Deep Reinforcement Learning for Simultaneous Sensing and Channel Access in Cognitive Networks

Yoel Bokobza, Ron Dabora, *Senior Member, IEEE*, and Kobi Cohen, *Senior Member, IEEE*

***Abstract*— We consider the problem of dynamic spectrum access (DSA) in cognitive wireless networks, consisting of primary users (PUs) and secondary users (SUs), where only partial observations are available at the SUs due to narrowband sensing and transmissions. The network operates in a time-slotted regime, where the traffic patterns of the PUs are modeled as finite-memory Markov chains, that are unknown to the SUs. Since observations are partial, then both channel sensing and access actions affect the throughput. Focusing on the case in which there is a single SU, our objective is to maximize the SU's long-term throughput. To that aim, we develop a novel algorithm that learns *both* access and sensing policies via deep Q-learning, dubbed Double Deep Q-network for Sensing and Access (DDQSA). To the best of our knowledge, this is the first work that jointly optimizes both sensing and access policies for DSA via deep Q-learning. Next, we consider wireless networks with access policy which implements a fixed channel hopping dynamics, for which we analytically determine the optimal SU sensing and access policy and its associated throughput. Then, we demonstrate that indeed, the proposed DDQSA algorithm can achieve near-optimal performance for the considered network. Our results show that the proposed DDQSA algorithm learns a policy that implements both sensing and channel access, which significantly outperforms existing approaches, and can achieve the optimal performance in certain scenarios.**

***Index Terms*— Cognitive radio networks, deep reinforcement learning, dynamic spectrum access, wireless channels.**

I. Introduction

THE increasing demand for wireless communications and the limited availability of the electromagnetic spectrum have triggered the development of efficient methods to increase the spectrum utilization. A main paradigm in this context is dynamic spectrum access (DSA), in which users monitor the spectrum to detect and opportunistically access free channels for communications [1]. There are two main approaches for implementing DSA in wireless networks: A centralized approach and a distributed approach. In centralized access management, there is a central network processor, which is a single point of contact for information sharing, whereas in the distributed management, every node makes access decisions based only on its own observations without sharing information with other nodes. Clearly, distributed access is very attractive due to its low overhead which facilitates higher throughputs.

This work considers DSA for cognitive communications networks. In such networks, every user is designated as either a primary user (PU) or a secondary user (SU). When a PU requires a channel for transmission, then a channel is allocated according to a predetermined resource allocation policy. In contrast, the SUs access the channel opportunistically and independently. To that aim, each SU independently monitors the wireless spectrum to identify free channels which are not being used by the PUs for communication. When properly designed, the incorporation of opportunistic SUs can lead to the desired overall increase in spectrum utilization [2]. DSA has attracted much attention in past as well as in more recent years, see e.g., [1] and [3]. Study and analysis of DSA based on multi-armed bandit (MAB) formulations has been a very active area of research, see e.g., [4], [5], [6], [7], [8], [9], [10], [11], [12], and [13]. In the case of i.i.d channels, such that each channel is modeled as a 2-state Markov chain, representing the channel status as either "busy" or "free", where in addition, the state transition probabilities are *known a-priori* at the SU, and under the assumption that when the channel is in a "busy" state it has a larger probability to remain in the "busy" state than to switch to the "free" state, the myopic policy has been proven to be optimal [4]. In this strategy, the SU accesses the channel that will maximize the expected immediate reward without considering the effect of this action on future rewards. While the myopic policy is easy to understand and simple to implement, it generally does not achieve optimal performance if one of the aforementioned assumptions is violated [4]. Another algorithm that achieves the optimal policy under the same optimality conditions as for the myopic policy is the Whittle index policy [5]. This algorithm is advantageous to the myopic policy in the sense that it can lead

Manuscript received 7 June 2022; revised 1 November 2022; accepted 16 December 2022. Date of publication 3 January 2023; date of current version 12 July 2023. This work was supported in part by the Israel Science Foundation under Grant 584/20 and in part by the 5G WIN Consortium. An earlier version of this paper was presented at the 2022 IEEE International Symposium on Information Theory [DOI: 10.1109/ISIT50566.2022.9834756]. The associate editor coordinating the review of this article and approving it for publication was J. Gross. *(Corresponding author: Ron Dabora.)*

Yoel Bokobza is with Samsung Israel R&D Center, Tel Aviv-Yafo 6492103, Israel (e-mail: yoell017@gmail.com).

Ron Dabora is with the School of Electrical and Computer Engineering, Ben-Gurion University of the Negev, Be'er-Sheva 8410501, Israel, and also with the Department of Electrical and Computer Engineering, Princeton University, Princeton, NJ 08544 USA (e-mail: ron@ee.bgu.ac.il).

Kobi Cohen is with the School of Electrical and Computer Engineering, Ben-Gurion University of the Negev, Be'er-Sheva 8410501, Israel (e-mail: kobi.cohen10@gmail.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TWC.2022.3230872.

Digital Object Identifier 10.1109/TWC.2022.3230872

to the derivation of good access policies even if the channels are not identically distributed. A major weakness of both the myopic policy and the Whittle index policy is that they are not applicable in scenarios in which the channels are correlated. Another major disadvantage is that both the myopic policy and the Whittle index policy require full knowledge of the state transition probabilities, which is often unavailable in practical scenarios. This requirement has motivated the introduction of methods which can acquire a nearly optimal policy without requiring such a-priori knowledge. A major technique which is capable of achieving this goal is reinforcement learning (RL), which is a class of machine learning algorithms that can learn an optimal policy via interaction with the environment, without knowledge of the system dynamics (such algorithms are also known as model-free algorithms) [14, Ch. 1]. One of the most popular RL techniques is Q-learning [15], which can learn the optimal policy online by estimating the optimal action-value function. Early works which applied Q-learning to DSA used the classical tabular Q-learning method [16], [17]. However, it becomes computationally difficult to apply this method when the state space becomes large. This issue has motivated the combination of deep learning with RL, giving rise to the deep reinforcement learning (DRL) class of algorithms. These algorithms have attracted much attention in recent years due to their ability to approximate the action-value function for large state and action spaces. Recently, the work [18] proposed a DRL-based algorithm called deep Q-network (DQN), which combines deep neural networks and Q-learning. Recent studies which derived DRL-based algorithms for DSA problems can be found in [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], and [29]: In [19] and [20], the authors applied DQN to solve the DSA problem, where it was assumed that at each time step, the SU can choose one channel to access for which it receives an ACK/NACK signal as feedback, based-on which a reward is computed. In [19] and [20], the observations consist of the indices of the past accessed channels and the corresponding rewards, which are used as the inputs to the DQN algorithm. In [21], the authors assumed multi-band spectrum sensing without power limitations, which corresponds to a fully-observed scenario, and trained a DQN to select which channel to access in the next time step based on the current state of the entire spectrum. In [23] and [26], the authors used a deep recurrent Q-Network (DRQN), which is a combination of a DQN and a long short-term memory (LSTM) network to derive the optimal access policy when observations are obtained using an a-priori set sensing pattern, in which at each time step the SU senses half of the available channels, such that a different half is sensed at each subsequent time step. The LSTM layer in the DRQN algorithm uses past observations for the prediction of the state, which, in turn, allows the agent to select a channel for accessing at the next time step. In [22] and [25], the authors used another DRL algorithm called deep actor-critic algorithm, which is a policy-based RL algorithm in which the policy is learned directly via a deep neural network (DNN). They compared their results with that of the algorithm in [20] and showed that their proposed algorithm achieves better performance. The work in [30] used the same system model as in [20] and applied an algorithm called hierarchical-DQN (h-DQN) [31]. h-DQN facilitates exploration in complicated environments by decomposing the original problem into a hierarchy of sub-problems such that higher-level tasks invoke lower levels as if they were primitive actions. In [32] the authors extended the work of [20] to the case in which at each time slot, the SU selects a block of channels to access from a predefined set of channel blocks, and the sensing decision is completely determined by the access decision. The multi-user problem, where a multiple users access the channel opportunistically, was considered in [24], [33], [34], and [35] and formulated, with different variations among these works, as a multi-agent RL problem [36].

In this work we consider the design of a DSA algorithm for cognitive communications networks consisting of multiple PUs and a single SU, which is a common benchmark for the design of DSA algorithms, see, e.g., [19], [20], [21], [22], [23], [25], [26], and [27]. In practical implementations, due to bandwidth limitations inherent to the sensing operation, an SU can sense only a part of the available spectrum (e.g., narrowband sensing), which implies that when operating in a distributed manner, access decisions are based on *partial* observations. Naturally, due to the operation of physical components, sensing also exhibits errors and the sensing result can be assigned a level of certainty, e.g., corresponding to the distance of the test value from the decision threshold. For the purpose of this study, channel access of the PUs and SU is implemented in a time-division multiple access (TDMA) manner, with fixed-length time slots. PUs' transmissions take place in frames, which span a random number of TDMA time slots, such that for each PU, the frame length is modeled as a finite-memory Markov chain, where different PUs may have different state transition probabilities.

The SU is oblivious to the PUs' statistics, hence, at each time slot, based on its previous observations, the SU selects which channels to sense, and whenever it needs to transmit, it also selects a single channel for transmission at the next time slot. Whenever the SU transmits on a channel that is not occupied by a PU, it receives an acknowledgment (ACK) signal indicating a successful transmission. Otherwise, a negative acknowledgment (NACK) signal is received, denoting an unsuccessful transmission. The objective of the DSA algorithm in such a setup is to maximize the long-term rate of successful transmissions. As an SU can sense only a subset of the network's bandwidth, this problem can be formulated as a partially observable Markov decision process (POMDP) [20]. In a case where the transition probabilities of the PUs' Markov chains are *known to the SU*, an exact solution to this problem is P-SPACE hard and has an exponential computational complexity [37]. In real-world models, as considered in the current work, the DSA problem is even harder to solve, since the *SU does not know the state probabilities of the PUs*, nor does it know which PU uses which channel. As a result, the SU does not know the state transition probabilities of the wireless network. As the designed agent *operates independently*, s.t. it bases its model updates and decisions only on its own sensing and access outcomes, the proposed approach is suitable

for distributed implementation at multiple SUs. This point is demonstrated and discussed in Sec. V-E.

A. Context and Main Contributions

To facilitate model-free learning, we design a DRL-based algorithm to achieve the optimal policy. A major novel and unique aspect of our approach is that as successful access decisions heavily rely on sensing, it is advantageous to let the agent select *both* the channels to be sensed, as well as the channel to be accessed in the next time step. This is in contrast to previous works which designed DRL-based access policies to maximize the throughput using *predetermined* sensing policies. Thus, the performance of previous algorithms are generally not optimal for all PU access policies. In this work, we develop a novel algorithm which decouples the sensing and the access operations by learning both access and sensing policies via deep Q-learning. For efficient learning, we employ a double deep Q-network (DDQN), which is a combination of double Q-learning [38] and deep neural network, which facilitates learning from experience in an unknown environment with a large state space via interactions with the environment [39]. Accordingly, our proposed algorithm is dubbed Double Deep Q-network for Sensing and Access (DDQSA). Our numerical tests evaluated the throughput achieved by DDQSA under several PU access strategies, and compared the throughput obtained by DDQSA with those obtained by a DRL which uses deterministic sensing (i.e., a DRL which makes only access decisions with a predetermined sensing policy), and by an algorithm that performs an access without any sensing. We show that DDQSA facilitates using smaller sensing bandwidths compared to previously proposed schemes for achieving the same througputs, whereas for the same sensing bandwidth DDQSA achieve superior performance. To the best of our knowledge, this is the first paper that solves both sensing and access policies for DSA via deep Q-learning. Furthermore, our algorithm can handle situations in which the SU does not have data to send, hence reward is not available. Another important property of the proposed algorithm is the combination of sensing optimization with soft sensing, which indicates when sensing outcomes have a low certainty. An interesting and practically important outcome of this work is that learning both sensing and access facilitates *decreasing the sensing bandwidth with only a minor decrease in the throughput*, as the SU can *focus sensing on the relevant channels* for making access decision for the next time step. In contrast, sensing bandwidth decrease results in a significant performance loss in the benchmark algorithms.

An additional major novelty of the current work is the theoretical derivation of the *optimal joint sensing and access policy* for a generalized network implementing PU random access based on randomly switching channels w.r.t a fixed channel hopping pattern, originally considered in [20]. We note that frequency hopping is an important approach for achieving diversity, and DSA for frequency hopping networks is an important scenario, which attracted considerable attention [26], [40]. The optimal policy is derived by exploiting the structure of the users' dynamics to derive the optimal *sensing* policy, which transforms the optimal access policy problem into a Markov decision problem (MDP), instead of a POMDP. Then, we derive the corresponding optimal access policy using the Bellman optimality equations [14, Ch. 3] explicitly. It is numerically shown that DDQSA achieves near-optimal performance for the fixed hopping pattern dynamics (FHPD) network, again exceeding the performance of the benchmark schemes.

B. Organization and Notations

The rest of this paper is organized as follows: Section II details the network setup and assumptions; Section III, motivates and discusses the rationale for the selected DRL approach, and details the proposed DDQSA algorithm. Section IV, presents the derivation of the optimal sensing and access policies for a network with a FHPD. These optimal policies serve as a benchmark for testing our DDQSA algorithm. Section V reports simulation results, including a comparison with approaches proposed in previous works and with the optimal scheme (when possible). These results clearly demonstrate the advantages of the proposed approach over other approaches. Lastly, Section VI concludes this work.

We use $\mathbb{N}$, $\mathbb{R}$ to denote set of natural numbers and of real number, respectively. Bold letters, e.g., $\mathbf{S}$, denote vectors, and $\mathbf{S}_i$ denotes the i'th element in the vector $\mathbf{S}$, $i \geq 0$. Calligraphic letters denote sets, e.g., $\mathcal{S}$, and the cardinality of a set is denoted by $|\cdot|$, e.g., $|\mathcal{S}|$ is the cardinality of the set $\mathcal{S}$. Lastly, $\mathbb{E}\{\cdot\}$ denotes the stochastic expectation.

II. Problem Formulation

We consider a wireless network with $N \in \mathbb{N}$ channels, $K_p \in \mathbb{N}$ PUs, $K_p \leq N$, and a single SU. We denote the i'th PU by pu_i, for $i \in \{0, 1, \ldots, K_p - 1\}$, and abbreviate the n'th channel as ch_n, $n \in \{0, 1, \ldots, N-1\}$. For each PU, the random length of the transmitted frame is modeled as a finite-memory Markov chain, where different PUs may have different state transition probabilities for their corresponding chains. The SU does not have knowledge of the Markov chains of the PUs. The PUs access the channel according to a predetermined policy, which guarantees channel allocation to every PU when it needs to transmit. For pu_i, $i \in \{0, 1, \ldots, K_p - 1\}$, we set the maximal frame length to M_i. Let $\mathcal{L}_i = \{0, 1, \ldots, M_i\}$ denote the state space for pu_i, and m_i denote its current state. When pu_i is at the k'th time slot of its frame we set its state to $m_i = k$. When pu_i is not transmitting, referred to as idle, we set its state to $m_i = 0$. Denote by $P_i(k|j)$, $j, k \in \mathcal{L}_i$ the probability that PU pu_i will make a transition from state j to state k. Because the frame length is bounded, $P_i(0|M_i) = 1$, $\forall i \in \{0, 1, \ldots, K_p - 1\}$. In addition, $P_i(k|j) = 0$ $\forall j, k$ such that $0 \leq j < M_i$, $0 < k \leq M_i$ and $k \neq j + 1$. A diagram which illustrates the state transition probabilities for pu_0 with $M_0 = 3$ is depicted in Fig. 1.

At each time step, the SU selects a set of $L \in \mathbb{N}$ channels, $L \leq N$, for sensing. The sensing outcomes constitute the observations, based on which the SU selects a channel for transmission at the next time step, such that the long-term

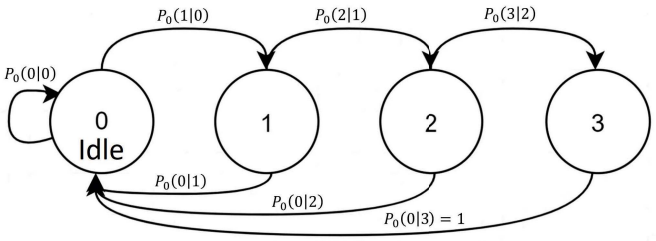

Fig. 1. An illustration of the state transition diagram for pu_0 with $M_0 = 3$.

throughput is maximized. Note that the approach of selecting a channel to use in the next time slot is a common practice in DSA literature, see, e.g., [20], [21], [22], and [26]. We note that this setup is quite general, as for $L = 1$ sensing is strictly narrowband (i.e., a single channel is sensed), while for $L > 1$ sensing is considered wideband (i.e., multiple channels are sensed simultaneously). It is also noted that wideband sensing is very common in the DSA literature, see, e.g., [21] and [26]. For simplicity of the presentation we further assume that L is a divisor of N. The sensing action outcome can be either "free", when the channel is not being accessed by any PU, "busy" when the channel is being accessed by a PU, or "undetermined", when the test metric of the sensing algorithm is too close to the decision threshold, as determined by the system designer. Based on its past and current observations, the SU selects a channel for transmission in the *next* time step. If the SU has data for transmission, it transmits at the next time-step using the selected channel, and receives feedback: If the selected channel was free, the SU transmission was successful and it receives an ACK feedback from its destination. Otherwise, the SU receives a NACK feedback which indicates that transmission has failed, and implies that the channel was busy. Note that in practice, NACK can be determined by a timer and is not necessarily a response received from the SU's destination. It is emphasized that an underlying assumption of the DSA system is that PUs which affect the SU's communications link are received at the SU, and for PUs which are not received at the SU, it holds that their communications performance are not affected by SU's transmissions. This assumption is not specific to the use of ACK/NACK feedback, and is relevant to any other scheme in which SU's decisions are based only on its independently obtained inputs. It is clarified that there is a fundamental difference between the *sensing action outcome*, which can be either "free", "busy", or "undetermined", and the *access action outcome* which is the ACK/NACK feedback: The sensing action outcome is obtained after the sensing action is applied to the channel subset selected at the previous time-slot. As the sensing action may span the entire time-slot duration, its outcome is typically reliable. The sensing action outcome provides the algorithm with *partial state information* for making the next access decision. The access action outcome has a different role in the algorithm, as it indicates if the algorithm *correctly selected a free channel to access*, namely, it determines the *reward* for the algorithm's decision, which is critical for updating the Q-values in order to improve the future access decisions of the algorithm (i.e., its processing of the future partial state information and history). It is noted that, similarly to previous works, [20], [25], [30], we do not make any assumption on PU reception when a collision occurs between the SU's and a PU's transmissions, as the outcome of PU's reception at the PU's destination is not available at the SU, and hence does not impact the algorithm. It is intuitively understandable that maximizing SU's throughput is correlated with minimizing SU-PUs collisions, which will minimize the impact of the SU on the PU network. It is also noted that as the ACK/NACK mechanism constitutes a feedback indicating whether the access decision was successful or not, it can be replaced by any other feedback mechanism which can achieve this objective. One alternative is to let the SU apply carrier sensing (CS) to the channel selected for access and determine if it is "busy" or "free", without actually transmitting over it, see, e.g., [41]. With CS, if the SU concludes that the channel is busy, it will typically not have time for applying CS to another channel at the same time slot, therefore it will not transmit at all during this slot to avoid collision. It follows that, as the SU selects a channel for access at the *next time slot* based on its *current and previous* observations and feedback, then the role of the CS outcome in a DSA algorithm is identical to that of the ACK/NACK feedback.

Note that as with CS collision is avoided, then the chances of successful PU transmission may improve. Yet, at the same time, CS has several major drawbacks, including less accurate sensing performance due to the short sensing interval, which results in increased chances of feedback error (inducing errors in the reward) and a decrease in throughput due to the allocation of slot time for CS instead of for data transmission. Hence, in the subsequent description, similarly to [20], [21], [25], and [26], we will use the ACK/NACK feedback as an indication of the success/failure of SU's access decisions.

As the agent has two actions – sensing and channel access, we begin by formulating the problem as a POMDP with two policies: One policy for sensing and one for channel access. This formulation is quite intuitive and simplifies the explanations in the following sections. In Prop. 1 in Section III-C, we will show that this problem can be formulated as a single policy problem, and thus can be solved with a *single agent*. Let $\mathcal{S} = \{-1, 1\}^N$ be the channel state space, where '1' denotes that the channel is currently being used for transmissions and '-1' denotes that the channel is free. Let $\mathbf{s}^{(i)} \in \mathcal{S}$ be the length-N polar form representation of the integer $i \in \{0, 1, \ldots, 2^N - 1\}$, where the j'th element in $\mathbf{s}^{(i)}$, $\mathbf{s}_j^{(i)}$, represents the state of the j'th channel ('1' denotes that the channel is busy, and '-1' denotes that the channel is free), and let $\mathbf{s}(t) \in \mathcal{S}$ denote the state at time step $t \in \mathbb{N}$. We define two action sets: $\mathcal{A}_s \triangleq \{0, 1, \ldots, \frac{N}{L} - 1\}$ is the action set for sensing, and $\mathcal{A}_{ac} \triangleq \{0, 1, \ldots, N - 1\}$ is the action set for access. Accordingly, the set of N channels is partitioned into sensing subsets, each sensing subset consists of L channels, and the action $a_s(t) = i \in \mathcal{A}_s$ denotes that the SU decides to sense the channels $\{\text{ch}_m\}_{m=i\cdot L}^{(i+1)\cdot L-1}$ at time step $t + 1$. The action $a_{ac}(t) = j \in \mathcal{A}_{ac}$ denotes that the

SU decides to access ch_j for transmission at time step $t+1$. As the SU senses only a subset of L channels of the network, the observations space, denoted by $\mathcal{X}$, $\mathcal{X} = \{-1,0,1\}^L$, is the set of all possible observation outcomes in the set of the currently observed L channels, where '-1' denotes that a sensed channel was determined free, '1' denotes that it was determined busy, and '0' denotes the sensing outcome is undetermined. The observation outcome at time $t \in \mathbb{N}$ is denoted by $\mathbf{x}(t) \triangleq \big[l, \big(x_0(t), x_1(t), \ldots, x_{L-1}(t)\big)\big]$, where the first element $\mathbf{x}_0(t) = l \in \{0,1,\ldots,\frac{N}{L}-1\}$ indicates that at time t, the l'th subset of channels was sensed, i.e., $a_s(t-1) = l$, and $\big(x_0(t), x_1(t), \ldots, x_{L-1}(t)\big) \in \mathcal{X}$, denotes the sensing outcome at time step $t \in \mathbb{N}$, where the sensing outcome $x_i(t) \in \{-1,0,1\}$ denotes the sensing outcome at time step t for the i'th channel of the l'th sensing subset. We next define two policies: The sensing policy, denoted by π_s, and the access policy, denoted by π_{ac}. Let $R(t)$ denote the reward at time step t. When a transmission is successful, i.e., when at time step t the algorithm correctly selects a free channel for transmission at time $t+1$ (i.e., the SU receives an ACK signal at time $t+1$), we set the reward to $R(t+1) = 1$, to encourage the agent to access free channels. When the selected channel at time t is busy at time $t+1$ (i.e, SU receives a NACK signal), then $R(t+1) = -1$, to discourage the agent from selecting a busy channel for transmission. When no transmission takes place, e.g., the SU does not have data to transmit, then no reward is obtained. It is obvious that the sensing policy and the access policy are related, which poses the interesting question of *which channels should be sensed and what is the matching access policy, in order to maximize the throughput.* In this work we try to answer this question via learning.

Let $\mathbf{o}^H(t)$ denote the state of the network at time $t \in \mathbb{N}$, to be defined explicitly in Section III-C. Our objective is to derive an RL-based algorithm to identify the pair of policies π_s^*, and π_{ac}^* that maximize the expected accumulated discounted reward over an infinite time horizon, i.e.

$$(\pi_s^*, \pi_{ac}^*) = \underset{\pi_s, \pi_{ac}}{\text{argmax}} \left\{ \mathbb{E}_{\pi_s,\pi_{ac}} \left\{ \sum_{t=1}^{\infty} \gamma^{t-1} R(t+1) \Big| \mathbf{o}^H(1) \right\} \right\}, \tag{1}$$

for a discount factor $\gamma \in [0,1)$.

III. The Proposed DRL-Based Algorithm for Channel Sensing and Access

In this section, we describe the proposed DDQSA algorithm. Prior to detailing the algorithm, we briefly review Q-learning and DQN, to motivate our selection of this algorithmic approach.

A. Q-Learning

Q-learning is an RL algorithm, which, under certain assumptions, obtains the optimal policy for an MDP, in the sense of maximizing the expected accumulated discounted reward for any given initial state [14, Ch. 6]. The Q-learning algorithm is a value-based RL algorithm, which means that it computes the optimal action-value function in order to identify the optimal policy. Let $\mathcal{A}$ denote the set of actions, $\mathcal{S}$ denote the set of states, and let $q_\pi(s,a)$, $s \in \mathcal{S}$, $a \in \mathcal{A}$, denote the action-value function, which is the expected accumulated discounted reward starting from state s, picking action a, and following policy π afterwards. Letting $\gamma \in [0,1)$, denote the discount factor, and considering the infinite time-horizon problem, $q_\pi(s,a)$ can be expressed as [14, Ch. 3]

$$q_\pi(s,a) \triangleq \mathbb{E}_\pi \left\{ \sum_{k=1}^{\infty} \gamma^{k-1} R(t+k) \Big| \mathbf{s}(t) = s, a(t) = a \right\}.$$

The optimal policy π^* is a policy that satisfies $q_{\pi^*}(s,a) \geq q_\pi(s,a)$ for any policy π and for every possible state-action pair, $(s,a) \in \mathcal{S} \times \mathcal{A}$. The optimal policy can be obtained easily from the optimal action-value function, $q_{\pi^*}(s,a)$, as $\pi^*(s) = \underset{a \in \mathcal{A}}{\text{argmax}}\{q_{\pi^*}(s,a)\}$.

The Q-learning algorithm iteratively estimates the optimal action-value function for each valid state-action pair in an online manner as follow: At each time step $t \in \mathbb{N}$, the agent observes a state $s \in \mathcal{S}$, selects an action $a \in \mathcal{A}$, receives a reward r for executing the selected action $a \in \mathcal{A}$, and observes the next state $s' \in \mathcal{S}$. Then, the *estimation* of the corresponding $q_{\pi^*}(s,a)$, referred to as the Q-value and denoted as $Q(s,a)$, is updated according to the update rule:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \cdot \Big(r + \gamma \cdot \max_{a' \in \mathcal{A}} \big\{ Q(s',a') \big\} - Q(s,a) \Big), \tag{2}$$

for some $\alpha \in (0,1)$ referred as the learning rate. To explore various state-action pairs, the action a is selected according to an ϵ-greedy policy, meaning that $(1-\epsilon)$ of the time the selected action maximizes the estimated optimal action-value function, whereas in ϵ part of the time, the action is selected randomly from the set of all valid actions. Mathematically, the agent at state $s \in \mathcal{S}$, selects an action $a = \underset{a' \in \mathcal{A}}{\text{argmax}}\{Q(s,a')\}$ with probability $1-\epsilon$, and a uniformly random action over all possible actions in state s, with probability ϵ. According to [14, Ch. 6], this algorithm is proven to converge to $q_{\pi^*}(s,a)$ with probability 1 if all of the state-action pairs are visited infinitely often, as long as a variant of the usual stochastic approximation conditions is satisfied. In a general DSA setting, as considered here, the *transition probabilities are unknown* and only *partial observations* are available, consequently, *convergence is not guaranteed* by the theory.

B. Deep Q-Network

While Q-learning performs well when the state and action spaces are small and provably converges for MDP formulation, it becomes impractical for large state and action spaces, due to two main reasons: The first reason is that in the Q-learning algorithm, the agent has to visit multiple states and select different actions at each state to learn the optimal Q-value, which requires an extensive exploration, resulting in a long learning time. The second reason is that in the Q-learning algorithm, the agent has to store the Q-value for every state-action pair, which leads to a large storage requirement for large action and state spaces. Recently, a class

of DRL algorithms that combines Q-learning and DNNs, referred to as DQN, has been proposed. The role of the DNN in the DQN is to map observations and actions into their Q-values, which eliminates the need to store the Q-values in a table, thereby significantly reducing the storage requirement for large action and state spaces. Furthermore, the DNN is able to extract features from previous observations in order to infer the Q-value for situations which have not yet been observed [42]. This capability does not exist when using tabular methods, as when using tables, each state and action pair has to be visited to estimate the corresponding Q-value. To improve stability of convergence, DQN is implemented using a pair of DNNs: One DNN maps the action-state pairs into Q-values in the calculation of the target value, defined as $r+\gamma\cdot\max_{a'\in\mathcal{A}}\left\{Q(s',a')\right\}$, and a second DNN, with an identical structure, maps the action-state pairs into Q-values in the calculation of the difference between the target value and the Q-value, used in the network parameters update step, see, e.g., [39].

It should be noted that DQN-based learning does not have a guaranteed convergence to the optimal solution, even for problems which can be formulated as an MDP. In practice, however, it achieves very good performance even in various POMDP models with infinitely large state space. For example, the work of [18] developed a DQN algorithm for teaching an agent how to play Atari games directly from screen images, and achieved very good performance in various Atari games. It was observed in [38] that the standard DQN implementation, which uses a single DNN in the calculation of the target value (in which both the action that maximizes the estimated Q-value is obtained, and subsequently the Q-value with the selected action is evaluated), usually results in an overestimation of the Q-values, and consequently causes performance degradation. In [39], it was proposed to implement the target computation using two DNNs: One for selecting the maximizing action and the other for estimating the Q-value associated with the selected action. This scheme, called DDQN, was shown to achieve better performance.

While DQN achieves good performance in MDP problems, in the POMDP framework corresponding to the DSA problem, it suffers from performance degradation due to partial observations as they do not provide sufficient information, namely do not provide the state of *all the channels* in the network. Our proposed approach handles this issue by letting the algorithm select the current observation (from a predefined set), and combine it with a collection of a sufficient number of past observations. This improves the algorithm's inference about the actual system state, and, in some situations, facilitates finding a (near-)optimal access policy which maximizes the throughput. This is the first time a *joint learning of sensing and access policies* is implemented via online learning by using DDQN; Another novel aspect in our approach is accommodating the practical scenario in which an SU does not always have data for transmission, hence reward is not received at every time instance. We also accommodate the inherent uncertainty in sensing decisions by supporting soft sensing, which is implemented by setting a range of sensing metric values for which the detector output is set to "undetermined". Such a mechanism may improve performance by decreasing the probability of erroneous sensing decisions.

C. The Proposed DDQSA Algorithm

We start by showing that the DSA problem defined in Section II can be formulated as a single agent problem with a single policy for both sensing and access. This formulation implies better convergence properties for the DRL algorithm, as compared to using two agents – one for optimizing the access policy and one for optimizing the sensing policy. This single agent formulation is next stated in Proposition 1:

Proposition 1: For the network setup in Section II, the optimal (in the sense of maximal throughput) sensing and access policies can be achieved by a single agent.

Proof: Due to the partial observations and the memory of the PUs' traffic, we maintain a history vector consisting of $H\in\mathbb{N}$ most recent past observations to facilitate extracting all available relevant information about the state of the channel. We define the history-observations space as $\mathcal{O}_H=\{-1,0,1\}^{N\cdot H}$, where $\mathbf{o}^H(t)=[\mathbf{o}(t-H+1),\mathbf{o}(t-H+2),\ldots,\mathbf{o}(t)]\in\mathcal{O}_H$. $\mathbf{o}(t)$ is a length N vector that represents the observation outcomes at time step $t\in\mathbb{N}$, where $\mathbf{o}_i(t)=1$ denotes that ch_i was sensed at time step t and was found to be busy, $\mathbf{o}_i(t)=-1$ denotes that ch_i was sensed at time step t and was found to be free, and $\mathbf{o}_i(t)=0$ denotes that ch_i was not sensed at time step t, or, alternatively, it was sensed, and the sensing outcome is "undetermined". The extended action space which facilitates selection of both sensing and access actions is denoted by $\mathcal{A}_{ex}=\left\{0,1,\ldots,\frac{N^2}{L}-1\right\}$. At each time step, the agent picks an action $a(t)\in\mathcal{A}_{ex}$, where $a(t)=i$, means that at time step t, the agent chooses to sense channels $\{\text{ch}_m\}_{m=\lfloor\frac{i}{N}\rfloor\cdot L}^{(\lfloor\frac{i}{N}\rfloor+1)\cdot L-1}$ and to transmit on channel $\text{ch}_{(i\pmod N)}$ at the next time step. It thus follows that with the extended action space, using a single index we can enumerate all possible pairs of access and sensing actions, hence there is an invertible one-to-one mapping between every element of the extended space $\mathcal{A}_{ex}$ and a pair of actions in the space $\mathcal{A}_s\times\mathcal{A}_{ac}$. Indeed, as there are $\frac{N}{L}$ sensing actions and N access actions, the number of pairs is $|\mathcal{A}_s\times\mathcal{A}_{ac}|=\frac{N^2}{L}=|\mathcal{A}_{ex}|$.

Note that although the immediate reward $R(t+1)$ is the same for every action $a(t)$ with the same value of $a(t)\pmod N$, the subsequent observation actions are different when $\left\lfloor\frac{a(t)}{N}\right\rfloor$ are different, thus the target, $R(t+1)+\gamma\cdot\max\limits_{a'\in\mathcal{A}_{ex}}\left\{Q\big(\mathbf{o}^H(t+1),a'\big)\right\}$ will be different and this would result in the agent learning a better joint sensing and access policy over time, where the learned access policy is coordinated with the learned sensing policy. Following this formulation, the optimization in (1) can be equivalently expressed as finding the optimal policy π^* such that

$$\pi^*=\underset{\pi}{\text{argmax}}\left\{\mathbb{E}_\pi\Big\{\sum_{t=1}^{t=\infty}\gamma^{t-1}R(t+1)\Big|\mathbf{o}^H(1)\Big\}\right\}.$$

From this unified policy we can obtain both the sensing policy and the access policy by setting $\pi_s^*(\mathbf{o}^H)=\left\lfloor\frac{\pi^*(\mathbf{o}^H)}{N}\right\rfloor\in\mathcal{A}_s$, and $\pi_{ac}^*(\mathbf{o}^H)=\pi^*(\mathbf{o}^H)\pmod N\in\mathcal{A}_{ac}$ for any history observation outcome $\mathbf{o}^H\in\mathcal{O}_H$. ■

The DDQSA algorithm utilizes the DDQN architecture originally proposed in [39], which combines double Q-learning and deep neural networks. In DDQSA, we use the observations history $\mathbf{o}^H(t)$ as the state of the wireless network, which is used directly as an input to the DDQN. The output layer of the network consists of $|\mathcal{A}_{ex}|$ neurons, where the i'th neuron at the output layer, $i \in \mathcal{A}_{ex}$, represents an estimation of $q_{\pi^*}(\mathbf{o}^H(t), i)$. To balance between exploration and exploitation, we used the ϵ-greedy policy: At each time step $t \in \mathbb{N}$, the agent selects an action $a(t) = \underset{a' \in \mathcal{A}_{ex}}{\operatorname{argmax}} \{Q(\mathbf{o}^H(t), a')\}$ with probability $1 - \epsilon(t)$, and selects a random action uniformly among all actions with probability $\epsilon(t)$, where $\epsilon(t)$ decays to zero as time increases. This decay follows as the agent improves its policy over time, hence it should rely more on the learned policy and less on the random exploration.

From the above description it is straightforward to conclude that the size of the state space is $2^{N \cdot H}$, the size of the input space is $|\tilde{\mathcal{X}}|^{L \cdot H} \cdot (\frac{N}{L})^H$, where $\tilde{\mathcal{X}}$ is set of possible sensing outputs at each sensed channel, and the size of action space $\mathcal{A}_{ex}$ is given as $\frac{N^2}{L}$. Even for modest values of L, N, and H, as used in the simulations in Sec. V we obtain a very large state-action space which further motivates the use of a DQN. For example, in the experiments reported in Sec. V-B, with $N = 10, L = 5$, $H = 6$, and $|\tilde{\mathcal{X}}| = 2$, the size of the state space is 2^{60}, the size of the input space is 2^{36} and the size of the action space is 20.

D. Handling SU's Random Transmission Requests

In practical scenarios a node may also have idle times. In those idle times there is no ACK/NACK signal received. In order to facilitate handling SU's idle times, we update the value of $\epsilon(t)$ only at time steps in which the SU transmits, i.e., letting $\tilde{t} \in \mathbb{N}$ denote the counter of SU's transmissions, then we set $\epsilon(\tilde{t})$ be a decaying function of $\tilde{t}$. Then, the agent executes both actions $a_s(t)$ and $a_{ac}(t)$ only when the SU needs to transmit at time step $t + 1$, whereas when the SU has nothing to transmit at time step $t + 1$ the agent executes only action $a_s(t)$. Accordingly, when the SU has nothing to transmit at time step $t+1$, then, as no reward is received, the replay buffer, which stores tuples of observations, actions and rewards, which are used for carrying out DNN training [18], is not updated. It is noted that *training continues irrespective of whether the SU transmits or not*, as training is based on mini-batches selected from the replay buffer.

E. Pseudocode of DDQSA

Let $\mathcal{D}$ denotes the replay buffer, and $\boldsymbol{\theta}, \boldsymbol{\theta}^-$ denote the policy network weights, and the target network weights, respectively. The steps of the proposed DDQSA algorithm are summarized in Algorithm 1.

Algorithm 1 The DDQSA Algorithm for Jointly Optimizing Spectrum Sensing and Access

Set replay buffer size to C, update rate to J, minibatch size to M_B, exploration decay rate to ξ, time counter to $t = 0$ and SU transmission counter to $\tilde{t} = 0$.
Initialize replay buffer $\mathcal{D}$ with capacity C, and a mini-batch array $\mathcal{M}_B$ with size M_B.
Initialize the policy network weights $\boldsymbol{\theta}$ randomly.
Initialize the target network weights $\boldsymbol{\theta}^- \leftarrow \boldsymbol{\theta}$.
Set $a_s(1)$ randomly from $\mathcal{A}_s$ and observe $\mathbf{o}^H(1)$.
for time step $t = 1, 2, \ldots$ **do**
  Set $\epsilon(\tilde{t}) = \frac{1}{1+\xi \cdot \tilde{t}}$.
  $\epsilon(\tilde{t})$-greedy:
$$a(t) = \begin{cases} \underset{a' \in \mathcal{A}_{ex}}{\operatorname{argmax}} \{Q(\mathbf{o}^H(t), a')\} & \text{w.p. } 1 - \epsilon(\tilde{t}), \\ \text{uniformly random action, } a \in \mathcal{A}_{ex} & \text{w.p. } \epsilon(\tilde{t}). \end{cases}$$
  if SU has data to send **then**
    Execute actions: $a_s(t) = \lfloor \frac{a(t)}{N} \rfloor$; $a_{ac}(t) = a(t) \pmod N$.
    $\tilde{t} \leftarrow \tilde{t} + 1$.
    Obtain the reward $R(t + 1)$, and observe the next state $\mathbf{o}^H(t + 1)$.
    Store the tuple $\big(\mathbf{o}^H(t),\ a(t),\ R(t + 1),\ \mathbf{o}^H(t + 1)\big)$ in $\mathcal{D}$.
  else $a_s(t) = \lfloor \frac{a(t)}{N} \rfloor$.
  end if
  Sample a mini-batch $\mathcal{M}_B \triangleq \{(\mathbf{o}_i, a_i, r_i, \mathbf{o}'_i)\} | 1 \le i \le |\mathcal{M}_B|\}$ randomly uniformly from $\mathcal{D}$.
  Set target
$$y_i = r_i + \gamma \cdot Q(\mathbf{o}'_i, \underset{a' \in \mathcal{A}_{ex}}{\operatorname{argmax}} \{Q(\mathbf{o}'_i, a'; \boldsymbol{\theta})\}; \boldsymbol{\theta}^-)$$
  Perform batch training with inputs $\mathbf{o}_i$, and outputs y_i, using all $(\mathbf{o}_i, a_i, r_i, \mathbf{o}'_i) \in \mathcal{M}_B$.
  Every J iterations set $\boldsymbol{\theta}^- \leftarrow \boldsymbol{\theta}$.
end for

IV. Analytical Derivation of the Optimal Sensing and Access Policies for a Network With a Fixed Channel Hopping Dynamics

In this section, we analytically derive the optimal sensing and access policy for a network in which PUs' access is based on randomly selecting a single shift or a double shift over a fixed channel hopping pattern. Frequency hopping is a very important diversity mechanism and thus constitutes an important benchmark scenario, see [20] and [26]. In this work we consider a hopping pattern whose structure facilitates intuitive understanding of the operations of the different channel access schemes. While the general DSA problem is P-SPACE hard [37], for a network with a fixed hopping pattern we were able to analytically derive the optimal sensing and access policies.

A. Network Setup

Consider a network consisting of $N \ge 2$ channels, where N is assumed to be an even integer, let the size of the sensing subset be $L = 2$, and let the number of PUs be $K_p = N - 1$. Denote the set of possible indexes of the free channel with $\mathcal{U} = \{0, 1, \ldots, N - 1\}$, and define $U(t) \in \mathcal{U}$, such that $U(t) = i$ denotes that the free channel at time step $t \in \mathbb{N}$ is the i'th channel, ch_i. We also define the channel hopping pattern vector as $\mathbf{B} \triangleq [\mathbf{B}_0, \mathbf{B}_1, \ldots, \mathbf{B}_{N-1}] = [2b_0, 2b_0 + 1, 2b_1, 2b_1 + 1, \ldots, 2b_{N/2-1}, 2b_{N/2-1} + 1]$, where $\{b_i\}_{i=0}^{N/2-1}$

is a random permutation of the numbers $0, 1, \ldots, \frac{N}{2} - 1$. The PUs access the channel at every time step (i.e., PUs always transmit) according to the following policy: At each time step, the PUs either transmit at the same channel as in the previous time step with probability P_{stay}, or *jointly* switch to the next channel in the hopping pattern vector in a cyclic manner (namely, the node using the last channel in the hopping pattern vector switches to the channel indexed by the first element in the vector), with probability P_{switch}, or *jointly* switch two channels to the right in the hopping pattern vector, in a cyclic manner, with probability $P_{Dswitch} = 1 - P_{stay} - P_{switch}$. It follows that the state space of this network consists of N states, each corresponding to one possible location of the single free channel that can switch its position with probabilities P_{stay}, P_{switch}, and $P_{Dswitch}$ according to the PU access policy. With these definitions, for any $\mathbf{B}_s, \mathbf{B}_{s'} \in \mathcal{U}$, the state transition probability is given by:

$$\Pr(\mathbf{B}_{s'}|\mathbf{B}_s) = \begin{cases} P_{stay} & \text{if } s' = s, \\ P_{switch} & \text{if } s' = s+1 \pmod N, \\ P_{Dswitch} & \text{if } s' = s+2 \pmod N, \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

B. Derivation of the Optimal Sensing and Access Policy for an FHPD Network

Letting $P_{max} \triangleq \max\{P_{stay}, P_{switch}, P_{Dswitch}\}$, the sensing and access policy which maximizes the throughput is stated in the following theorem:

Theorem 1: For the FHPD policy detailed in Section IV-A, assuming ideal sensing (i.e., sensing outcome is either 1 or −1 and there are no sensing errors), the following three assertions hold:

1) *For any finite history length $H \geq 2$, there exists a finite $t_0 \in \mathbb{N}$ such that, there exists a sensing policy which results in the state being fully observable for all $t > t_0$.*
2) *Given that the sensing policy in item 1 is followed, the optimal access policy for $\mathbf{B}_s \in \mathcal{U}$ is*:

$$\pi^*_{ac}(\mathbf{B}_s) = \begin{cases} \mathbf{B}_s & \text{if } P_{stay} = P_{max} \\ \mathbf{B}_{s+1 \pmod N} & \text{if } P_{switch} = P_{max} \\ \mathbf{B}_{s+2 \pmod N} & \text{if } P_{Dswitch} = P_{max} \end{cases}.$$

3) *The maximal throughput is P_{max}.*

Proof: We first note that for a fixed channel hopping patterns and with $L = 2$, as the channels are permuted in adjacent pairs of channels, w.l.o.g, we can re-enumerate the channels to obtain a cyclic ordering. We will therefore focus in the proof on such an enumeration only for simplifying the notations. We shall refer to the re-enumerated network as a cyclic PU network. In this case, $\mathbf{B}_s = s, \forall s = 0, 1, \ldots, N-1$.

We will show the theorem for $H = 2$. clearly the results hold for all $H > 2$ as well. Setting the history length to $H = 2$, we first analyze the possible observation outcomes $\mathbf{x}(t)$, $\mathbf{x}(t-1)$ and show that the optimal sensing policy π^*_s results in a one-to-one mapping between the observations in the last two time indexes and the current channel state $U(t)$, thus, when $H = 2$ the state space is fully observable:

- When $\mathbf{x}(t) = [l, (-1, 1)]$ or $\mathbf{x}(t) = [l, (1, -1)]$ then, since there is only one free channel, it follows that the current state is $U(t) = l \cdot L$, and $U(t) = l \cdot L + 1$, respectively, irrespective of $\mathbf{x}(t-1)$.
- Consider $\mathbf{x}(t) = [l, (1, -1)]$: We conclude that $U(t) = l \cdot L + 1$, thus, $U(t+1)$ can be either $l \cdot L + 1$, $l \cdot L + 2 \pmod N$, or $l \cdot L + 3 \pmod N$. Hence, in the next time step, $t+1$, the agent should sense the pair of channels in subset $l+1 \pmod{\frac{N}{L}}$, $a_s(t) = l+1 \pmod{\frac{N}{L}}$. Consider the three possible sensing outcomes:
  - If $\mathbf{x}(t+1) = [l+1 \pmod{\frac{N}{L}}, (1, 1)]$ then, $U(t+1) = l \cdot L + 1$.
  - If $\mathbf{x}(t+1) = [l+1 \pmod{\frac{N}{L}}, (-1, 1)]$, then $U(t+1) = l \cdot L + 2 \pmod N$.
  - If $\mathbf{x}(t+1) = [l+1 \pmod{\frac{N}{L}}, (1, -1)]$, then $U(t+1) = l \cdot L + 3 \pmod N$.
- Consider $\mathbf{x}(t) = [l, (-1, 1)]$: It immediately follows that $U(t) = l \cdot L$, hence, $U(t+1)$ can be either $l \cdot L$, $l \cdot L + 1$, or $l \cdot L + 2 \pmod N$. Consequently, at in the next time step, the agent should sense the same pair of channels in subset $a_s(t) = l$:
  - If $\mathbf{x}(t+1) = [l, (-1, 1)]$ then, $U(t+1) = l \cdot L$,
  - If $\mathbf{x}(t+1) = [l, (1, -1)]$, then $U(t+1) = l \cdot L + 1$,
  - If $\mathbf{x}(t+1) = [l, (1, 1)]$, then $U(t+1) = l \cdot L + 2 \pmod N$.
- Consider $\mathbf{x}(t) = [l, (1, 1)]$ and assume that $\mathbf{x}(t-1) = [l, (-1, 1)]$. Then, it is guaranteed that the free channel at time t is $U(t) = l \cdot L + 2 \pmod N$. This state is equivalent to the partial observation $\mathbf{x}(t) = [l+1 \pmod{\frac{N}{L}}, (-1, 1)]$. Therefore, similar to the previous analysis, we obtain that the optimal sensing policy in this case is $a_s(t) = l + 1 \pmod{\frac{N}{L}}$.
- Consider $\mathbf{x}(t) = [l, (1, 1)]$ and assume that $\mathbf{x}(t-1) = [l-1 \pmod{\frac{N}{L}}, (1, -1)]$. Then, $U(t) = (l-1) \cdot L + 1 \pmod N$. This state is equivalent to the partial observation $\mathbf{x}(t) = [l - 1 \pmod{\frac{N}{L}}, (1, -1)]$. Therefore, similarly to the analysis above we obtain that the optimal sensing policy in this case is $a_s(t) = l$.

Let $\mathcal{X}_{init} \triangleq \left\{[l, (x_0, x_1)] \middle| x_0 = -1 \text{ or } x_1 = -1 \text{ and } l \in \{0, 1, \ldots, \frac{N}{L} - 1\}\right\}$ denote the initial set. Under the assumption of $\mathbf{x}(t_0) \in \mathcal{X}_{init}$, for some $t_0 \in \mathbb{N}$, then for all $t > t_0$, any consecutive pair of observations, $\big(\mathbf{x}(t-1), \mathbf{x}(t)\big)$, contains sufficient information to fully determine the network state $U(t)$. Table I summarizes all of the possible observations for $N = 4$ channels, their corresponding full network state, and the sensing policy, where $\pi_s\big(\mathbf{x}(t-1), \mathbf{x}(t)\big) = 0$ denotes that for the given pair of observations at time steps $t-1$ and t, then at time step $t+1$ the SU will sense subset 0 of the network channels, consisting of ch_0, and ch_1, whereas $\pi_s\big(\mathbf{x}(t-1), \mathbf{x}(t)\big) = 1$ denotes that it will sense subset 1 of the network channels consisting of ch_2 and ch_3.

Following this sensing policy, it holds that when $\mathbf{x}(t_0) \in \mathcal{X}_{init}$, the network state is fully observable at each subsequent time step, and hence this sensing policy is necessarily the optimal sensing policy.

If $\mathbf{x}(1) \in \mathcal{X}_{init}$, then $t_0 = 1$ and the proof of first assertion is complete. If $\mathbf{x}(1) \notin \mathcal{X}_{init}$, then we choose sensing and access actions randomly and uniformly.

TABLE I
OPTIMAL SENSING POLICY FOR $N = 4$

$\mathbf{x}(t-1)$	$\mathbf{x}(t)$	$U(t)$	$\pi_s\big(\mathbf{x}(t-1), \mathbf{x}(t)\big)$
don't care	$[0,(-1,1)]$	0	0
don't care	$[0,(1,-1)]$	1	1
don't care	$[1,(-1,1)]$	2	1
don't care	$[1,(1,-1)]$	3	0
$[0,(-1,1)]$	$[0,(1,1)]$	2	1
$[1,(-1,1)]$	$[1,(1,1)]$	0	0
$[1,(1,-1)]$	$[0,(1,1)]$	3	0
$[0,(1,-1)]$	$[1,(1,1)]$	1	1
$[1,(1,1)]$	$[1,(1,1)]$	1	1
$[0,(1,1)]$	$[0,(1,1)]$	3	0
$[0,(1,1)]$	$[1,(1,1)]$	0	0
$[1,(1,1)]$	$[0,(1,1)]$	2	1

The probability that $\mathbf{x}(t) \notin \mathcal{X}_{init}$ for exactly n consecutive time steps (starting from $t = 1$) is given by $\Pr\big(\mathbf{x}(n+1) \in \mathcal{X}_{init}\big) \cdot \prod_{t=2}^{n} \Pr\big(\mathbf{x}(t) \notin \mathcal{X}_{init}\big)$, as the events are statistically independent because the observations $\big\{\mathbf{x}(t)\big\}_{t=2}^{n}$ are observed via randomly and uniformly sensing actions. Note that $\Pr\big(\mathbf{x}(t) \notin \mathcal{X}_{init}\big) = \frac{N/2-1}{N/2} < 1$, $\forall 1 < t \leq n$, thus, $\Pr\big(\mathbf{x}(t) \notin \mathcal{X}_{init}, \forall t \in \mathbb{N}\big) = \lim_{n\to\infty} \Pr\big(\mathbf{x}(n+1) \in \mathcal{X}_{init}\big) \cdot \left(\frac{N/2-1}{N/2}\right)^{n-1} = 0$. It thus follows that the probability that there exists t_0 such that $\mathbf{x}(t_0) \in \mathcal{X}_{init}$ for some $t_0 > 0$ is 1. Once $\mathbf{x}(t_0) \in \mathcal{X}_{init}$ for some $t_0 > 0$, the optimal sensing policy can be applied $\forall t > t_0$, which establishes the first assertion.

As the state is fully observable, finding the best access policy can be formulated as an MDP problem. According to the Bellman optimally equations [14, Ch. 3], the optimal policy π^*_{ac} must satisfy $\pi^*_{ac}(s) = \underset{a\in\mathcal{A}_{ac}}{\text{argmax}}\{q_{\pi^*_{ac}}(s,a)\}$, where $q_{\pi^*_{ac}}(s,a)$ is the action-value function for the optimal policy (i.e., the policy which maximizes the Q-function) at $(s,a) \in \mathcal{U}\times\mathcal{A}_{ac}$. Next, we characterize the optimal access policy: Letting $s' \in \mathcal{U}$ denote the next state and $v_{\pi^*}(s)$ denote the optimal access value function for state s, $v_{\pi^*}(s) \triangleq \max_{a\in\mathcal{A}_{ac}} \big\{q_{\pi^*}(s,a)\big\}$, then $q_{\pi^*_{ac}}(s,a)$ can be computed as:

$$\begin{aligned} q_{\pi^*_{ac}}(s,a) &= \sum_{s'\in\mathcal{U}} \sum_{r\in\{-1,1\}} \Pr(r,s'|s,a)\cdot\big(r+\gamma\cdot v_{\pi^*_{ac}}(s')\big) \\ &= \sum_{s'\in\mathcal{U}} \Pr(s'|s,a)\Bigg(\sum_{r\in\{-1,1\}} \Pr(r|s',s,a) \\ &\qquad\qquad \cdot\big(r+\gamma\cdot v_{\pi^*_{ac}}(s')\big)\Bigg) \\ &\overset{(a)}{=} \gamma\cdot\sum_{s'\in\mathcal{U}} \Pr(s'|s)\cdot v_{\pi^*_{ac}}(s') \\ &\quad + \sum_{s'\in\mathcal{U}} \Pr(s'|s)\cdot\Bigg(\sum_{r\in\{-1,1\}} r\cdot\Pr(r|s',a)\Bigg), \end{aligned} \tag{4}$$

where in step (a), we used the fact that given the next state and the current action, the reward is fully determined, i.e., $\Pr(r|s',s,a) = \Pr(r|s',a)$, and that in the fully-observed case, the action does not affect the probability of the next state given the previous state, i.e., $\Pr(s'|s,a) = \Pr(s'|s)$. Note that in the considered setup, $\Pr(r|s',a)$ is deterministic, i.e.,

$$\Pr(r=1|s',a) = \mathbb{1}(s'=a),\ \Pr(r=-1|s',a) = \mathbb{1}(s'\neq a), \tag{5}$$

where $\mathbb{1}(\mathsf{A})$ stands for the indicator function of the event A. From (5),

$$\begin{aligned} &\sum_{s'\in\mathcal{U}} \Pr(s'|s) \sum_{r\in\{-1,1\}} r\cdot\Pr(r|s',a) \\ &\quad= \Pr(s'=a|s) - \sum_{s'\neq a}\Pr(s'|s) \\ &\quad= \Pr(s'=a|s) - (1-\Pr(s'=a|s)) \\ &\quad= 2\cdot\Pr(s'=a|s) - 1. \end{aligned} \tag{6}$$

Plugging (6) into (4) we obtain the optimal access policy, $\pi^*_{ac}(s) = \underset{a\in\mathcal{A}_{ac}}{\text{argmax}}\big\{q_{\pi^*_{ac}}(s,a)\big\}$, as:

$$\begin{aligned} \pi^*_{ac}(s) &= \underset{a\in\mathcal{A}_{ac}}{\text{argmax}}\Bigg\{\gamma\cdot\sum_{s'\in\mathcal{U}}\Pr(s'|s)\cdot v_{\pi^*_{ac}}(s') \\ &\qquad + \sum_{s'\in\mathcal{U}}\Pr(s'|s)\cdot\Bigg(\sum_{r\in\{-1,1\}} r\cdot\Pr(r|s',a)\Bigg)\Bigg\} \\ &\overset{(a)}{=} \underset{a\in\mathcal{A}_{ac}}{\text{argmax}}\big\{2\cdot\Pr(s'=a|s)-1\big\} \\ &= \underset{a\in\mathcal{A}_{ac}}{\text{argmax}}\big\{\Pr(s'=a|s)\big\} \\ &\overset{(b)}{=} \begin{cases} s & \text{if } P_{stay} = P_{max} \\ s+1 \pmod N & \text{if } P_{switch} = P_{max} \\ s+2 \pmod N & \text{if } P_{Dswitch} = P_{max} \end{cases}, \end{aligned} \tag{7}$$

where (a) follows as the first summand is independent of $a \in \mathcal{A}_{ac}$, and (b) follows from (3). This proves the second assertion.

Let suc_i denote the event of a successful transmission at time step $i \in \mathbb{N}$ and define the throughput for this scenario as $T \triangleq \lim_{t\to\infty} \frac{\sum_{i=1}^{t}\mathbb{1}(suc_i)}{t}$. Following the optimal sensing policy implies that the events $\{\mathbb{1}(suc_i)\}_{i=t_0+1}^{\infty}$ are i.i.d random variables, as the states are fully-observable. Then, according to the weak law of large numbers [43] we have:

$$\begin{aligned} T &= \lim_{t\to\infty}\frac{\sum_{i=1}^{t}\mathbb{1}(suc_i)}{t} \\ &= \lim_{t\to\infty}\frac{\sum_{i=1}^{t_0}\mathbb{1}(suc_i)}{t} + \lim_{t\to\infty}\frac{\sum_{i=t_0+1}^{t}\mathbb{1}(suc_i)}{t} \\ &\overset{(a)}{=} \lim_{t\to\infty}\frac{t-t_0}{t}\frac{\sum_{j=1}^{t-t_0}\mathbb{1}(suc_{j+t_0})}{t-t_0} \\ &\overset{(b)}{=} \mathbb{E}\big\{\mathbb{1}(suc_i)\big\} \\ &= \Pr(suc_i) = P_{max}. \end{aligned} \tag{8}$$

where (a) follows as $0 \leq \sum_{i=1}^{t_0}\mathbb{1}(suc_i) \leq t_0$, hence, $\lim_{t\to\infty}\frac{\sum_{i=1}^{t_0}\mathbb{1}(suc_i)}{t} = 0$; and (b) follows from the weak law

of large numbers [43] and since $\lim_{t\to\infty}\frac{t-t_0}{t}=1$. Hence, $T=\lim_{t\to\infty}\frac{\sum_{i=1}^{t}\mathbb{1}(suc_i)}{t}=P_{max}$, for any $\mathbf{x}(1)=\big[l,\big(x_0(1),x_1(1)\big)\big], l\in\{0,1,\ldots,\frac{N}{L}-1\},\big(x_0(1),x_1(1)\big)\in\mathcal{X}$. ■

C. Discussion and Insights

The analysis in the proof of Thm. 1 implies two important insights: First, from the derivation of (7) we conclude that if the observations can be selected such that the agent can infer the full network state (as in the case analyzed in this section), the discount factor $\gamma\in[0,1)$ can be set arbitrarily. Practically, the best option will be to set $\gamma=0$, since, for example, in the Q-learning algorithm the update rule (2) simplifies to $Q(s,a)\leftarrow(1-\alpha)\cdot Q(s,a)+\alpha\cdot r$. As a result, the algorithm converges faster due to the elimination of the unnecessary term $\gamma\cdot\max_{a'\in\mathcal{A}}\{Q(s',a')\}$ from the update rule. This follows as the term $\gamma\cdot\max_{a'\in\mathcal{A}}\{Q(s',a')\}$ includes an estimate of the function $q_{\pi^*}(s',.)$ which may be very different from its true value at the beginning of the learning process, and consequently, using $\gamma>0$ adds unnecessary noise to the update rule. The second insight we obtain from our analysis is that for the partially-observed state space corresponding to the general problem setup in Sec. II, the proposed algorithm requires $\gamma>0$ to converge to the maximal throughput, which makes the suggested problem formulation *a non-degenerated RL problem* in the sense that actions are selected such that the accumulated *future* reward is maximized and not only the immediate reward. The reason is that in general, the full network state cannot be determined from *a finite number of past observations*. Furthermore, even if the full network state can be determined from a finite number of past observations, then at the beginning of the learning process, the sensing policy is arbitrary. Then, the agent cannot infer about the channel state at the beginning. This implies that the agent must consider the effect of selecting which channels to sense on future rewards thereby improving its sensing policy.

V. Experiments

In this section, we report the outcomes of the experiments carried out to test and evaluate the performance of the proposed DDQSA algorithm. DDQSA was implemented as described in Algorithm 1 in Section IV, with two hidden layers of fully connected DNNs, where each layer consists of 128 neurons with the rectified linear unit (ReLU) activation function, $ReLU(x)=\max\{0,x\}$. The activation function for each neuron in the output layer is the linear activation function $f(x)=x$. The ϵ-greedy policy has been applied such that $\epsilon(\tilde{t})=\frac{1}{1+\xi\cdot\tilde{t}}$, i.e., $\epsilon(\tilde{t})$ decays as the number of time slots in which the SU has data to send, increases. The rate in which $\epsilon(\tilde{t})$ decreases as function of $\tilde{t}$ is determined by the exploration decay rate ξ. At each time step, a mini-batch of 64 samples, $|\mathcal{M}_B|=64$, from the replay buffer is uniformly sampled and used for training. The Adam algorithm [44] is used as the optimizer with the smooth L1 loss function [45]. We set the discount factor to $\gamma=0.8$, the learning rate is $\alpha=10^{-4}$, the replay buffer capacity is $C=30000$, and the update rate is $J=20$. We define the relative throughput $\rho(\tau)\triangleq\frac{\eta(\tau)}{\eta_{bound}(\tau)}$, $\tau\in\mathbb{N}$, where $\eta(\tau)$ is the number of successful transmissions in the range of time steps beginning from $(\tau-1)\cdot100+1$ up to time step $\tau\cdot100$ divided by 100, and $\eta_{bound}(\tau)$ is defined as the number of time steps in which at least one channel was free, in the range of time steps beginning from $(\tau-1)\cdot100+1$ up to $\tau\cdot100$, divided by 100. Thus, $\eta_{bound}(\tau)$ is an upper bound on the throughput of any DSA algorithm for the setup defined in Section II. In the following, $\rho(\tau)$ is used as the figure-of-merit for evaluating the performance of the algorithms.

In the experiments, we compare the performance of DDQSA with that of three other algorithms with fixed sensing or access policies:

1) Random Access: In this policy, the SU does not employ sensing, and at each time step selects randomly and uniformly a channel for accessing.
2) Random Sensing: In this policy, the SU randomly selects a subset of channels to sense, and uses these observations to learn an access policy by employing a DDQN.
3) Alternating Sensing: In this policy, the SU senses each of the subsets of channels alternately, as in [26], and uses these observations to learn an access policy by applying a DDQN.

The performance of the different algorithms was obtained by averaging the outcomes of 30 independent experiments for each algorithm at each scenario.

A. Comparing DDQSA With the Optimal Policy for an FHPD Network

In this section we consider a FHPD network defined in Section IV with $N=10$ channels and an observation subset size of $L=2$ channels, and ideal sensing (no sensing errors). In this network there is always a single free channel at each time step, hence, $\eta_{bound}(\tau)=1,\forall\tau\in\mathbb{N}$, and $\rho(\tau)=\eta(\tau)$. As obtained in Thm. 1, the throughput of the optimal access and sensing policies for this network is P_{max}. In this experiment we set $P_{stay}=0.1, P_{switch}=0.1$, and $P_{Dswitch}=0.8$, which implies that for $\tau\gg1$, $\rho(\tau)=\eta(\tau)\approx P_{max}=P_{Dswitch}=0.8$. The throughput of the random access algorithm can be analytically obtained as $\rho_{RA}(\tau)=0.1$ (independent of the probabilities P_{stay}, P_{switch}, and $P_{Dswitch}$), which follows by noting that there are 10 channels, where at each time-step there is a single free channel. Fig. 2 depicts the simulation results for this scenario. We observe that the DDQSA algorithm performs well and asymptotically attains near-optimal performance (the throughput is numerically evaluated at approximately 0.75), whereas under alternating sensing policy, the throughput is about 0.28 and under the random sensing policy it is about 0.36, both are highly sub-optimal. We conclude that *DDQSA is indeed capable of learning a near-optimal joint sensing and access policy*, where optimality requires a non-trivial combination of sensing and access, thereby justifying the rationale of our proposed approach. It is emphasized that with the common sensing approach of [26], the DDQN algorithms achieve half the throughput of DDQSA, again indicating to the effectiveness of the proposed approach.

TABLE II
STOCHASTIC PUS STATE TRANSITION PROBABILITIES

	$P_i(0\|0)$	$P_i(0\|1)$	$P_i(0\|2)$	$P_i(0\|3)$	$P_i(0\|4)$	$P_i(0\|5)$	$P_i(0\|6)$	$P_i(0\|7)$
$i=4$	0.1	0.1	0.15	1	-	-	-	-
$i=5$	0.04	0.2	0.1	0.12	0.08	1	-	-
$i=6$	0.15	0.18	0.3	0.1	1	-	-	-
$i=7$	0.19	0.2	0.02	0.15	0.1	0.17	1	-
$i=8$	0.1	0.05	0.02	0.07	0.1	0.1	0.2	1
$i=9$	0.1	0.11	0.02	0.11	0.01	1	-	-

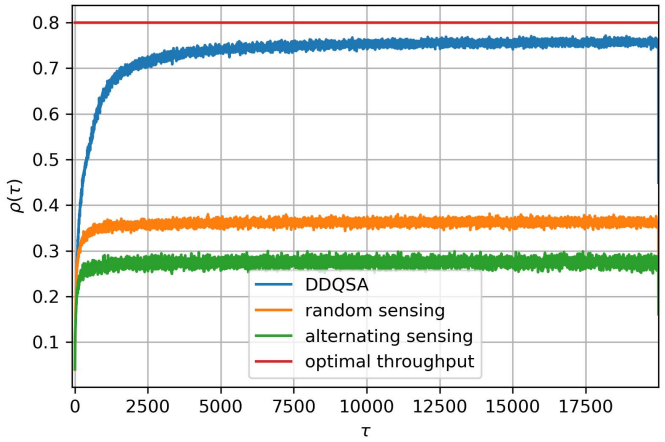

Fig. 2. Relative throughputs for the FHPD network defined in Section IV, for $P_{stay} = 0.1$, $P_{switch} = 0.1$, and $P_{Dswitch} = 0.8$.

B. Experiment Results for General Scenarios

We consider now a network consisting of $N = 10$ channels, with 10 PUs, $K_p = 10$, observations subsets of size $L \in \{2, 5\}$, a history length of $H = 6$, and ideal sensing (no sensing errors). Sensing with $L = 5$ is referred to as *wideband sensing*, as half the channels in the network are sensed at each time step, and sensing with $L = 2$ is referred to as *narrowband sensing*. It is assumed that $\{\text{pu}_i\}_{i=0}^{3}$ are *legacy* PUs, i.e., each PU occupies one fixed channel at each time step, see e.g., [26]. Specifically, pu_i occupies ch_i for $i = 0, \ldots, 3$. The other 6 PUs, denoted $\{\text{pu}_i\}_{i=4}^{9}$, referred to as *stochastic* PUs, transmit in frames which may span more than one time step according to the Markov model described in Section II. As $\{\text{ch}_i\}_{i=0}^{3}$ are occupied by legacy PUs, the stochastic PUs can transmit over channels $\{\text{ch}_i\}_{i=4}^{9}$, which is a situation which clearly impacts the access policy. The stochastic PUs' frame lengths are modelled as finite-memory Markov chains with respective maximal lengths of $M_4 = 3, M_5 = 5, M_6 = 4$, $M_7 = 6$, $M_8 = 7$, and $M_9 = 5$. The transition probabilities of the Markov chains for all the stochastic PUs are summarized in Table II.

In the following simulations, we consider three PU access policies:

- In PU policy 1, each stochastic PU can access a single, fixed, pre-determined channel at any time step, i.e., pu_i can access only ch_i, whenever it needs to transmit.
- In PU policy 2, we set the stochastic PUs access policy as follows:
  - Once a stochastic PU begins transmitting at a given channel, it will transmit its entire frame over that channel, e.g., if channel ch_j was allocated to pu_i when $m_i = 1$, then channel ch_j will be allocated to pu_i until $m_i = 0$, at which time this channel allocation is terminated and the channel becomes free.
  - When a new stochastic PU, e.g., pu_i begins to transmit at a given time step ($m_i = 1$), it will use channel ch_k for transmission, where ch_k is the free channel with the minimal index k. This can be justified by an ordering of channels according to some measure of quality, e.g., signal-to-noise ratio (SNR), where a channel with a larger SNR is assigned a smaller index.
  - If several stochastic PUs begin to transmit at the same time step, the PU with the smaller index will use the free channel with the smaller index for transmission. For example, if at some time step, both pu_i, and pu_j, $i < j$, begin to transmit, and ch_k, ch_l, $k < l$, are free, then pu_i will transmit on ch_k, and pu_j will transmit on ch_l. This represents a priority assignment where the user with the higher priority has a smaller index, resulting in allocation of better channels.
- In PU policy 3, the PUs follow the same policy as described for PU policy 2, but at every even time step, all the 10 channels are flipped, i.e., every 2 time steps, ch_i will switch with ch_{9-i} for $i = 0, 1, \ldots, 4$, which corresponds to a frequency hopping network.

First, we note that the throughput of the random access algorithm can be evaluated numerically irrespective of the PU access policy. This follows as the random access algorithm does not apply learning. Then, we let the PUs transmit according to the statistical Markov chain models detailed in Table II for sufficiently long time and randomly select channel for SU access at each time step. Applying this computation for $1.5 \cdot 10^6$ time steps and averaging the resulting throughput we numerically obtain the throughput of the random access algorithm as $\rho_{RA}(\tau) \approx 0.189$, $\forall \tau \in \mathbb{N}$, irrespective of the PU access policy.

Fig. 3 depicts the relative throughputs of the different algorithms for PU policy 1, for $L = 5$, and $L = 2$, in Figs 3a and 3b, respectively. It can be observed from the figure that for both narrowband and wideband sensing, the DDQSA outperforms the other sensing and access policies. In particular, DDQSA outperforms the commonly used alternating sensing, even in PU policies with simple PU

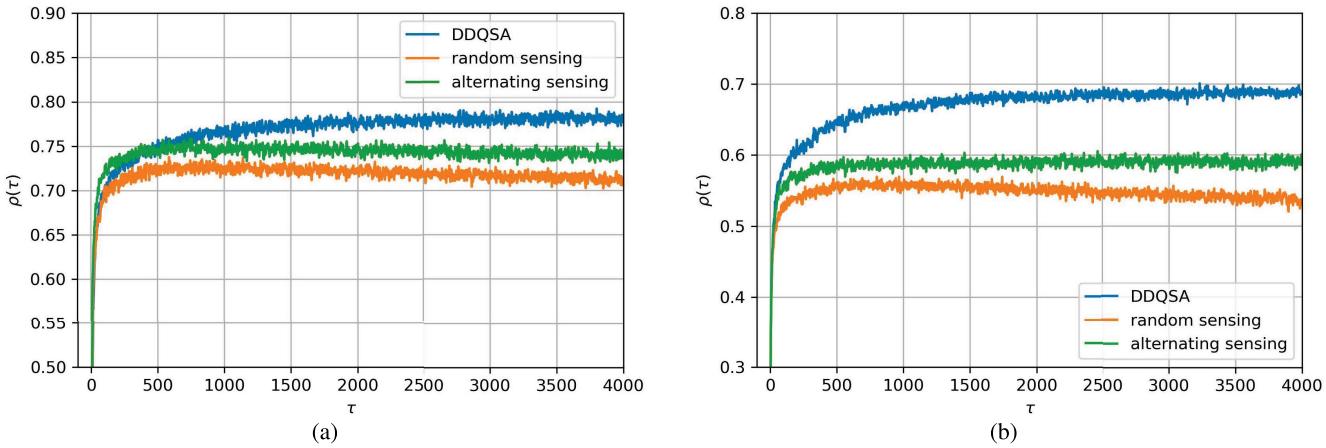

Fig. 3. Relative throughputs of the different algorithms for PU policy 1 (a) with $L = 5$, and (b) with $L = 2$.

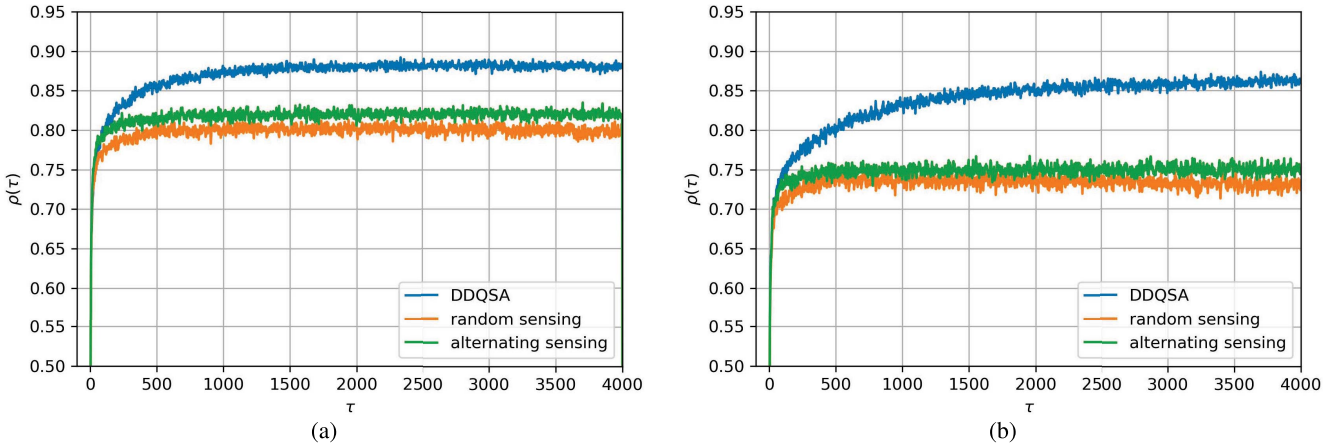

Fig. 4. Relative throughputs of the different algorithms for PU policy 2 (a) with $L = 5$ and (b) with $L = 2$.

access policy, namely, each PU accesses a fixed, predetermined channel. Another conclusion from the simulations is that when the PUs use a simple access policy, and the SU applies wideband sensing, then alternating sensing has nearly the same performance as DDQSA. We also observe that indeed wideband sensing facilitates better performance than narrowband sensing, yet it is important to note that that the performance loss when the SU uses the more practical narrowband sensing, compared to wideband sensing, is much smaller for DDQSA (approximately 11%) than the performance loss with alternating sensing (approximately 21%) and the performance loss due for random sensing (approximately 25%). This clearly indicates the substantial benefit of sensing optimization in DSA, as this allows the SU to focus its limited sensing on the channels which contain the most relevant information for access decisions. Fig. 4 depicts the relative throughputs of the different algorithms for PU policy 2, where the throughput with $L = 5$ is depicted in Fig. 4a, and the throughput with $L = 2$ is depicted in Fig. 4b. Observe that for PU policy 2, the throughput achieved by the DDQSA algorithm with both $L = 5$ and $L = 2$ is clearly superior to that achieved by the other three algorithms and is about 0.88 and 0.86, respectively. Comparing the throughput for wideband and narrowband sensing we observe that, the DDQSA algorithm has a negligible performance degradation for narrowband sensing, yet the performance gap between the DDQSA and the reference algorithms is larger for narrowband sensing than for wideband sensing, approximately 14% and 8%, respectively. This experiment provides further demonstration of the substantial benefits carried by our newly proposed approach: Due to cleverly selecting channels for sensing, the DDQSA is able to maintain nearly the same throughput with a smaller sensing bandwidth. As the reference algorithm do not optimize their sensing policy, then decreasing the sensing bandwidth causes a significant performance loss due to insufficient information for making channel access decisions.

Lastly, Fig. 5 depicts the relative throughputs for PU policy 3, with $L = 5$ depicted in Fig. 5a and $L = 2$ depicted in Fig. 5b. Comparing wideband and narrowband sensing for PU policy 3 we observe that the performance of DDQSA remain the same and the relative throughput is approximately 0.85. Additionally, for both sensing bandwidths DDQSA is superior to the alternating sensing and the random sensing algorithms. yet, it is clearly evident from the figures that the gap between the DDQSA and the reference algorithms is much larger for narrowband sensing than for wideband, which provides an

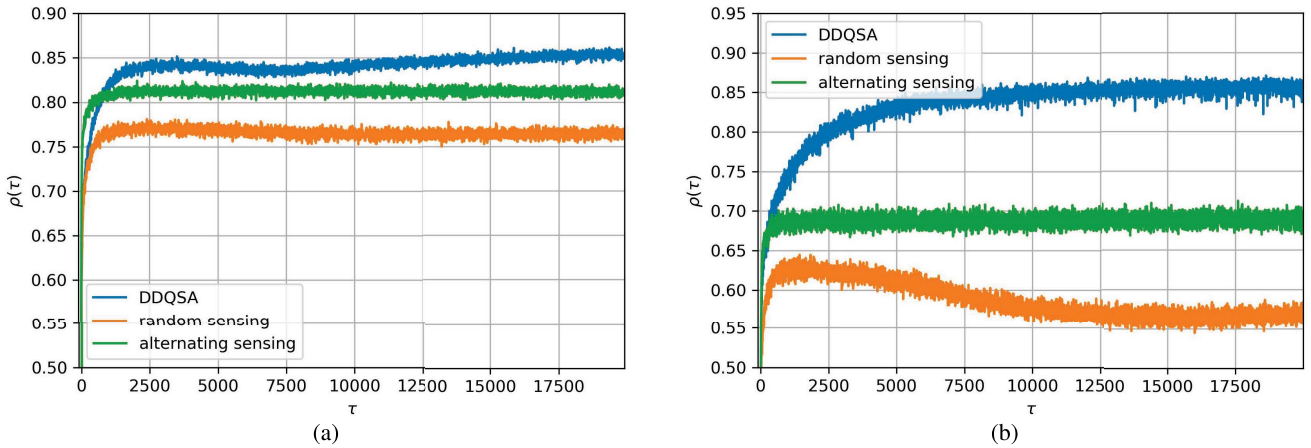

Fig. 5. Relative throughputs of the different algorithms for PU policy 3 (a) with $L = 5$ and (b) with $L = 2$.

additional evidence to the strength of this algorithm in the challenging environment of hopping channels. In particular, letting the agent learn the optimal sensing strategy allows achieving the same throughput with a smaller sensing bandwidth, which is a very important advantage, moreover so as with the commonly used alternating sensing, a large bandwidth is required for the higher throughput.

In summary, the results clearly demonstrate that the DDQSA is able to learn a sensing policy and correspondence access policy which improve the throughput compared to previously proposed approaches. An interesting phenomenon that we discovered is that optimizing both sensing and access leads to a sensing policy which is able to obtain the necessary information for maximizing the throughput with a narrower sensing bandwidth than needed for the same performance with deterministic sensing patterns.

C. Impact of SU's Random Transmission Requests

In the above simulations it was assumed that the SU accesses the channel at each time step, and thus receives an ACK/NACK signal at every time step. To accommodate the practical situation in which the SU may have idle times, we let the SU accesses the channel with probability $P_{ac} < 1$, and remain idle with probability $P_{nac} = 1 - P_{ac} > 0$. Fig. 6 compares the relative throughput of the DDQSA algorithm for the FHPD when the SU transmits with three different probabilities: $P_{ac} = 1$, $P_{ac} = 0.7$ and $P_{ac} = 0.2$.

From Fig. 6 it is observed that when convergence is achieved, the relative throughput when $P_{ac} = 0.7$ and $P_{ac} = 0.2$ are very close to the relative throughput when the SU transmits at every time step. When the SU transmits with a small probability, e.g., $P_{ac} = 0.2$, $\rho(\tau)$ is noisier because $\rho(\tau)$ involves an averaging operation over 100 time-steps, and consequently, with smaller transmission probability there are fewer transmissions in that interval. In addition, it is observed that as P_{ac} decreases, the convergence rate becomes slower due to the fact that the replay buffer is updated less frequently when the SU transmits infrequently. However, this degradation in convergence rate is graceful w.r.t the transmission probability. For example, with $P_{ac} = 0.7$, the convergence rate

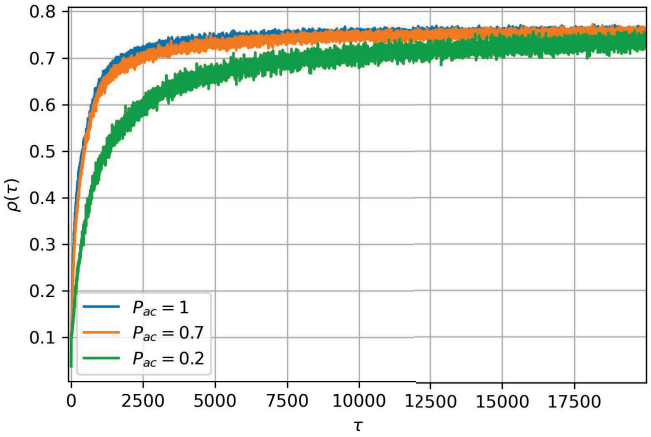

Fig. 6. Relative throughputs for the FHPD network for three different access probabilities: $P_{ac} = 1$, $P_{ac} = 0.7$, and $P_{ac} = 0.2$.

is almost the same as with $P_{ac} = 1$. We conclude that the DDQSA algorithm's steady-state performance is not affected by the SU transmission probability and the convergence duration is slightly increased when as SU transmissions become infrequent.

D. Dealing With Sensing Errors

Next, we examine the impact of sensing errors on the performance of our novel DDQSA algorithm. To that aim, we consider two situations with imperfect sensing:

- Sensing errors: At any time step, each channel can have a sensing error with probability 0.1.
- Undetermined sensing: At any time step, the sensing outcome of each channel can be undetermined with probability 0.1, or correct with probability 0.9.

Fig. 7 compares the performance of the DDQSA algorithm with perfect sensing, undetermined sensing, and sensing errors, for PU policy 2 with $L = 2$, corresponding to narrowband sensing. It is observed that when 10% of the sensing results are in error, the throughput decreases by 13%. However, when the sensor does not declare a sensing outcome when the probability of sensing errors is high, and instead marks

TABLE III
STOCHASTIC PUS WITH INDEXES 10, 11, 16, 17, 18, 19 STATE TRANSITION PROBABILITIES

	$P_i(0\|0)$	$P_i(0\|1)$	$P_i(0\|2)$	$P_i(0\|3)$	$P_i(0\|4)$	$P_i(0\|5)$	$P_i(0\|6)$	$P_i(0\|7)$	$P_i(0\|8)$
$i=10$	0.05	0.1	0.09	0.04	0.04	0.13	1	-	-
$i=11$	0.04	0.12	0.025	0.1	0.08	0.01	1	-	-
$i=16$	0.17	0.18	0.2	0.1	0.034	0.05	0.25	0.04	1
$i=17$	0.19	0.03	0.2	0.1	0.01	0.08	1	-	-
$i=18$	0.1	0.05	0.02	0.07	0.1	0.01	0.04	1	-
$i=19$	0.15	0.1	0.02	0.1	0.01	0.1	0.2	1	-

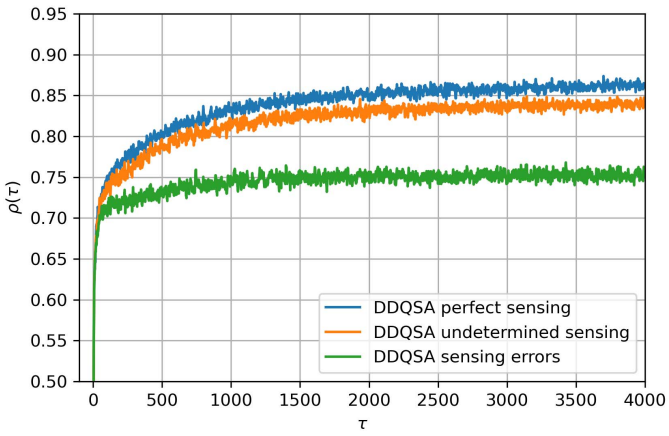

Fig. 7. Relative throughputs of the DDQSA with perfect sensing, with undetermined sensing with probability 0.1, and with error probability of 0.1 for PU policy 2, with $L = 2$.

these readings as undetermined, the throughput is only slightly reduced compared with the case of perfect sensing (only a 2% decrease). We conclude that the DDQSA is quite robust to sensing errors, exhibiting a gradual performance degradation, and in particular, if soft sensing is used to decrease the probability of sensing errors, DDQSA performance almost do not degrade.

E. Distributed Channel Access via the DDQSA Algorithm in a Practical Deployment

In this section we test the performance of the proposed DDQSA algorithm in an extended network scenario whose parameters are largely based on the 8802.11-2018 standard [46], which is the evolution of the 801.11ah standard [47], commonly accepted a baseline for implementing cognitive radio networks, see, e.g., [48] and [49]. The extended network scenario consists of $K_p = 20$ PUs and 2 SUs, sharing a total of $N = 20$ channels, each has a bandwidth of 1 [MHz]. PUs and SUs were randomly placed in a 2.5 [Km] $\times$ 2.5 [Km] square area, according to a uniform distribution, and the 2-ray propagation model was applied to evaluate the power of the signals received at the SUs. It is assumed that every node has an isotropic antenna (i.e., an antenna gain of 1) whose height is 1.5 [m]. In the tested network scenario, the PUs may access the entire set of N channels according to their predetermined channel access policy. SU_1 was assigned the channels with indexes in the set $\mathcal{N}_1 \triangleq \{0, 1, 2, \ldots, N/2-1\}$, and SU_2 was assigned the channels with indexes in the set $\mathcal{N}_2 \triangleq \{N/2, N/2+1, \ldots, N-1\}$. Each SU may access only its assigned channel set for sensing and access: Each SU applies sensing to its assigned channels and, when it needs to transmit, it selects one channel from its set of assigned channels for access. The sensing action at each SU uses an observation subset size of $L = 2$ channels, i.e., *narrowband sensing*. We set the time-slot duration to 1.1 [msec], [47, Pg. 157], and assuming a sampling rate of 1 mega samples per second per sensed channel, then, at each time-slot an SU accumulates $N_{\text{samp}} = 1100$ samples per sensed channel. The transmit power of each node is 20 [dBm], and receiver sensitivity is $\sigma_u^2 = -95$ [dBm], see [50]. For the sensing action, the SU implements an energy detector (ED) with threshold at $\xi = -90$ [dBm]. The ED processes the N_{samp} samples collected from each sensed channel for making a busy/free decision for that channel, see, e.g., [51] and [52]. It is also assumed that ACK/NACK feedback messages may be erroneous with a probability of 5%, see, e.g., [53] and [54]. We note that ACK/NACK errors were not considered in simulations in previous works, e.g., [21], [25], and [26]. In [20], there is a brief discussion on the impact of such errors without a simulation test. We set PUs with indexes $0, 1, 2, 3, 12, 13, 14, 15$ to be legacy PUs, and the remaining PUs to be stochastic PUs. The transition probabilities of the Markov chains for the stochastic PUs with indexes $4-9$ are summarized in Table II, in section V-B. The transition probabilities of the Markov chains for the stochastic PUs with indexes $10, 11, 16-19$ are summarized in Table III.

The performance were evaluated with two PU policies: PU policy 1 and PU policy 3, via the relative throughputs of SU_1 and SU_2. The relative throughput results for SU_1 and SU_2 for PU policy 1 and PU policy 3 are depicted in Figs. 8 and 9, respectively. It is observed that the performance of the different algorithms, including DDQSA, are not affected by a 5% ACK/NACK error rate, cf. Figs. 3b and 8a. It is also observed that the throughputs achieved with the distributed application of the DDQSA algorithm are considerably higher than the throughputs obtained by the reference schemes: For PU policy 1, the throughput of SU_1 is 30% higher than the throughput obtained with random sensing while for SU_2 it is and 43% higher, whereas for alternating sensing SU_1 achieves a 15% increase in throughput and SU_2 achieves a 30% increase in the throughput. For PU policy 3 both SU_1 and SU_2 achieve a throughput increase of 11% over random sensing and 6% over alternating sensing. This follows as PU hopping results in each SU observing the same PU statistics over its assigned channels. The results clearly demonstrate the applicability of

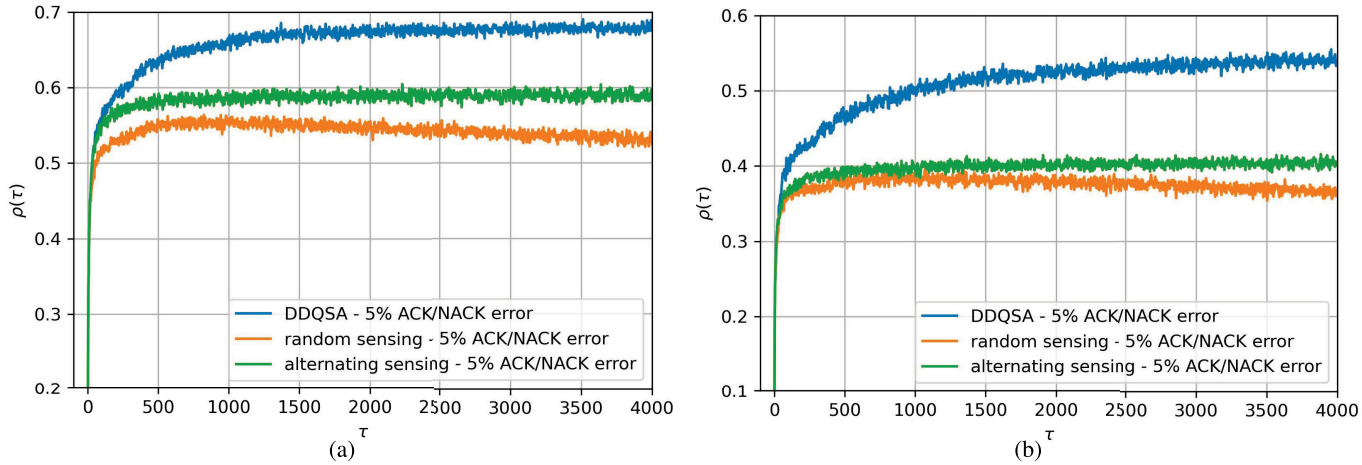

Fig. 8. Relative throughputs of the different algorithms for PU policy 1 with $L = 2$ for (a) SU_1 and (b) SU_2.

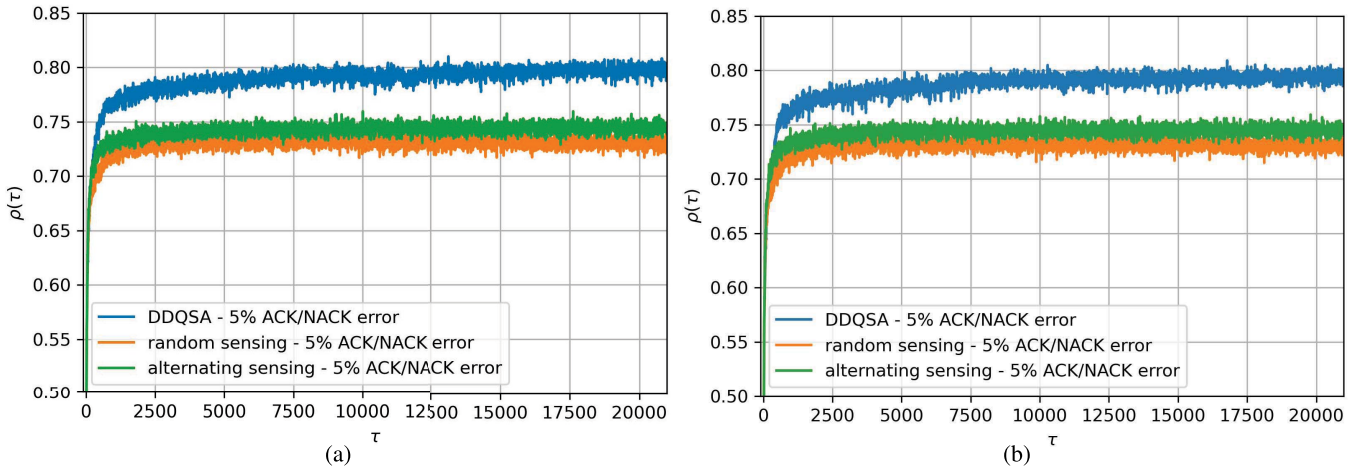

Fig. 9. Relative throughputs of the different algorithms for PU policy 3 with $L = 2$ for (a) SU_1 and (b) SU_2.

the proposed scheme to practical scenarios and the fact that it is readily extendable to distributed implementation at multiple SUs, as well as *its robustness to ACK/NACK errors*. Thus, the extended simulation provides a strong evidence to the effectiveness and robustness of the proposed DDQSA algorithm, and demonstrates that the novel elements introduced in this work have a significant impact on the performance of DSA algorithms.

VI. Conclusion

We considered the DSA problem, where multiple PUs access a network according to a predetermined policy and a cognitive SU, which has no prior knowledge about the PUs dynamics and the access policy they use, attempts to access the channel. In order to successfully transmit, the SU has to determine which channels will be free at the next time step. To that aim, the SU is capable of sensing a subset of the network's channels at each time step, thus it does not know the current state of the entire network. This raises the important and unanswered question of which channels should be sensed at each time step such that the network throughput achieved by a matching access policy is maximal. To identify the SU policy which maximizes its throughput, we developed a novel DDQSA algorithm, which aims to determine the best sensing strategy and the corresponding best access strategy, based on past observations collected by the SU via online learning. We compared the throughput of the proposed DDQSA algorithm with that of three other algorithms which use pre-determined sensing and access policies for four different PU policies. The results showed that DDQSA outperforms the baseline algorithms in all cases. Furthermore, our results show that the proposed approach facilitates decreasing the sensing bandwidth without decreasing the throughput as the SU can focus its sensing on the relevant channels for making access decision for the next time step. Moreover, for the FHPD network, we analytically derived the optimal sensing and access policies and the corresponding maximal throughput. We also demonstrated that the DDQSA algorithm can be applied to more practical scenarios in which the SU does not transmit at every time step. These results clearly demonstrate the ability of DDQSA to learn near-optimal policies and the overall superiority of the proposed approach over existing methods. Finally, the robustness of DDQSA to ACK/NACK error was demonstrated as well as its applicability to distributed implementation when operating in a large network scenario.

REFERENCES

[1] Q. Zhao and B. M. Sadler, "A survey of dynamic spectrum access," *IEEE Signal Process. Mag.*, vol. 24, no. 3, pp. 79–89, May 2007.

[2] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, "Next generation/dynamic spectrum access/cognitive radio wireless networks: A survey," *Comput. Netw.*, vol. 50, pp. 2127–2159, Sep. 2006.

[3] N. Luong et al., "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3133–3174, 4th Quart., 2019.

[4] S. H. A. Ahmad, M. Liu, T. Javidi, Q. Zhao, and B. Krishnamachari, "Optimality of myopic sensing in multichannel opportunistic access," *IEEE Trans. Inf. Theory*, vol. 55, no. 9, pp. 4040–4050, Sep. 2009.

[5] K. Liu and Q. Zhao, "Indexability of restless bandit problems and optimality of whittle index for dynamic multichannel access," *IEEE Trans. Inf. Theory*, vol. 56, no. 11, pp. 5547–5567, Nov. 2010.

[6] C. Tekin and M. Liu, "Online learning of rested and restless bandits," *IEEE Trans. Inf. Theory*, vol. 58, no. 8, pp. 5588–5611, Aug. 2012.

[7] H. Liu, K. Liu, and Q. Zhao, "Learning in a changing world: Restless multiarmed bandit with unknown dynamics," *IEEE Trans. Inf. Theory*, vol. 59, no. 3, pp. 1902–1916, Mar. 2013.

[8] C. Tekin and M. Liu, "Approximately optimal adaptive learning in opportunistic spectrum access," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 1548–1556.

[9] J. Oksanen and V. Koivunen, "An order optimal policy for exploiting idle spectrum in cognitive radio networks," *IEEE Trans. Signal Process.*, vol. 63, no. 5, pp. 1214–1227, Mar. 2015.

[10] K. Cohen, Q. Zhao, and A. Scaglione, "Restless multi-armed bandits under time-varying activation constraints for dynamic spectrum access," in *Proc. 48th Asilomar Conf. Signals, Syst. Comput.*, Nov. 2014, pp. 1575–1578.

[11] S. Bagheri and A. Scaglione, "The restless multi-armed bandit formulation of the cognitive compressive sensing problem," *IEEE Trans. Signal Process.*, vol. 63, no. 5, pp. 1183–1198, Mar. 2015.

[12] T. Gafni and K. Cohen, "Learning in restless multiarmed bandits via adaptive arm sequencing rules," *IEEE Trans. Autom. Control*, vol. 66, no. 10, pp. 5029–5036, Oct. 2021.

[13] T. Gafni and K. Cohen, "Distributed learning over Markovian fading channels for stable spectrum access," 2021, *arXiv:2101.11292*.

[14] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[15] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 279–292, May 1992.

[16] Y. Li, H. Ji, X. Li, and C. M. Leung, "Dynamic channel selection with reinforcement learning for cognitive WLAN over fiber," *Int. J. Commun. Syst.*, vol. 25, no. 8, pp. 1077–1090, Mar. 2012.

[17] P. Venkatraman, B. Hamdaoui, and M. Guizani, "Opportunistic bandwidth sharing through reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 59, no. 6, pp. 3148–3153, Jul. 2010.

[18] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[19] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, 2017, pp. 257–265.

[20] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access in wireless networks," *IEEE Trans. Cogn. Commun. Netw.*, vol. 4, no. 2, pp. 257–265, Jun. 2018.

[21] H. Q. Nguyen, B. T. Nguyen, T. Q. Dong, D. T. Ngo, and T. A. Nguyen, "Deep Q-learning with multiband sensing for dynamic spectrum access," in *Proc. IEEE Int. Symp. Dyn. Spectr. Access Netw. (DySPAN)*, Oct. 2018, pp. 1–5.

[22] C. Zhong, Z. Lu, M. C. Gursoy, and S. Velipasalar, "Actor-critic deep reinforcement learning for dynamic multichannel access," in *Proc. IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, Nov. 2018, pp. 599–603.

[23] Y. Xu, J. Yu, and R. M. Buehrer, "Dealing with partial observations in dynamic spectrum access: Deep recurrent Q-networks," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Oct. 2018, pp. 865–870.

[24] O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for distributed dynamic spectrum access," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 310–323, Jan. 2019.

[25] C. Zhong, Z. Lu, M. C. Gursoy, and S. Velipasalar, "A deep actor-critic reinforcement learning framework for dynamic multichannel access," *IEEE Trans. Cogn. Commun. Netw.*, vol. 5, no. 4, pp. 1125–1139, Dec. 2019.

[26] Y. Xu, J. Yu, and R. M. Buehrer, "The application of deep reinforcement learning to distributed spectrum access in dynamic heterogeneous environments with partial observations," *IEEE Trans. Wireless Commun.*, vol. 19, no. 7, pp. 4494–4506, Jul. 2020.

[27] H. Zhang, N. Yang, W. Huangfu, K. Long, and V. C. M. Leung, "Power control based on deep reinforcement learning for spectrum sharing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 6, pp. 4209–4219, Jun. 2020.

[28] J. Tan, Y.-C. Liang, L. Zhang, and G. Feng, "Deep reinforcement learning for joint channel selection and power control in D2D networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1363–1378, Feb. 2021.

[29] D. Livne and K. Cohen, "PoPS: Policy pruning and shrinking for deep reinforcement learning," *IEEE J. Sel. Topics Signal Process.*, vol. 14, no. 4, pp. 789–801, May 2020.

[30] S. Liu, J. Wu, and J. He, "Dynamic multichannel sensing in cognitive radio: Hierarchical reinforcement learning," *IEEE Access*, vol. 9, pp. 25473–25481, 2021.

[31] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum, "Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 3675–3683.

[32] Y. Li, W. Zhang, C.-X. Wang, J. Sun, and Y. Liu, "Deep reinforcement learning for dynamic spectrum sensing and aggregation in multi-channel wireless networks," *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 2, pp. 464–475, Jun. 2020.

[33] U. Challita, L. Dong, and W. Saad, "Proactive resource management for LTE in unlicensed spectrum: A deep learning perspective," *IEEE Trans. Wireless Commun.*, vol. 17, no. 7, pp. 4674–4689, Jul. 2018.

[34] R. Mennes, F. A. P. De Figueiredo, and S. Latré, "Multi-agent deep learning for multi-channel access in slotted wireless networks," *IEEE Access*, vol. 8, pp. 95032–95045, 2020.

[35] P. M. Pawar and A. Leshem, "Distributed deep reinforcement learning for collaborative spectrum sharing," 2021, *arXiv:2104.02059*.

[36] L. Buşoniu, R. Babuška, and B. De Schutter, "Multi-agent reinforcement learning: An overview," in *Innovations in Multi-Agent Systems and Applications—1*. Berlin, Germany: Springer, 2010, pp. 183–221.

[37] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of Markov decision processes," *Math. Oper. Res.*, vol. 12, no. 3, pp. 441–450, 1987.

[38] H. Hasselt, "Double Q-learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 23, 2010, pp. 2613–2621.

[39] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI Conf. Artif. Intell.*, 2016, vol. 30, no. 1, pp. 1–7.

[40] F. Wunsch et al., "DySPAN spectrum challenge: Situational awareness and opportunistic spectrum access benchmarked," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 3, pp. 550–562, Sep. 2017.

[41] L. Kleinrock and F. Tobagi, "Packet switching in radio channels: Part I—Carrier sense multiple-access modes and their throughput-delay characteristics," *IEEE Trans. Commun.*, vol. COM-23, no. 12, pp. 1400–1416, Dec. 1975.

[42] M. A. Nielsen, *Neural Networks and Deep Learning*, vol. 25. San Francisco, CA, USA: Determination Press, 2015.

[43] J. Tabak, *Probability and Statistics: The Science of Uncertainty*. New York, NY, USA: Facts on File, 2011.

[44] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, May 2015, pp. 1–15.

[45] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[46] *ISO/IEC/IEEE International Standard—Information Technology—Telecommunications and Information Exchange Between Systems—Local and Metropolitan Area Networks—Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Sub 1 GHz License Exempt Operation*, Standard ISO/IEC/IEEE 8802-11:2018/Amd.2:2019(E), 2019, pp. 1–596.

[47] *IEEE Standard for Information Technology—Telecommunications and Information Exchange Between Systems—Local and Metropolitan Area Networks—specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Sub 1 GHz License Exempt Operation*, IEEE Standard 802.11ah-2016 (Amendment to IEEE Standard 802.11-2016, as Amended by IEEE Standard 802.11ai-2016), 2017, pp. 1–594.

[48] M. Shafiq et al., "Multiple access control for cognitive radio-based IEEE 802.11ah networks," *Sensors*, vol. 18, no. 7, p. 2043, 2018.

[49] M.-C. Kim and Y.-T. Kim, "Smart control for energy efficient networking of IEEE 802.11ah-based IoT," in *Proc. 20th Asia–Pacific Netw. Oper. Manag. Symp. (APNOMS)*, Matsue, Japan, Sep. 2019, pp. 1–6.
[50] N. Andrade, P. Toledo, G. Guimaraes, H. Klimach, H. Dornelas, and S. Bampi, "Low power IEEE 802.11ah receiver system-level design aiming for IoT applications," in *Proc. 30th Symp. Integr. Circuits Syst. Design Chip Sands (SBCCI)*, 2017, pp. 11–16.
[51] Y.-C. Liang, Y. Zeng, E. C. Y. Peh, and A. T. Hoang, "Sensing-throughput tradeoff for cognitive radio networks," *IEEE Trans. Wireless Commun.*, vol. 7, no. 4, pp. 1326–1337, Apr. 2008.
[52] A. T. Hoang, D. T. C. Wong, and Y.-C. Liang, "Design and analysis for an 802.11-based cognitive radio network," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Budapest, Hungary, Apr. 2009, pp. 1–6.
[53] M. Woltering, D. Wubben, A. Dekorsy, V. Braun, and U. Doetsch, "Link level performance assessment of reliability-based HARQ schemes in LTE," in *Proc. IEEE 79th Veh. Technol. Conf. (VTC Spring)*, Seoul, South Korea, May 2014, pp. 1–5.
[54] *5G; NR; Base Station (BS) Radio Transmission and Reception*, document 3GPP TS 38.104, Version 16.6.0, Release 16, 2021.

Yoel Bokobza received the B.Sc. and M.Sc. degrees in electrical and computer engineering from Ben-Gurion University of the Negev, Israel, in 2020 and 2022, respectively. Since 2019, he has been an Algorithm Researcher and a Developer at the Samsung Israel R&D Center (SIRC). His research interests include machine learning, signal processing, information theory, and communications.

Ron Dabora (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in electrical engineering from Tel-Aviv University in 1994 and 2000, respectively, and the Ph.D. degree in electrical engineering from Cornell University, USA, in 2007. From 1994 to 2000, he was with the Ministry of Defense of Israel and from 2000 to 2003, he was with the Algorithms Group, Millimetrix Broadband Networks, Israel. From 2007 to 2009, he was a Post-Doctoral Researcher with the Department of Electrical Engineering, Stanford University, USA. Since 2009, he has been with the Department of Electrical and Computer Engineering, Ben-Gurion University of the Negev, Israel, where he is currently an Associate Professor. Currently, he is also a Visiting Fellow with the Department of Electrical Engineering, Princeton University, USA. His research interests include network information theory, wireless communications, power line communications, and machine learning. He served as a TPC Member in a number of international conferences, including WCNC, PIMRC, and ICC. From 2012 to 2014, he served as an Associate Editor for the IEEE SIGNAL PROCESSING LETTERS and from 2014 to 2019, he served as a Senior Area Editor for the IEEE SIGNAL PROCESSING LETTERS.

Kobi Cohen (Senior Member, IEEE) received the B.Sc. and Ph.D. degrees in electrical engineering from Bar-Ilan University, Ramat Gan, Israel, in 2007 and 2013, respectively. He was with the Coordinated Science Laboratory, University of Illinois at Urbana–Champaign, from August 2014 to July 2015, and the Department of Electrical and Computer Engineering, University of California at Davis, Davis, from November 2012 to July 2014, as a Post-Doctoral Research Associate. In October 2015, he joined the School of Electrical and Computer Engineering, Ben-Gurion University of the Negev (BGU), Beer Sheva, Israel, where he is currently an Associate Professor. He is also a member of the Cyber Security Research Center and the Data Science Research Center, BGU. His main research interests include statistical inference and learning, signal processing, communication networks, decision theory and stochastic optimization with applications to large-scale systems, cyber systems, and wireless and wireline networks. Since 2021, he has been serving as an Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING and the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. Other selected awards and honors include highlighting in top 50 popular paper list, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS in 2019 and 2020 for paper: "Deep multi-user reinforcement learning for distributed dynamic spectrum access," highlighting in popular paper list, *IEEE Signal Processing Magazine* (2022) for paper: "Federated Learning: A signal processing perspective," receiving the Best Paper Award in the International Symposium on Modeling and Optimization in Mobile, Ad hoc and Wireless Networks (WiOpt) 2015, the Feder Family Award (second prize), awarded by the Advanced Communication Center at Tel Aviv University (2011), and President Fellowship (2008–2012) and top Honor List's Prizes (2006, 2010, and 2011) from Bar-Ilan University.