randomly select frequency channels and transmit within the power limit.
10.  end if
11.  Measure $\{\mathrm{SINR}_{k,n,t}\}_{n\in\mathcal{N}}$ and detect $\mathcal{M}_{k,t}$ with warnings of channel conflict. Broadcast $\mathcal{M}_{k,t}$ in the secondary user network.
12.  Determine $\mathcal{M}_t = \bigcup_{k=1}^{K} \mathcal{M}_{k,t}$. Set $\mathrm{SINR}_{k,m,t} = 0$ if $m \in \mathcal{M}_k$. Calculate information rate $q_{k,t}$.
13.  Calculate individual reward $r_{k,t}$. Exchange individual reward with other agents. Calculate unified reward $r_t$.
14.  Obtain new observation $o_{k,t+1}$.
15.  Refresh the replay buffer with the new transition tuple $(o_{k,t}, a_{k,t}, r_t, o_{k,t+1})$.
16.  if Replay buffer is full then
17.    Randomly sample a mini-batch from the replay buffer.
18.    Set $\tilde{Q}_{k,t} = r(o_{k,t}, a_{k,t}) + \gamma Q'(o_{k,t+1}, \mu'(o_{k,t+1} \mid \theta^{\mu'}) \mid \theta^{Q'})$.
19.    Update $\theta^Q$ in the $Q$-network by minimizing the TD-loss $L = \sum \left[Q(o_{k,t}, a_{k,t} \mid \theta^Q) - \tilde{Q}_{k,t}\right]^2$.
20.    Update $\theta^\mu$ in the policy network with the sampled policy gradient $\nabla_{\theta^\mu} J \approx \sum \nabla_a Q(o_{k,t}, a \mid \theta^Q)|_{a=\mu(o_{k,t})} \nabla_{\theta^\mu} \mu(o_{k,t} \mid \theta^\mu)$.
21.    Update the parameters of the target $Q$-network and the target policy network by Polyak averaging $\theta^{Q'} \leftarrow \rho\theta^Q + (1-\rho)\theta^{Q'}$, $\theta^{\mu'} \leftarrow \rho\theta^\mu + (1-\rho)\theta^{\mu'}$.
22.  end if
23. end for
```

6.2 Coordination in multi-agent deep reinforcement learning

When considering multi-user dynamic spectrum access and sharing as cooperative multi-agent deep reinforcement learning, we specify a unified reward, hence a common cumulative discounted reward. In a distributed approach, each agent is an independent decision-maker. The agents have local Q-functions. Compared with the ideal centralized method, the distributedly generated policies are suboptimal.

Besides the rewards, the agents can exchange other types of information for different degrees of coordination. The agents can exchange both rewards and state observations with each other. The signaling overhead in the secondary user network is moderate because the reward is scalar and the state observation is ternary. The individual state observations $\{o_{k,t}\}_{k\in\mathcal{K}}$ may be partial observation, i.e., they may contain many zero elements. The aggregated state observation o_t is the consensus of the individual state observations such that the number of zero elements is reduced. Referring to Figs. 3 and 4, we see that the aggregated state observation o_t can be the same input to the actor and the critic of every agent. Therefore, the agents can better learn the temporal and spectral correlations of the spectrum.

The agents learn simultaneously in multi-agent deep reinforcement learning. Each agent can further explore and adapt to the behaviors of other agents. The agents' choices of actions must be consistent with each other to achieve the desired outcome. Toward this end, the agents can exchange rewards, state observations, and action choices. Referring to Fig. 4, in the third hidden layer of the Q-network, we concatenate the layer output with action vectors $\{a_{k,t}\}_{k\in\mathcal{K}}$ not only from this agent's actor but from other agents' as well. Each agent has a local actor-critic learning mechanism and updates the network parameters independently. However, by critiquing the actions taken by all the agents, the agent can improve its policy and make the overall result closer to the optimal joint-action solution of the ideal centralized method.

6.3 Scalability and stability

The algorithm of multi-agent deep reinforcement learning for dynamic spectrum access and sharing has a high degree of scalability. First, each agent exploits actor-critic deep reinforcement learning for continuous state-action space. It alleviates the curse of dimensionality caused by the exponential growth of discrete state-action space in the number of state and action variables. Second, multiple secondary users are autonomous agents who learn to deal with spectrum issues independently. Since the agents perform deep reinforcement learning in a distributed manner, they do not encounter a joint state-action space like that in a centralized approach. Therefore, the learning algorithm complexity does not scale with the increase in the number of agents. There is moderate signaling overhead among the agents. As the degree of agent coordination increases, signaling overhead will increase. Third, when one or more agents leave the multi-agent system, the remaining agents can continue to operate normally. When new agents are inserted into the system, they can quickly join the cooperation of multi-agent reinforcement learning. Therefore, the algorithm is inherently robust.

Non-stationarity appears in multi-agent deep reinforcement learning when all agents learn simultaneously [35]. The agents explore the situation by learning the environment (the spectrum usage of the primary users) and the behaviors of other agents (the transmission actions of other secondary users). In dynamic spectrum access and sharing, the transmission of secondary users will not affect the primary spectrum usage. In other words, the behaviors of the agents will not have a significant impact on the environment. By learning the statistics of the changing environment, the agents can converge to an equilibrium with some stability. At the equilibrium, the secondary users do not transmit on some channels that contribute little to the sum information rate but are very likely to cause conflict with the primary users. The stability of a multi-agent learning process is important because a more stable agent can help other agents learn quickly. The agents then adapt to changes in the environment and each other's behaviors. A balance of stability and adaptability can be achieved for good performance.

7 Experimental methods

We conduct experiments using a wireless communication network in which primary and secondary users of the spectrum share multiple frequency channels. The wireless transceivers use the Wireless Open Access Research Platform (WARP) v3 boards with the FMC-RF-2X245 modules and the Universal Software Radio Peripheral (USRP) X310 and N210 boards with the CBX daughterboards. Each secondary user's wireless transceiver is connected to a GPU-accelerated computer. The GPUs are NVIDIA GeForce RTX 2080 Ti supported by the CUDA toolkit. Learning algorithms are programmed using Python/Tensorflow and executed on the dedicated computers. An N9030A PXA Signal Analyzer is used to measure the received SINR. Figure 5 shows the experiment setup in the laboratory.

In the experiments, each secondary user has a transmission power limit of $P = 30$ mW (14.77 dBm). The transmission power can be allocated across $N = 14$ frequency channels in the 2.4 GHz range. The channels are spaced 5 MHz apart from each other. The average channel gain $|\bar{h}|$ of the secondary users is measured at about −61.29 dB. The receiver noise in one frequency channel is $WN_0 = 1$ nW (−60 dBm). Channel usage by the primary users is arbitrary, but the channel usage and switching pattern is periodic over the long term. A channel conflict warning is issued when the primary user experiences excessive interference from the secondary users. In the experiments, this is usually caused by the secondary users transmitting more than 100 μW (−10 dBm) in any frequency channel occupied by the primary user. The secondary users do not know the primary users' channel usage and switching pattern. They also do not know the channel gains $\{|h|\}$ of user and interference channels among the secondary users.

The secondary users implement the twin delayed deep deterministic policy gradient (TD3) algorithm to dynamically access and share the spectrum. In Table 1, we list the parameters of the TD3 algorithm. The discount rate of the cumulative discounted

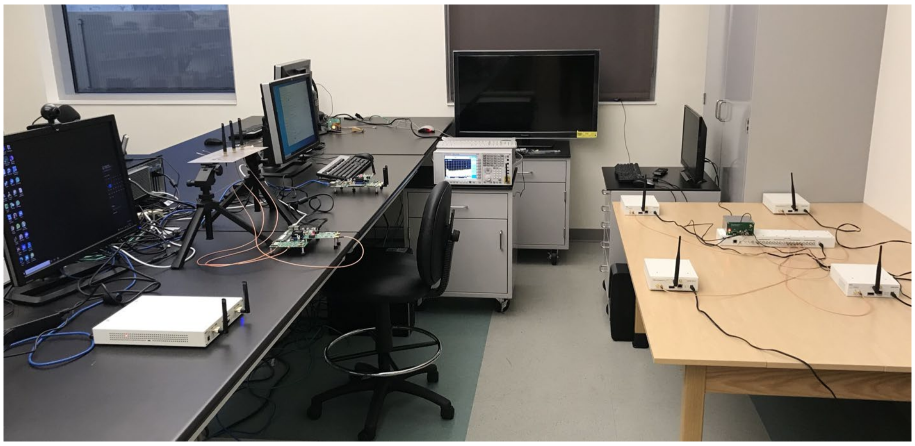

Fig. 5 Experiment setup in the laboratory

Table 1 Parameters of the TD3 algorithm implemented by the secondary user

Discount rate γ of cumulative reward	0.5
Learning rate of actor	0.0001
Learning rate of critic	0.0003
Update parameter of target networks ρ	0.001
TD3 delayed update of actor	1 actor update for 10 critic updates
Experience replay buffer size	100,000
Mini-batch size	128
State observation time span T_0	32 time slots
Reward coefficient β	0.05 (bit/s/Hz) / mW
Exploration noise w added to the action, decreasing during training	Start at $\sigma_w = 10$ mW $\sigma_{w,t+1} = 0.99995\sigma_{w,t}$

reward is $\gamma = 0.5$. The learning rate of the actor is 0.0001, and the learning rate of the critic is 0.0003. The lower learning rate of the actor makes the actor converge slower. The update parameter of the target networks is $\rho = 0.001$. The TD3 algorithm takes one policy update for every 10 Q-function updates. The experience replay buffer can store 100,000 tuples of data (over 100,000 time slots). The size of a mini-batch is 128 tuples. The state observation spans $T_0 = 32$ time slots, and the first 1D convolutional layer in the policy network and the Q-network has a kernel size of 24. The reward coefficient β that balances maximizing information rate and minimizing transmission conflict is set empirically at 0.05 (bit/s/Hz)/mW. We implement the exploration of the TD3 algorithm by adding uncorrelated zero-mean Gaussian noise to the deterministic actions. The standard deviation of the noise starts at $\sigma_w = 10$ mW. The noise standard deviation decreases during training with $\sigma_{w,t+1} = 0.99995\sigma_{w,t}$.

The secondary user first randomly selects the frequency channels and transmits within the power limit. It observes the system state, takes random actions at time slots, and calculates rewards to fill the replay buffer. After the replay buffer is full, the secondary user iterates the TD3 algorithm to adjust its transmission power levels on the frequency channels. It continuously collects new data and updates the replay buffer. The secondary user samples mini-batches from the replay buffer and learns to alleviate channel conflict with the primary users while increasing its information rate.

8 Results and discussion

Figures 6, 7, and 8 show the results when only one secondary user (User 1, User 2, and User 3, respectively) is active in the wireless communication network. Each figure shows the number of channels that conflict with the primary users and the secondary user's information rate over learning iterations in the number of time slots. The curves in the figures are the moving average of 1000 data points. The performance of dynamic spectrum access and sharing is compared with the baseline case. In the baseline case, the secondary user is unaware of the channel usage status and distributes its transmission power evenly over the frequency channels. The information rate is also compared with the rate under ideal conditions when the secondary user is fully aware of channel availability and channel gains. The secondary user transmits on the available channels with the "water-filling" power allocation to maximize its information rate. The results show

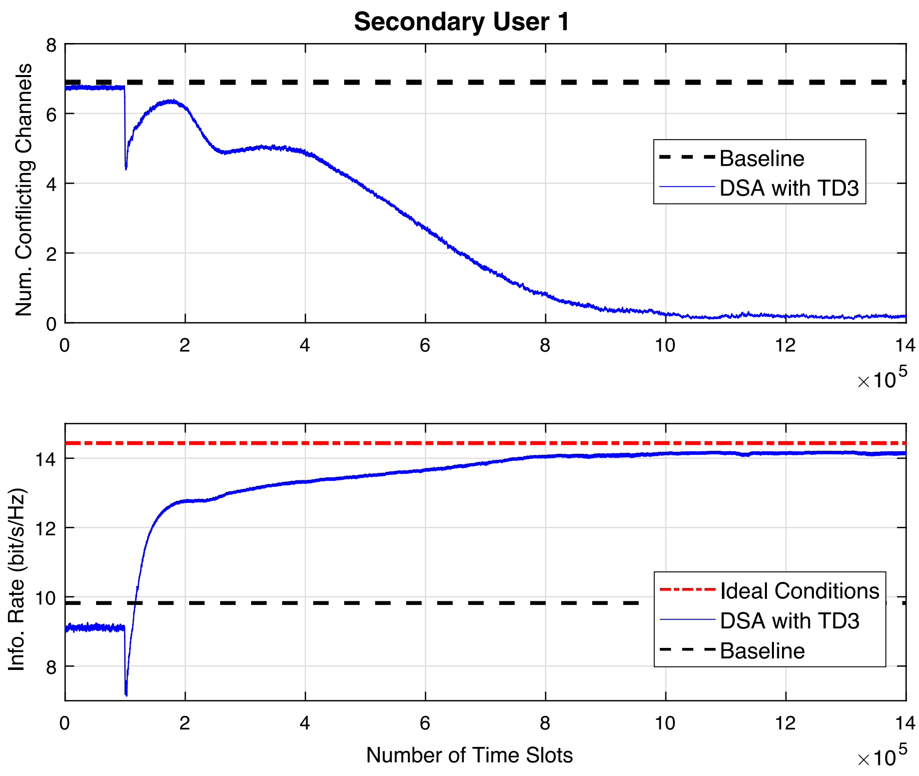

Fig. 6 Secondary User 1 is active alone in the secondary user network. It performs dynamic spectrum access and sharing using the TD3 algorithm. Top: The number of frequency channels that conflict with the primary users over time slots of training iterations. Bottom: The information rate of the secondary user over time slots of training iterations

that the secondary user can learn an almost optimal policy for dynamic spectrum access and sharing using the TD3 algorithm. Over time, the channel conflict with the primary users can be avoided, and the secondary user's information rate approaches that under the ideal conditions.

In the experiments, we also test the scenario where $K = 3$ secondary users are active simultaneously in the wireless communication network. They access and share the spectrum with the primary users and aim to alleviate channel conflict while maximizing their sum information rate. They implement the multi-agent deep reinforcement learning with the TD3 algorithm. The agents coordinate by exchanging the information of rewards, the information of rewards and state observations, or the information of rewards, state observations, and actions. Figure 9 shows the number of channels that conflict with the primary users as the secondary users iterate the multi-agent learning algorithm over time in the number of time slots. As the degree of coordination increases, the agents can adapt faster to the spectrum environment and avoid most channel conflicts.

Figure 10 compares the sum information rate achieved through the proposed dynamic spectrum access and sharing with two baseline cases. In the first baseline case, the secondary users do not know channel availability or channel gains. Nevertheless, they can cooperate to be assigned exclusive frequency channels and do not interfere with each other. Each secondary user distributes transmission power evenly

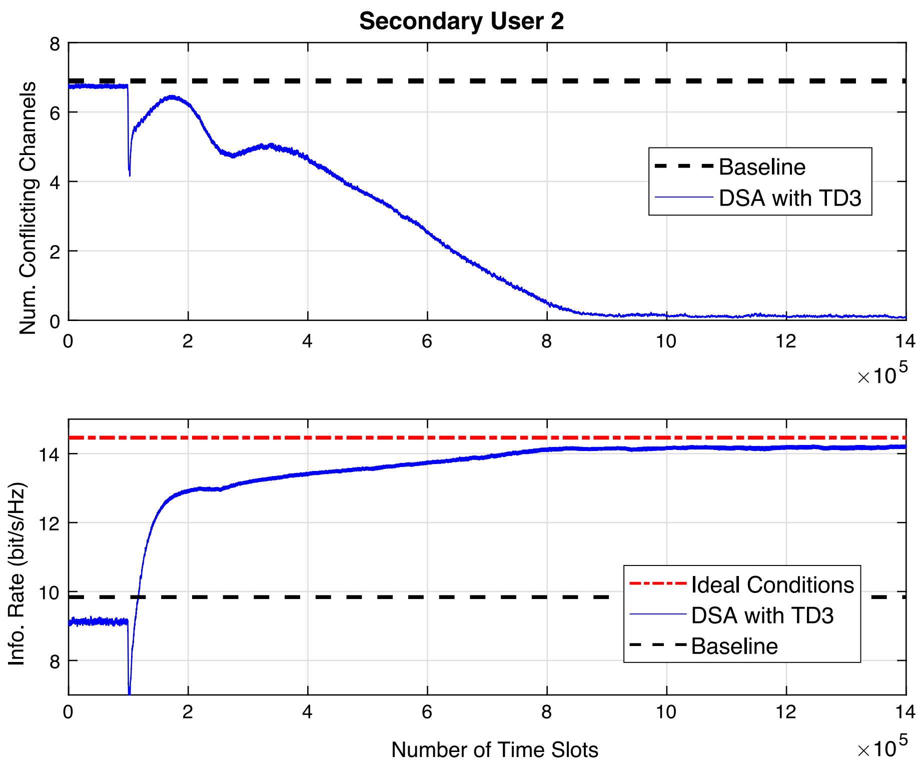

Fig. 7 Secondary User 2 is active alone in the secondary user network. It performs dynamic spectrum access and sharing using the TD3 algorithm. Top: The number of frequency channels that conflict with the primary users over time slots of training iterations. Bottom: The information rate of the secondary user over time slots of training iterations

on its exclusive channels. In the second baseline case, each secondary user distributes power evenly on all frequency channels. In the figure, we also plot the sum information rate that can be achieved under ideal conditions as an upper bound. In the ideal situation, the secondary users are fully aware of the spectrum environment and can transmit on available frequency channels. They also perfectly know the user and interference channels among the secondary users and can negotiate the best allocation of the available channels. Each secondary user then transmits on the interference-free channels with the "watering-filling" power distribution. As revealed in the figure, through multi-agent deep reinforcement learning, the secondary users can obtain a sum information rate that exceeds the benchmark of the baseline cases. As the degree of coordination increases, the achieved sum information rate gets closer to the upper bound.

9 Conclusion

A practical method for dynamic spectrum access and sharing is proposed and experimentally verified in a wireless communication network. The secondary users of the spectrum implement deep reinforcement learning algorithms to avoid or reduce interference to the primary users while maximizing the secondary user's information rate. A framework of actor-critic deep deterministic policy gradient algorithm is tailored for the task, and the

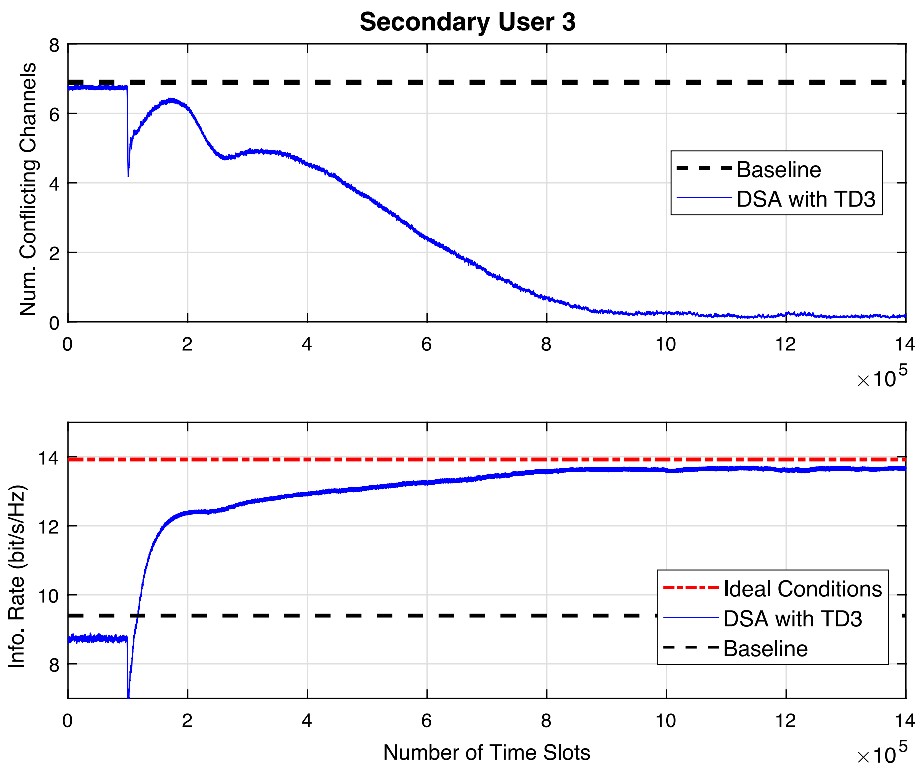

Fig. 8 Secondary User 3 is active alone in the secondary user network. It performs dynamic spectrum access and sharing using the TD3 algorithm. Top: The number of frequency channels that conflict with the primary users over time slots of training iterations. Bottom: The information rate of the secondary user over time slots of training iterations

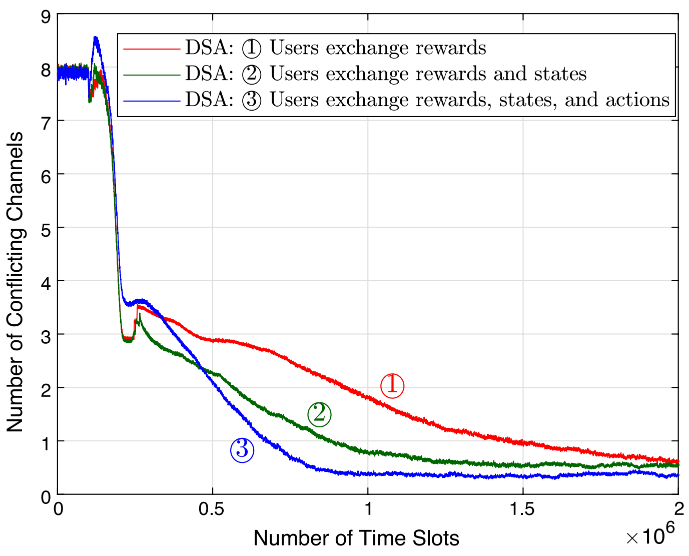

Fig. 9 Three secondary users perform dynamic spectrum access and sharing using multi-agent deep reinforcement learning with TD3. Users exchange rewards, exchange rewards and states, or exchange rewards, states and actions. The figure shows the number of frequency channels that conflict with the primary users over time slots of training iterations

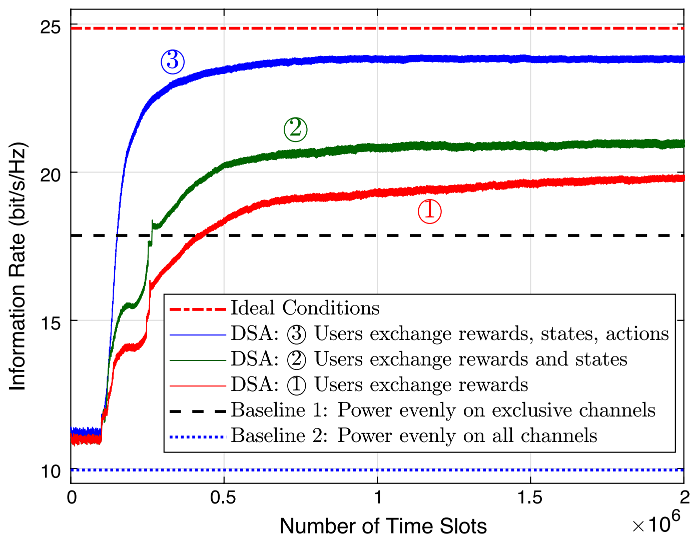

Fig. 10 Three secondary users perform dynamic spectrum access and sharing using multi-agent deep reinforcement learning with TD3. Users exchange rewards, exchange rewards and states, or exchange rewards, states and actions. The figure shows the sum information rate of the secondary users over time slots of training iterations

deep neural networks are designed. The secondary users learn the preferred distribution of transmission power on the frequency channels through limited interaction with the communication environment. Each secondary user exploits the historical warnings of channel conflict and the received SINRs to adapt to rapid switching of channel usage by the primary users. Multiple secondary users implement multi-agent deep reinforcement learning with various degrees of coordination among the secondary users. The communication overhead in the secondary user network is moderate. The algorithms are effective, enabling the secondary users to quickly establish transmission policies that achieve good spectrum utilization. For future work, we plan to investigate the dynamic spectrum access and sharing problem with more random channel usage and switching patterns. The switching of the channel usage state, for example, can be modeled as a Markov chain with a state transition probability of less than one. Additionally, we plan to investigate the robustness and convergence speed of the TD3 algorithm.

Abbreviations

DDPG	Deep deterministic policy gradient
TD3	Twin delayed deep deterministic policy gradient
TD-loss	Temporal-difference loss
LSTM	Long short-term memory
1D	One-dimensional
ReLU	Rectified linear unit
GPU	Graphics processing unit
CUDA	Compute unified device architecture

Author contributions

LD conceived of the study, carried out the system design, developed the algorithms, and drafted the manuscript. YQ participated in the design and coordination of the experiments and helped analyze the experimental results. YX participated in the algorithm programming and helped test the algorithms. All authors participated in method development and implementation. All authors read and approved the final manuscript.

Funding

This work was supported in part by research grants from Baylor University and by funds from Arvon Technologies.

Availability of data and materials
Data sets used and analyzed during the current study are available from the corresponding author upon request.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 14 February 2022 Accepted: 21 April 2022
Published online: 04 June 2022

References

1. X. Xing, T. Jing, W. Cheng, Y. Huo, X. Cheng, Spectrum prediction in cognitive radio networks. IEEE Trans. Wirel. Commun. **20**(2), 90–96 (2013)
2. H. Eltom, S. Kandeepan, R.J. Evans, Y.C. Liang, B. Ristic, Statistical spectrum occupancy prediction for dynamic spectrum access: a classification. EURASIP J. Wirel. Commun. Netw. **29**, 1–17 (2018)
3. G. Ding, Y. Jiao, J. Wang, Y. Zou, Q. Wu, Y. Yao, L. Hanzo, Spectrum inference in cognitive radio networks: algorithms and applications. IEEE Commun. Surv. Tutor. **20**(1), 150–182 (2018)
4. Y. Zhao, Z. Hong, Y. Luo, G. Wang, L. Pu, Prediction-based spectrum management in cognitive radio networks. IEEE Syst. J. **12**(4), 3303–3314 (2018)
5. V. Nguyen, O. Shin, Cooperative prediction-and-sensing-based spectrum sharing in cognitive radio networks. IEEE Trans. Cogn. Commun. Netw. **4**(1), 108–120 (2018)
6. V. Mnih, K. Kavukcuoglu, D. Silver et al., Human-level control through deep reinforcement learning. Nature **518**, 529–533 (2015)
7. D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, M. Riedmiller, Deterministic policy gradient algorithms. In: *31st International Conference on Machine Learning (ICML)* (2014)
8. T.P. Lillicrap, J.J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning. In: *International Conference on Learning Representations (ICLR)* (2016)
9. Y. Duan, X. Chen, R. Houthooft, J. Schulman, P. Abbeel, Benchmarking deep reinforcement learning for continuous control. In: *33rd International Conference on International Conference on Machine Learning (ICML)*, vol. 48, pp. 1329–1338 (2016)
10. S. Fujimoto, H. van Hoof, D. Meger, Addressing function approximation error in actor-critic methods. In: *Proceedings of the 35th International Conference on Machine Learning*, vol. 80, pp. 1587–1596. PMLR, Stockholmsmässan, Stockholm (2018)
11. S. Wang, H. Liu, P.H. Gomes, B. Krishnamachari, Deep reinforcement learning for dynamic multichannel access in wireless networks. IEEE Trans. Cogn. Commun. Netw. **4**(2), 257–265 (2018)
12. O. Naparstek, K. Cohen, Deep multi-user reinforcement learning for distributed dynamic spectrum access. IEEE Trans. Wirel. Commun. **18**(1), 310–323 (2019)
13. H.-H. Chang, H. Song, Y. Yi, J. Zhang, H. He, L. Liu, Distributive dynamic spectrum access through deep reinforcement learning: a reservoir computing-based approach. IEEE Internet Things J. **6**(2), 1938–1948 (2019)
14. H.-H. Chang, L. Liu, Y. Yi, Deep echo state Q-network (DEQN) and its application in dynamic spectrum sharing for 5G and beyond. IEEE Trans. Neural Netw. Learn. Syst. **33**(3), 929–939 (2022)
15. C. Zhong, Z. Lu, M.C. Gursoy, S. Velipasalar, A deep actor-critic reinforcement learning framework for dynamic multichannel access. IEEE Trans. Cogn. Commun. Netw. **5**(4), 1125–1139 (2019)
16. J. Huang, Y. Yang, G. He, Y. Xiao, J. Liu, Deep reinforcement learning-based dynamic spectrum access for D2D communication underlay cellular networks. IEEE Commun. Lett. **25**(8), 2614–2618 (2021)
17. J. Huang, Y. Yang, Z. Gao, D. He, D.W.K. Ng, Dynamic spectrum access for D2D-enabled Internet-of-things: a deep reinforcement learning approach. IEEE Internet Things J. **9** (2022)
18. A. Doshi, S. Yerramalli, L. Ferrari, T. Yoo, J.G. Andrews, A deep reinforcement learning framework for contention-based spectrum sharing. IEEE J. Sel. Areas Commun. **39**(8), 2526–2540 (2021)
19. Z. Guo, Z. Chen, P. Liu, J. Luo, X. Yang, X. Sun, Multi-agent reinforcement learning-based distributed channel access for next generation wireless networks. IEEE J. Sel. Areas Commun. **40**(5), 1587–1599 (2022)
20. X. Li, J. Fang, W. Cheng, H. Duan, Z. Chen, H. Li, Intelligent power control for spectrum sharing in cognitive radios: a deep reinforcement learning approach. IEEE Access **6**, 25463–25473 (2018)
21. Y.S. Nasir, D. Guo, Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks. IEEE J. Sel. Areas Commun. **37**(10), 2239–2250 (2019)
22. H. Zhang, N. Yang, W. Huangfu, K. Long, V.C.M. Leung, Power control based on deep reinforcement learning for spectrum sharing. IEEE Trans. Wirel. Commun. **19**(6), 4209–4219 (2020)
23. F. Meng, P. Chen, L. Wu, J. Cheng, Power allocation in multi-user cellular networks: deep reinforcement learning approaches. IEEE Trans. Wirel. Commun. **19**(10), 6255–6267 (2020)
24. H. Ye, G.Y. Li, B.-H.F. Juang, Deep reinforcement learning based resource allocation for V2V communications. IEEE Trans. Veh. Technol. **68**(4), 3163–3173 (2019)
25. L. Liang, H. Ye, G.Y. Li, Spectrum sharing in vehicular networks based on multi-agent reinforcement learning. IEEE J. Sel. Areas Commun. **37**(10), 2282–2292 (2019)
26. Y. Xu, C. Yang, M. Hua, W. Zhou, Deep deterministic policy gradient (DDPG)-based resource allocation scheme for NOMA vehicular communications. IEEE Access **8**, 18797–18807 (2020)

27. Z. Li, C. Guo, Multi-agent deep reinforcement learning based spectrum allocation for D2D underlay communications. IEEE Trans. Veh. Technol. **69**(2), 1828–1840 (2020)
28. J. Tan, Y.-C. Liang, L. Zhang, G. Feng, Deep reinforcement learning for joint channel selection and power control in D2D networks. IEEE Trans. Wirel. Commun. **20**(2), 1363–1378 (2021)
29. H. Song, L. Liu, J. Ashdown, Y. Yi, A deep reinforcement learning framework for spectrum management in dynamic spectrum access. IEEE Internet Things J. **8**(14), 11208–11218 (2021)
30. H. Yang, J. Zhao, K.-Y. Lam, Z. Xiong, Q. Wu, L. Xiao, Distributed deep reinforcement learning based spectrum and power allocation for heterogeneous networks. IEEE Trans. Wirel. Commun. **21** (2022)
31. X. Foukas, M.K. Marina, K. Kontovasilis, Iris: deep reinforcement learning driven shared spectrum access architecture for indoor neutral-host small cells. IEEE J. Sel. Areas Commun. **37**(8), 1820–1837 (2019)
32. S. Jacob, V.G. Menon, S. Joseph, P.G. Vinoj, A. Jolfaei, J. Lukose, G. Raja, A novel spectrum sharing scheme using dynamic long short-term memory with CP-OFDMA in 5G networks. IEEE Trans. Cogn. Commun. Netw. **6**(3), 926–934 (2020)
33. H. Eltom, S. Kandeepan, Y.-C. Liang, R.J. Evans, Cooperative soft fusion for HMM-based spectrum occupancy prediction. IEEE Commun. Lett. **22**(10), 2144–2147 (2018)
34. R. Mennes, M. Claeys, F.A.P. De Figueiredo, I. Jabandžć, I. Moerman, S. Latré, Deep learning-based spectrum prediction collision avoidance for hybrid wireless environments. IEEE Access **7**, 45818–45830 (2019)
35. L. Buşoniu, R. Babuška, B.D. Schutter, Multi-agent reinforcement learning: an overview, in *Innovations in Multi-Agent Systems and Applications-1, Chap. 7*. ed. by D. Srinivasan, L.C. Jain (Springer, Berlin, 2010), pp. 183–221

Publisher's Note