Deep Reinforcement Learning for Dynamic Multichannel Access in Wireless Networks

Shangxing Wang, Hanpeng Liu, Pedro Henrique Gomes and Bhaskar Krishnamachari
University of Southern California, Los Angeles, USA
Email: shangxiw@usc.edu, lhp13@mails.tsinghua.edu.cn, {pdasilva, bkrishna}@usc.edu

***Abstract*—We consider a dynamic multichannel access problem, where multiple correlated channels follow an unknown joint Markov model and users select the channel to transmit data. The objective is to find a policy that maximizes the expected long-term number of successful transmissions. The problem is formulated as a partially observable Markov decision process (POMDP) with unknown system dynamics. To overcome the challenges of unknown dynamics and prohibitive computation, we apply the concept of reinforcement learning and implement a Deep Q-Network (DQN). We first study the optimal policy for fixed-pattern channel switching with known system dynamics and show through simulations that DQN can achieve the same optimal performance without knowing the system statistics. We then compare the performance of DQN with a Myopic policy and a Whittle Index-based heuristic through both more general simulations as well as real data trace and show that DQN achieves near-optimal performance in more complex situations. Finally, we propose an adaptive DQN approach with the capability to adapt its learning in time-varying scenarios.**

***Index Terms*—Multichannel Access, Cognitive Sensing, POMDP, DQN, Reinforcement Learning, Online Learning**

I. Introduction

PRIOR work [2], [3] has shown that dynamic spectrum access is one of the keys to improving the spectrum utilization in wireless networks and meeting the need for more capacity. In the context of cognitive radio research, a standard assumption has been that secondary users may search and use idle channels that are not being used by their primary users (PU). While prior work has generally assumed a simple independent-channel (or PU activity) model, in practice external interference can cause the channels in wireless networks to be highly correlated, and the design of new algorithms and schemes in dynamic multichannel access is required to tackle this challenge.

We consider in this work a multichannel access problem with N correlated channels. Each channel has two possible states: *good* or *bad*, and their joint distribution follow a 2^N-states Markovian model. There is a single user (wireless node) that selects one channel at each time slot to transmit a packet. If the selected channel is in the *good* state, the transmission is successful; otherwise, there is a transmission failure. The goal is to obtain as many successful transmissions as possible over time. As the user is only able to sense his selected channel at each time slot, there is no full observation of the system available. In general, the problem can be formulated as a partially observable Markov decision process (POMDP), which is PSPACE-hard and the best known solution for finding the exact solution requires an exponential computation complexity [4]. Even worse, the parameters of the joint Markovian model might not be known *a-priori*.

We investigate the use of Deep Reinforcement Learning, in particular, Deep Q learning, from the field of machine learning as a way to enable learning in an unknown environment as well as overcome the prohibitive computational requirements. By integrating deep learning with Q learning, Deep Q learning or Deep Q Network (DQN) [5] can use a deep neural network with states as input and estimated Q values as output to efficiently learn policies for high-dimensional, large state-space problems. We implement a DQN that can find a channel access policy through online learning. This DQN approach is able to deal with large systems, and find a good or even optimal policy directly from historical observations without any requirement to

This work is an extended version of our ICNC paper [1].
This material is based upon work supported by Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001117C0053. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. This work is also supported in part by NSF through grant number 1423624.

know the system dynamics *a-priori*.

The rest of the paper is organized as follows. Section II shows the related work. Section III formulates the dynamic multichannel access problem when channels are potentially correlated. Section IV presents a Myopic and a Whittle Index-based heuristic to solve this problem. Section V presents the DQN framework. Section VI presents an optimal policy study on known fixed-pattern channel switching, and Section VII shows through simulations that DQN can achieve optimal performance. The evaluation results considering both synthetic and testbed-based datasets are shown in section VIII. Section IX introduces an adaptive DQN approach and, finally, Section X concludes our work.

II. Related Work

The dynamic multichannel access problem has been widely studied. But unlike many decision making problems, such as vertical handoff [6] and power allocation [7], that can be modeled as MDP, the dynamic multichannel problem is modeled as a POMDP, as channels are generally (two-state) Markov chains and a user has only partial observations. Finding an optimal channel access policy requires exponential time and space complexities. When channels are independent and identically distributed (i.i.d.), a Myopic policy has been shown to be optimal under certain conditions [8], [9]. But the Myopic policy does not have any performance guarantee when channels are correlated or follow different distributions.

When channels are independent but may follow different Markov chains, the dynamic multichannel access problem can be modeled as a Restless Multi-armed bandit problem (RMAB). A Whittle Index policy is introduced in [10] and shares the same simple semi-universal structure and optimality result as the Myopic policy. Numerical results are also provided showing that the Whittle Index policy can achieve near-optimal performance when channels are nonidentical. But the Whittle Index approach is not applicable when channels are correlated, which is the focus of our work.

In recent years, some works began to focus on the more practical and complex problem where both the system statistics is unknown and the channels are correlated. Q-learning is widely used as it is a model-free method that can learn the policy directly via online learning. The authors in [11] apply Q-learning to design channel sensing sequences, while in [12] it is shown that Q-learning can also take care of imperfect sensing. All these works assume the system state is fully observable and formulate the problem as an MDP, which significantly reduces the state space. On the contrary, our problem falls into the framework of POMDP because of the limit of the partial observation, and its large state space makes it impossible to maintain a simple look-up Q table to update Q values. New methods able to find approximations of Q-values are required to solve the large space challenge.

Reinforcement learning, including Q learning, has been integrated with advanced machine learning techniques to tackle difficult high-dimensional problems [13]–[15]. In 2013, Google DeepMind used a deep neural network, called DQN, to approximate Q values in Q learning that overcomes the limitation of the traditional look-up table approach, and provide an end-to-end approach to allow an agent to learn a policy from its observations. To the best of our knowledge, ours is the first study and implementation of DQN in the field of dynamic multi-channel access.

III. Problem Formulation

Consider a dynamic multichannel access problem where there is a single user dynamically choosing one out of N channels. At the beginning of each time slot, a user selects one channel to sense and transmit a packet. If the channel quality is good, the transmission succeeds and the user receives a positive reward $(+1)$, else the user transmission fails and there is a negative reward (-1). The objective is to design a policy that maximizes the expected long-term reward.

To model correlation across channels, the whole system is described as a 2^N-state Markov chain. Formally, let the state space of the Markov chain be $\mathcal{S} = \{\mathbf{s}_1, ..., \mathbf{s}_{2^N}\}$. Each state $\mathbf{s}_i$ $(i \in \{1, ..., 2^N\})$ is a length-N vector $[s_{i1}, ..., s_{iN}]$, where s_{ik} is the binary representation of the state of channel k: good (1) or bad (0). The transition of the Markov chain is denoted as $\mathbf{P}$. Since the user can only sense one channel at the beginning of each time slot, the full state of all channels is not observable. However, the user can infer a distribution over the system state according to his sensing decisions and observations.

Thus, the dynamic multichannel access problem falls into the general framework of POMDP. Let $\Omega(t) = [\omega_{\mathbf{s}_1}(t), ..., \omega_{\mathbf{s}_{2^N}}(t)]$ represent the belief vector maintained by the user, where $\omega_{\mathbf{s}_i}(t)$ is the conditional probability that the system is in state $\mathbf{s}_i$ given all previous decisions and observations. Given the sensing action $a(t) \in \{1, ..., N\}$ representing which channel to sense at time slot t, the user can observe the state of channel $a(t)$, denoted as $o(t) \in \{0, 1\}$. Then, based on this observation, the user can update the belief vector at time slot t, denoted as $\hat{\Omega}(t) = [\hat{\omega}_{\mathbf{s}_1}(t), ..., \hat{\omega}_{\mathbf{s}_{2^N}}(t)]$. The belief of each possible state $\hat{\omega}_{\mathbf{s}_i}(t)$ is updated as follows:

$$\hat{\omega}_{\mathbf{s}_i}(t) = \begin{cases} \frac{\omega_{\mathbf{s}_i}(t)\mathbb{1}(\mathbf{s}_{ik}(t)=1)}{\sum_{i=1}^{2^N} \omega_{\mathbf{s}_i}(t)\mathbb{1}(\mathbf{s}_{ik}(t)=1)} & a(t) = k, o(t) = 1 \\ \frac{\omega_{\mathbf{s}_i}(t)\mathbb{1}(\mathbf{s}_{ik}(t)=0)}{\sum_{i=1}^{2^N} \omega_{\mathbf{s}_i}(t)\mathbb{1}(\mathbf{s}_{ik}(t)=0)} & a(t) = k, o(t) = 0 \end{cases}$$

where $\mathbb{1}(.)$ is the indicator function.

Combining the newly updated belief vector $\hat{\Omega}(t)$ for time slot t with the system transition matrix $\mathbf{P}$, the belief vector for time slot $t + 1$ can be updated as $\Omega(t + 1) = \hat{\Omega}(t)\mathbf{P}$.

A sensing policy $\pi : \Omega(t) \to a(t)$ is a function that maps the belief vector $\Omega(t)$ to a sensing action $a(t)$ at each time slot t. Given a policy π, the long-term reward considered in this paper is the expected accumulated discounted reward over infinite time horizon, defined as $\mathbb{E}_\pi[\sum_{t=1}^{\infty} \gamma^{t-1} R_{\pi(\Omega(t))}(t)|\Omega(1)]$, where $0 \leq \gamma < 1$ is a discounted factor, $\pi(\Omega(t))$ is the channel sensing action at time t given belief vector $\Omega(t)$, and $R_{\pi(\Omega(t))}(t)$ is the corresponding reward. Our objective is to find a sensing policy π^* that maximizes the expected accumulated discounted reward over infinite time

$$\pi^* = \arg\max_{\pi} \mathbb{E}_\pi[\sum_{t=1}^{\infty} \gamma^{t-1} R_{\pi(\Omega(t))}(t)|\Omega(1)]$$

As the dynamic multichannel access problem is a POMDP, the optimal sensing policy π^* can be found by considering its belief space and solving an augmented MDP instead, for example, via value iteration, however the dimension of the belief vector is exponentially large in the number of channels. Even worse, the infinite size of the continuous belief space and the impact of the current action on the future reward makes POMDP PSPACE-hard, which is even less likely to be solved in polynomial time than NP-hard problems [4]. To exemplify the time complexity of solving such POMDP problem, we simulate the multichannel access problem with known system dynamics and use a POMDP solver called SolvePOMDP [16] to find its optimal solution. In Figure 1, we show the run-time as we increase the number of channels in the system. When the number of channels is higher than 5, we find that the POMDP solver can not converge after a long interval, and it gets terminated when the run-time exceeds the time limit. All these factors make it impossible to find the optimal solution to the problem in general, and many existing works [8]–[10], [17]–[21] attempt to address this challenge of prohibitive computation by considering either simpler models or approximation algorithms.

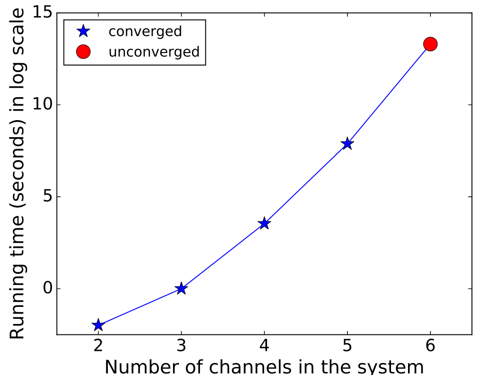


Fig. 1: Running time (seconds) in log scale of the POMDP solver as we vary the number of channels in the system

IV. Myopic Policy and Whittle Index

In the domain of dynamic multichannel access, there are many existing works on finding the optimal/near-optimal policy with low computation cost when the channels are independent and system statistics ($\mathbf{P}$) is known. The Myopic policy and the Whittle Index policy are two effective and easy-to-implement approaches for this setting.

A. *Myopic Policy*

A Myopic policy only focuses on the immediate reward obtained from an action and ignores its effects in the future. Thus the user always tries to select a channel which gives the maximized expected immediate reward.

The Myopic policy is not optimal in general. Researchers in [8], [9] have studied its optimality when N channels are independent and statistically identical Gilbert-Elliot channels that follow the same 2-state Markov chain with the transition matrix as $\begin{bmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{bmatrix}$. It is shown that the Myopic policy is optimal for any number of channels when the channel state transitions are positively correlated, i.e., $p_{11} \geq p_{01}$. The same optimal result still holds for two or three channels when channel state transitions are negatively correlated, i.e., $p_{11} < p_{01}$. In addition, the Myopic policy has a simple robust structure that follows a round-robin channel selection procedure.

B. Whittle Index Based Heuristic Policy

When channels are independent, the dynamic multichannel access problem can also be considered as a restless multi-armed bandit problem (RMAB) if each channel is treated as an arm. An index policy assigns a value to each arm based on its current state and chooses the arm with the highest index at each time slot. The index policy is not guaranteed to be optimal in general.

In [10], the Whittle Index is obtained in closed-form for the case when $\mathbf{P}$ is known and all channels are independent but may follow different 2-state Markov chain models. In the special case when all channels are identical, it is shown to coincide with the above-described Myopic policy.

When channels are correlated, the Whittle Index cannot be defined and thus the Whittle Index policy cannot be directly applied to our problem. Nevertheless, as a baseline in our evaluations, to leverage its simplicity, we propose an heuristic that ignores the correlations among channels and uses the joint transition matrix $\mathbf{P}$ and Bayes' Rule to compute the 2-state Markov chain for each individual channel. Once each channel model is found, we can apply the Whittle Index policy accordingly.

The Myopic policy and the Whittle Index policy are easy to implement in practice and have polynomial run-time, however they achieve optimality only under certain conditions when channels are independent. Moreover, both policies require prior knowledge of the system dynamics, which is hard to obtain beforehand. However, to the best of our knowledge, there is no easy-to-implement policy applicable to the general case where channels are correlated and the system dynamics are unknown — thus a new approach is needed.

V. Deep Reinforcement Learning Framework

When channels are correlated and system dynamics are unknown, there are two main approaches to tackle the dynamic multichannel access problem: (i) Model-based approach: first estimate the system model from observations and then apply dynamic programming or a computationally efficient heuristic policy such as Myopic/Whittle Index policies; (ii) Model-free approach: learn the policy directly through interactions with the system without estimating the system model. The model-based approach is less favored since the user's limited observation capability may result in a bad system model estimation. Even worse, even if the system dynamics is well estimated, solving a POMDP in a large state space is always a bottleneck as the dynamic programming method has exponential time complexity (as explained in Section III) and the heuristic approaches do not have any performance guarantee. All these challenges motivate us to follow the model-free approach, which, by incorporating the idea of Reinforcement Learning, can learn directly from observations without the necessity of finding an estimated system model and can be easily extended to very large and complicated systems.

A. Q-Learning

We focus on the reinforcement Learning paradigm, Q-learning [22] specifically, to incorporate learning for the dynamic multichannel access problem. The goal of Q-learning is to find an optimal policy, i.e., a sequence of actions that maximizes the long-term expected accumulated discounted reward. Q-learning is an empirical value iteration approach and the essence is to find the Q-value of each state and action pairs, where the state $\mathbf{x}$ is a function of observations (and rewards) and the action a is some action that a user can take given the state $\mathbf{x}$. The Q-value of a state-action pair $(\mathbf{x}, a)$ from policy π, denoted as $Q^{\pi}(\mathbf{x}, a)$, is defined as the sum of the discounted reward received when taking action a in the initial state $\mathbf{x}$ and then following the policy π thereafter. $Q^{\pi^*}(\mathbf{x}, a)$ is the Q-value with initial state $\mathbf{x}$ and initial action a, and then following the optimal policy π^*. Thus, the optimal policy π^* can be derived as $\pi^*(\mathbf{x}) = \arg\max_a Q^{\pi^*}(\mathbf{x}, a), \forall \mathbf{x}$.

One can use online learning method to find $Q^{\pi^*}(\mathbf{x}, a)$ without any knowledge of the system dynamics. Assume at the beginning of each time slot, the agent takes an action $a_t \in \{1, ..., N\}$ that maximizes its Q-value of state-action pair $(\mathbf{x}_t, a_t)$ given the state is $\mathbf{x}_t$, and gains a reward r_{t+1}. Then the online update rule of Q-values with learning rate $0 < \alpha < 1$ is given as follows:

$$Q(\mathbf{x}_t, a_t) \leftarrow Q(\mathbf{x}_t, a_t) + \alpha[r_{t+1} + \gamma \max_{a_{t+1}} Q(\mathbf{x}_{t+1}, a_{t+1}) - Q(\mathbf{x}_t, a_t)]$$

In the context of the dynamic multichannel access, the problem can be converted to an MDP when considering the belief space, and Q-learning can be applied consequently. However, this approach is impractical since the belief update is maintained by knowing the system transition matrix $\mathbf{P}$ *a-priori*, which is hardly available in practice. Instead, we apply Q-learning by directly considering the history of observations and actions. We define the state for the Q-learning at time slot t as a combination of historical selected channels as well as their observed channel conditions over previous M time slots, i.e., $\mathbf{x}_t = [a_{t-1}, o_{t-1}, ..., a_{t-M}, o_{t-M}]$. And intuitively, the more historical information we consider (i.e., the larger M is), the better Q-learning can learn.

B. Deep Q-Network

Q-learning works well when the problem's state space is small, as a look-up table can be used to update Q values. But this is impossible when the state space becomes large. The state space size in this work grows exponentially as $O(N^M)$, as we use a combination of M vectors of length N to represent historical observations and actions for a system with N 2-state channels over past M time slots. M is required to be large so that Q-learning can capture enough information for learning. Even worse, since many states are rarely visited, their corresponding Q-values are seldom updated. This causes Q learning to take a very long time to converge.

Motivated by its success in other domains [5], we adopt the deep Q-Network approach to address the very large state space. DQN takes the state-action pair as input and outputs the corresponding Q-value. Q-network updates its weights θ at each iteration i to minimize the loss function $L_i(\theta_i) = \mathbb{E}[(y_i - Q(\mathbf{x}, a; \theta_i))^2]$, where $y_i = \mathbb{E}[r + \gamma \max_{a'} Q(\mathbf{x}', a'; \theta_{i-1})]$ is derived from the same Q-network with old weights θ_{i-1} and new state $\mathbf{x}'$ after taking action a from state $\mathbf{x}$.

VI. Optimal Policy for Known Fixed-Pattern Channel Switching

To study the performance of DQN, we first consider a correlated channel model that we refer to as fixed-pattern channel switching, in which all the N channels in the system can be divided into several independent subsets and these subsets take turns to be activated following a fixed pattern. Specifically, we assume all channels in one currently activated subset are good and all channels in inactivated subsets are bad. At each time slot, with a known probability p $(0 \le p \le 1)$ the next following subset is activated, and with probability $1 - p$ the current subset remains activated. We assume the activation order of the subsets is known, fixed, and will not change over time. In this special case, the optimal policy can be found analytically and is summarized in Theorem 1, providing a baseline to evaluate the performance of DQN implementation in the next section.

Theorem 1: When the system follows a fixed-pattern channel switching model, the optimal channel access policy follows Algorithm 1 depending on the value of p.

Algorithm 1 Optimal policy

```
 1: At the beginning of time slot 0, choose a channel in the
    initial activated subset $C_1$
 2: for $n = 1, 2, \ldots$ do
 3:     At the beginning of time slot $n$,
 4:     if $0.5 \le p \le 1$ then
 5:         if The previous chosen channel is good then
 6:             Choose a channel in the next activated subset
                according to the subset activation order
 7:         else
 8:             Stay in the same channel
 9:     else
10:         if The previous chosen channel is good then
11:             Stay in the same channel
12:         else
13:             Choose a channel in the next activated subset
                according to the subset activation order
```

Proof: Please see proof in [23]. ■

It turns out that the optimal policy for the fixed-pattern channel switching shares a structure that is simple and robust similar to the Myopic policy in [8]: the optimal policy has a round-robin structure (in terms of the channel subset activation order) and does not require to know the exact value of p except whether it is above/below 0.5. This semi-universal property makes the optimal policy easy to implement in practice and robust to mismatches of system dynamics.

TABLE I: List of DQN Hyperparameters

Hyperparameters	Values
ϵ	0.1
Minibatch size	32
Optimizer	Adam
Activation Function	ReLU
Learning rate	10^{-4}
Experience replay size	$1,000,000$
γ	0.9

VII. Experiment and Evaluation of Learning for Unknown Fixed-Pattern Channel Switching

We present details of our DQN implementation and then evaluate its performance on the fixed-pattern switching pattern model, comparing it to the optimal policy, through experiments.

A. DQN Architecture

We design a DQN by following the *Deep Q-learning with Experience Replay Algorithm* [5] and implement it in TensorFlow [24]. The structure of our DQN is finalized as a fully connected neural network with each of the two hidden layers containing 200 neurons[1]. The activation function of each neuron is Rectified Linear Unit (*ReLU*), which computes the function $f(x) = \max(x, 0)$. The input of the DQN is defined as the combination of previous actions and observations over previous M time slots, selecting $M = N$. The output of the DQN is a vector of length N, where the ith item represents the Q value of a given state if channel i is selected. We apply the ϵ-greedy policy with the random action exploration probability ϵ fixed as 0.1. A technique called *Experience Replay* introduced in [5] is used to store previous observation data and break correlations among data samples that makes training stable and convergent. When updating the weights θ of the DQN, a minibatch of 32 samples are randomly selected from the replay memory to compute the loss function, and a recently proposed Adam algorithm [25] is used to update the weights (details on the hyperparameters are listed in Table I). In the following experiment settings, we consider a system of 16 channels.

B. Experiment and Evaluation

In our experiments, we considered different situations: single good channel or multiple good channels, sequential switching or arbitrary switching,

[1]Please refer to [23] for more explanations on this structure.

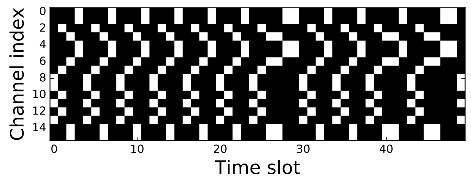


Fig. 2: A capture of a multiple good channels situation

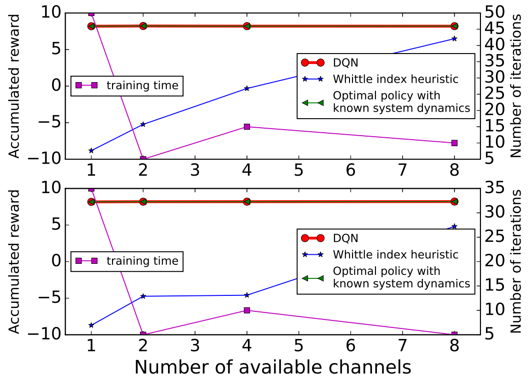


Fig. 3: Average discounted reward as we increase the number of good channels in the multiple good channels situation

and observe that DQN can achieve the optimal performance in all situations. Due to the space constraint, we only present results on the multiple good channels situation, and refer the reader to [23] for more results on other situations.

In this section, we investigate the fixed-pattern channel switching model with 16 channels are evenly divided into several subsets that take turns to become available with a switching probability fixed at $p = 0.9$. In Fig. 2, we provide a pixel illustration to visualize how the states of channels change in a multiple good channels situation over 50 time slots, where a white cell indicates the corresponding channel is good at the corresponding time.

We compare the DQN with two other policies: the Whittle Index heuristic policy and the optimal policy with known system dynamics from section VI. The optimal policy has full knowledge of the system dynamics and serves as a performance upper bound. In the Whittle Index heuristic, the user assumes all channels are independent and observes each channel individually for $10,000$ time slots to estimate its 2-state Markov chain transition matrix.

We vary the number of channels in a subset as 1, 2, 4 and 8 in the experiment, and present the experimental results in Fig. 3. The 16 channels in the system are in order and the subsets are activated in a sequential round-robin order in the upper graph in Fig. 3, while the channels are arranged arbitrarily and the activation order of subsets is also arbitrary in the bottom graph in Fig. 3. As can be seen, DQN remains robust and achieves the same optimal performance in all cases as the optimal policy and performs significantly better than the Whittle Index heuristic. This shows that DQN can implicitly learn the system dynamics including the correlation among channels, and finds the optimal policy accordingly. On the contrary, the Whittle Index heuristic simply assumes the channels are independent and is not able to find or make use of the correlation among channels. Moreover, the training time decreases as the number of good channels increases. This is because there is more chance to find a good channel when more good channels are available at a time slot, and the learning process becomes easier so that the DQN agent can take less time exploring and is able to find the optimal policy more quickly. This also explains why Whittle Index heuristic performs better when there are more good channels available.

VIII. Experiment and Evaluation of DQN for More Complex Situations

We now consider whether DQN can achieve a good or even optimal performance in more complex and realistic situations. For this set of experiments, we re-tuned our neural network structure to become a fully connected neural network with each hidden layer containing 50 neurons (and the learning rate is set as 10^{-5})[2], and considered more complex simulated situations as well as real data traces described below.

A. *Perfectly correlated scenario*

We consider a highly correlated scenario. In a 16-channel system, we assume only two or three channels are independent, and other channels are exactly identical or opposite to one of these independent channels. In addition to the Whittle Index heuristic, we also compare DQN with a Random Policy in which the user randomly selects one channel with equal probability at each time slot. Since the optimal policy is computationally prohibitive, we implement the Myopic policy instead as a genie (knowing the system statistics *a-priori*) since it is simple, robust and can achieve an optimal performance in certain situations.

[2]Please refer to [23] for the explanations on the DQN structure.

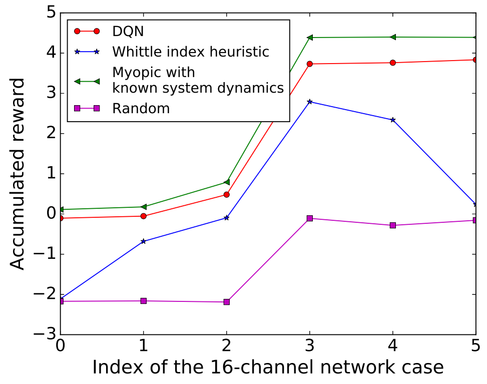


Fig. 4: Average discounted reward for 6 different cases. Each case considers a different set of correlated channels

During the simulation, we arbitrarily set the independent channels to follow the same 2-state Markov chain with $p_{11} \geq p_{01}$. When the correlation coefficient $\rho = 1$, the user can ignore those channels that are perfectly correlated with independent channels and only select a channel from the independent channels. In this case, the multichannel access problem becomes selecting one channel from several i.i.d. channels that are positively correlated, i.e., $p_{11} \geq p_{01}$. Then as it is shown in the previous work [8], [9], the Myopic policy with known $\mathbf{P}$ is optimal and has a simple round-robin structure alternating among independent channels. In the case when $\rho = -1$, the Myopic policy with known $\mathbf{P}$ also has a simple structure that alternates between two negatively perfectly correlated channels. Though more analysis needs to be done in future to show whether the Myopic policy is optimal/near-optimal when $\rho = -1$, it can still serve as a performance benchmark as the Myopic policy is obtained with full knowledge of the system dynamics.

In Fig. 4 we present the performance of all four policies: (i) DQN, (ii) Random, (iii) Whittle Index heuristic, and (iv) Myopic policy with known $\mathbf{P}$. In the first three cases (x-axis 0, 1 and 2), the correlation coefficient ρ is fixed as 1 and in the last three cases (x-axis 3, 4 and 5), ρ is fixed as -1. We also vary the set of correlated channels to make cases different. The Myopic policy in the first three cases is optimal, and in the last three cases is conjectured to be near-optimal. As shown in Fig. 4, the Myopic policy, which is implemented based on the full knowledge of the system, is the best among all six cases and serves as an upper bound. DQN provides a performance very close to the Myopic policy without any knowledge of the system dynamics. The Whittle Index policy performs worse than DQN in all cases.

B. Real data trace

We use real data trace collected from our indoor testbed Tutornet[3] to train and evaluate the performance of DQN on real systems. The testbed is composed of TelosB nodes with IEEE 802.15.4 radio. We programmed a pair of motes distanced approximately 20 meters to be transmitter/receiver. The transmitter continually transmits one packet rapidly to each one of the 16 available channels within one time slot and the synchronized receiver records the successful and failed attempts, with the only interference coming from surrounding WiFi networks that show high dynamic variability.

The data are collected for about 17 hours. In order to create a challenging environment to test the learning capability of DQN, we ignore 8 good channels and use only the data trace from the remaining 8 channels that show significant WiFi interference.

We use the same data trace to train the DQN and to compute the MLE of the transition matrices of each channel for the Whittle index based heuristic policy. We compare the performance of the DQN policy, the Whittle index based heuristic policy and the Random policy. The Myopic Policy is not considered as finding the transmission matrix of the entire system is computationally expensive. The average accumulated discounted reward from each policy is listed in descending order: 0.947 (DQN), 0.767 (Whittle Index) and -2.170 (Random Policy). It can be seen that DQN performs best in this complicated real scenario. We also present the channel utilization of each policy in Fig. 5 to illustrate the difference among them. It shows DQN benefits from using other channels when the two best channels (used by the Whittle Index heuristic all the time) may not be in good states.

C. Practical Issues

We now consider the practical issue of synchronization between the sender and receiver of the system. One approach is to run the same structured DQNs at the sender and the receiver separately. The two DQNs start with the same default channel and are trained concurrently. By allowing the receiver to send back an ACK/NAK to the sender indicating whether it receives a message or not every time slot, the sender and receiver are guaranteed to have the same channel observations and thus training samples. By using the same random seed on both sides to initialize the pseudorandom number generator, we can avoid any difference occurred during exploration and back propagation during training. Thus, the two DQNs always have the same parameters and select the same channels, and the final learned policy is guaranteed to be the same.

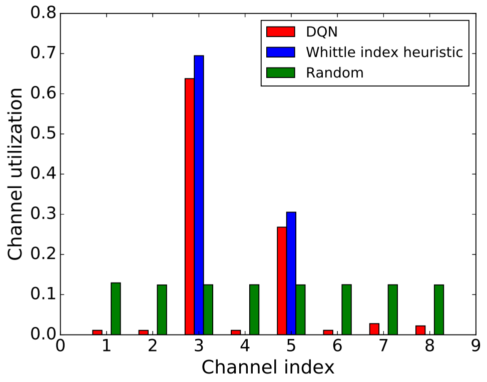


Fig. 5: Channel utilization of 8 channels in the testbed

The channel mismatch problem can still happen when an ACK/NAK is lost, which not only causes loss of communication, but also results in different learned DQN models at the sender and receiver that give different channel selection policies. One possible approach is to find a way to let the sender and the receiver be aware of the time when a channel mismatch happens, and try to recover in time. The sender can detect the mismatch event if no ACK/NAK is received. Once the mismatch happens, the sender stops updating its DQN model as well as training dataset and transmits data in the future using one single good channel - or a small set of channels known so far to have better channel conditions [26]. Along with the data messages, the sender also sends the timestamp when the channel mismatch was perceived. The sender keeps sending this channel mismatch time information until an ACK being received, which indicates the receiver is on the same channel again and receives the channel mismatch information. The receiver can then set its DQN model as well as its training dataset back to the state right before the channel mismatch happened, which guarantees that the sender and the receiver have the same DQN models as well

[3]More information about the testbed on http://anrg.usc.edu/www/tutornet/

as training datasets. They can resume operating and training thereafter. It can be shown that the expected number of time slots needed for re-syncing on a good channel after a channel mismatch is $\frac{N}{\epsilon p_{good}(1-p_{ack})}$, where N is the number of channels, p_{good} the probability this channel being good, p_{ack} the probability an ACK/NAK being lost, and ϵ the exploration probability. As long as the sender and the receiver can re-synchronize again after a channel mismatch, the effectiveness and performance of the DQN approach is guaranteed.

IX. Adaptive DQN for Unknown, Time-Varying Environments

Algorithm 2 Adaptive DQN

```
1: First train DQN to find a good policy to operate with
2: for $n = 1, 2, \ldots$ do
3:     At the beginning of period $n$
4:     Evaluate the accumulated reward of the current policy
5:     if The reward is reduced by a given threshold[4] then
6:         Re-train the DQN to find a new good policy
7:     else
8:         Keep using the current policy
```

To enhance DQN and make it more applicable in realistic, dynamic situations, we have designed an adaptive extension in Algorithm 2 to make DQN able to be aware of the system change and re-learn if needed. The main idea is to let DQN periodically evaluate the performance (i.e., the accumulated reward) of its current policy, and if the performance degrades by a certain amount, the DQN can infer that the environment has changed and start re-learning.

To evaluate this enhancement, we make the system initially follow one of the fixed-pattern channel switching cases from Section VII, and after some time it changes to another case. We consider both single good channel and multiple good channels situations. We let DQN automatically operate according to Alg. 2, while we manually re-train Whittle Index heuristic (as it is not able to detect any change) when there is a change in the environment. Fig. 6 compares the reward of both the old and new policies learned for DQN and the Whittle Index heuristic in the new environment, as we vary the pattern changes. As can be seen, DQN is able to find an optimal policy for the new environment as the genie optimal policy does, while Whittle Index heuristic, even manually tuned, does not.

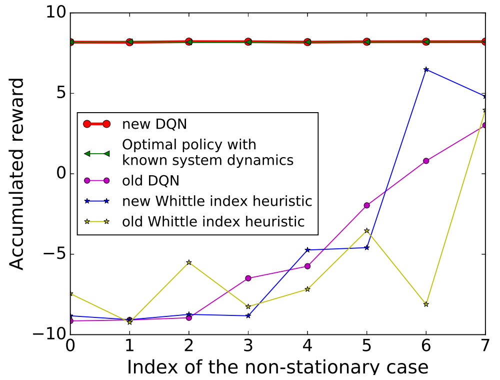


Fig. 6: Average discounted reward as we vary the channel switch

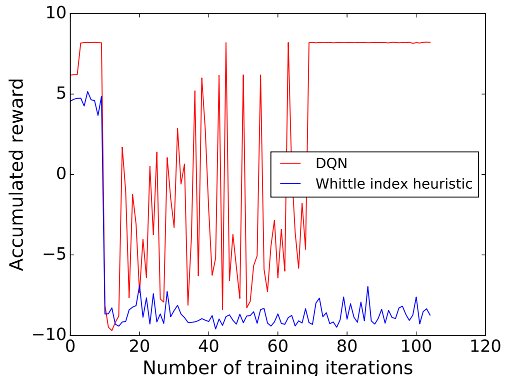


Fig. 7: Average discounted reward in real time during training in unknown fixed-pattern channel switching

We also provide the real-time accumulated reward during the learning process of DQN and the Whittle Index heuristic in one of the above pattern changing situations in Fig. 7. The system initially starts with an environment that has 8 channels being good at each time slot for the first 10 iterations. As can be seen, both DQN and the Whittle Index heuristic are able to quickly find a good channel access policy, but DQN achieves the optimal performance. At iteration 11, the environment changes to having only 1 channel being good at each time slot. As there is a significant drop in the reward, DQN can detect the change and starts re-learning. And at iteration 70, DQN finds the optimal policy and our system keeps following the optimal policy thereafter. On the other hand, even though we manually enable the Whittle Index heuristic to detect the change and re-estimate the system model and re-find a new policy, its performance is still unsatisfying as it cannot make use of the correlation among channels.

X. Conclusion and Future Work

In this paper, we have considered the dynamic multichannel access problem in a more general and practical scenario when channels are correlated and system statistics is unknown. The problem is an unknown POMDP without any tractable solution, and we have applied an end-to-end DQN approach that directly utilizes historical observations and actions to find the access policy via online learning. In the fixed-pattern channel switching, we have analytically found the optimal access policy with known system statistics and full observation ability. Through simulations, we have shown DQN is able to achieve the same optimal performance even without any prior knowledge. We have also shown from both simulations and real data trace that DQN can achieve near-optimal performance in more complex scenarios. In addition, we have designed an adaptive DQN and shown through numerical simulations that it can detect system changes and re-learn in non-stationary dynamic environments.

There are a number of open directions suggested by the present work. First, we plan to apply the DQN framework to consider more realistic and complicated scenarios such as multi-user, multi-hop and simultaneous transmissions in WSNs. The framework of DQN can be directly extended to consider these practical factors in a simple way. For example, in the situation of multiple users, to avoid interference and collisions among users, we can adopt a centralized approach: assuming there is a centralized controller that can select a subset of non-interfering channels at any time slot, and assign one to each user to avoid a collision. By redefining the action as selecting a subset of non-interfering channels, the DQN framework can be directly used for this multi-user scenario. As the action space becomes large when selecting multiple channels, the current DQN structure requires careful re-design and may require very long training interval before finding a reasonable solution. Instead, we use the same DQN structure as that in Section VII and consider the multiple-users situation in a smaller system that contains 8 channels where at any time slot 6 channels become good and channel conditions change in a round-robin pattern. The number of users varies from 2 to 4. As is shown in Fig. 8, DQN can still achieve a good performance in the multiple-user case. Other deep reinforcement learning approaches, such as Deep Deterministic Policy Gradient (DDPG) [27], will be studied in future to tackle the large action space challenge.

Second, when the number of users in the network

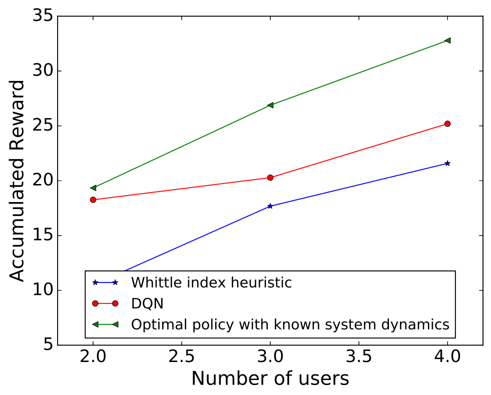


Fig. 8: Average discounted reward as we vary the number of users in the multiple-user situation

becomes large, the above proposed centralized approach becomes too computationally expensive. In future, we plan to study a more practical distributed approach where each user can learn a channel selection policy independently. One intuitive idea is to implement a DQN at each user independently. Then users can learn their channel selection policies parallelly, and avoid interference and conflicts by making proper channel-selection decisions based on the information gained from observations and rewards. However, whether a good or optimal policy can be learned, and whether an equilibrium exists are unknown and need further investigation.

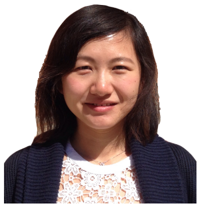
Moreover, as DQN is not easy to tune and may get stuck in local optima easily, we plan to work on improving our DQN implementation as well as considering other Deep Reinforcement Learning approaches to see if they have the ability to reach the optimal performance in general situations and study the tradeoff between implementation complexity and performance guarantee. Also as a way to test the full potential of DQN (or Adaptive DQN) as well as other deep reinforcement learning technologies in the problem of multichannel access, we encourage the networking community to work together to create an open source dataset that contains different practical channel access scenarios so that researchers can benchmark the performance of different approaches. We have published all the channel access environments and real data trace considered in this work[5]. This might serve as an useful benchmark dataset for researchers to use.

References

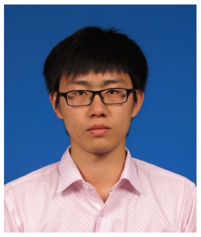
[1] S. Wang, H. Liu, P. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access in

[5]https://github.com/ANRGUSC/MultichannelDQN-channelModel

wireless networks," in *ICNC*, 2017.
[2] R. Knopp and P. Humblet, "Information capacity and power control in single-cell multiuser communications," in *IEEE ICC*, 1995.
[3] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, "Next generation/dynamic spectrum access/cognitive radio wireless networks: A survey," *Computer networks*, vol. 50, no. 13, pp. 2127–2159, 2006.
[4] C. Papadimitriou and J. N. Tsitsiklis, "The complexity of markov decision processes," *Math. Oper. Res.*, vol. 12, no. 3, pp. 441–450, 1987.
[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.
[6] E. Stevens-Navarro, Y. Lin, and V. W. S. Wong, "An mdp-based vertical handoff decision algorithm for heterogeneous wireless networks," *IEEE Trans on Vehicular Technology*, vol. 57, no. 2, pp. 1243–1254, March 2008.
[7] P. Sakulkar and B. Krishnamachari, "Online learning of power allocation policies in energy harvesting communications," in *SPCOM*, 2016.
[8] Q. Zhao, B. Krishnamachari, and K. Liu, "On myopic sensing for multi-channel opportunistic access: structure, optimality, and performance," *IEEE Trans. Wireless Commun.*, vol. 7, no. 12, pp. 5431–5440, dec 2008.
[9] 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, 2009.
[10] 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.
[11] P. Venkatraman, B. Hamdaoui, and M. Guizani, "Opportunistic bandwidth sharing through reinforcement learning," *IEEE Trans. on Vehicular Technology*, vol. 59, no. 6, pp. 3148–3153, July 2010.
[12] Y. Zhang, Q. Zhang, B. Cao, and P. Chen, "Model free dynamic sensing order selection for imperfect sensing multichannel cognitive radio networks: A q-learning approach," in *IEEE ICC*, 2014.
[13] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *arXiv preprint arXiv:1504.00702*, 2015.
[14] J.-A. M. Assael, N. Wahlström, T. B. Schön, and M. P. Deisenroth, "Data-efficient learning of feedback policies from image pixels using deep dynamical models," *arXiv preprint arXiv:1510.02173*, 2015.
[15] J. Ba, V. Mnih, and K. Kavukcuoglu, "Multiple object recognition with visual attention," *arXiv preprint arXiv:1412.7755*, 2014.
[16] "Solvepomdp," http://erwinwalraven.nl/solvepomdp/.
[17] H. Liu, K. Liu, and Q. Zhao, "Logarithmic weak regret of non-bayesian restless multi-armed bandit," in *IEEE ICASSP*, 2011.
[18] C. Tekin and M. Liu, "Online learning in opportunistic spectrum access: A restless bandit approach," in *IEEE INFOCOM*, 2011.
[19] W. Dai, Y. Gai, and B. Krishnamachari, "Efficient online learning for opportunistic spectrum access," in *IEEE INFOCOM*, 2012.
[20] ——, "Online learning for multi-channel opportunistic access over unknown markovian channels," in *IEEE SECON*, 2014.
[21] R. Ortner, P. A. D. Ryabko, and R. Munos, "Regret bounds for restless markov bandits," in *ALT*, 2012.
[22] C. J. C. H. Watkins and P. Dayan, "Q-learning," in *Machine Learning*, 1992, pp. 279–292.
[23] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access in wireless networks," *Technical Report arXiv:1802.06958*, 2018.
[24] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in *OSFI*, 2016, pp. 265–283.
[25] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[26] J.-Y. Kim, S.-H. Chung, and Y.-V. Ha, "A fast joining scheme based on channel quality for IEEE802.15.4e TSCH in severe interference environment," in *ICUFN*. IEEE, jul 2017.
[27] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning." *CoRR*, 2015.



Shangxing Wang is a Ph.D. candidate at Ming Hsieh Department of Electrical Engineering, University of Southern California. Her research focuses on the design and analysis of novel algorithms and applications in the field of networking, including Robotic Wireless Networks and Internet of Things (IoT). She received the M.S. degree in computer science from University of Southern California in 2017 and the B.E. degree in telecommunication engineering from Xidian University in 2010.



Hanpeng Liu is a Ph.D. student in Computer Science at University of Southern California. He received his B.E. in Computer Science from Tsinghua University in 2017. His research interests lie in machine learning, deep learning and tensor analysis.

Pedro Henrique Gomes is currently an experienced researcher at Ericsson Research, Brazil. He is also a Ph.D. candidate at Ming Hsieh Department of Electrical Engineering, University of Southern California. His research is focused on developing algorithms to optimize reliability, delay and energy efficiency of networks based on Timeslotted Channel Hopping (TSCH) protocol. He received his MSc in Computer Science (2011) and BSc in Computer Engineering (2007) from the University of Campinas, Brazil.

Bhaskar Krishnamachari received the B.E. degree in electrical engineering from Cooper Union, New York, NY, USA, in 1998 and the M.S. and Ph.D. degrees from Cornell University in 1999 and 2002, respectively. He is a Professor with the Department of Electrical Engineering, Viterbi School of Engineering, University of Southern California. His primary research interest is in the design, theoretical analysis, and experimental evaluation of algorithms and protocols for next-generation wireless networks, including low power wireless sensor networks. He has co-authored over 200 publications on this topic, including best paper awards at Mobicom in 2010, IPSN in 2004 and 2010, and MSWiM in 2006, that have been collectively cited over 20000 times per Google Scholar.