IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON COMPLEX NETWORKS ANALYSIS AND ENGINEERING IN 5G AND BEYOND TOWARDS 6G

Received March 23, 2020, accepted May 12, 2020, date of publication May 18, 2020, date of current version June 2, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.2995456*

# Multi-Agent Deep Learning for Multi-Channel Access in Slotted Wireless Networks

**RUBEN MENNES[1], (Student Member, IEEE), FELIPE A. P. DE FIGUEIREDO[2], (Student Member, IEEE), AND STEVEN LATRÉ[1], (Member, IEEE)**

[1]Department of Computer Science, University of Antwerp - imec, 2000 Antwerp, Belgium
[2]Department of Information Technology, Ghent University - imec, 9000 Ghent, Belgium

Corresponding author: Ruben Mennes (ruben.mennes@uantwerpen.be)

This work was supported by the Research Foundation Flanders (FWO) under Grant 1S52917N.

**ABSTRACT** As the number of devices connected to the internet and the amount of data they generate increases, the wireless spectrum is becoming an essential and scarce resource. Most connected devices use wireless technologies that use the industrial, scientific, and medical (ISM) radio bands. As a result, different technologies are interfering with each other. Today's existing collision avoidance techniques either apply a random back-off when a signal collision is detected or assume that knowledge about other nodes' spectrum occupation is known. These approaches are competent approaches to optimise inter-network spectrum usage, but fail to optimise overall channel capacity and throughput of all neighbouring wireless networks. In this paper, we present a Deep Neural Network (DNN) approach that can predict spectrum occupation of unknown neighbouring networks in the near future by using online supervised learning in a multi-agent setting. This prediction can be employed by existing network schedulers to avoid collisions with surrounding networks or other electromagnetic sources. The DNN is trained in an online way, as the problem is a partially observable stochastic game with continuous action space. Our findings show a reduction in the number of collisions between the own network and neighbouring networks of 30%, and an increase in overall throughput of 10% in a medium-sized network with an unknown set of neighbouring networks.

**INDEX TERMS** Collaborative wireless networks, deep learning, machine learning, wireless MAC.

## I. INTRODUCTION

Dynamic spectrum access is one of the keys to improving spectrum utilisation in wireless networks, as to meet the increasing need for capacity [1]. The phenomenon of spectrum scarcity is getting increasingly serious with the growing use of the unlicensed ISM radio bands. On licensed bands, on the other hand, governmental agencies regulate the spectrum and only a limited number of users can use a specific set of frequencies. Recently, some approaches try to make underutilised licensed spectrum available (e.g. Cognitive Radios (CRs)), or move partially to unlicensed bands when the licensed bands do not suffice (e.g. LTE-U). The number of wireless devices is expected to grow from 8 billion in 2016 to 28 billion in 2023, along with a projected increase in transmitted data from 7.2 exabytes per month in 2016 to 49.0 exabytes per month in 2021. Half of this data will be exchanged via Wi-Fi on the ISM band [2].

Optimising the spectrum usage in a wireless network is a well-studied area, with a lot of existing work focusing on Medium Access Control (MAC) algorithm design. Most of these algorithms, however, only try to optimise their own network performance, as information about the others is unavailable. At the same time, the number of different technologies that use the same ISM bands is also growing and, with dynamic spectrum access, these MAC algorithms should take cross-technology interference into account. Current collision avoidance techniques in MAC algorithms could be divided into three different classes [3]. (i) Random Access protocols: Nodes can access the spectrum at any time to transmit packets. Most of the time, these protocols use a Carrier Sensing Multiple Access (CSMA) collision avoiding technique to reduce collisions. (ii) Time-slotted protocols: The spectrum is divided into fixed slots, and nodes can only use the

The associate editor coordinating the review of this manuscript and approving it for publication was Ryan Thomas.

spectrum during slots defined by some scheduling algorithm. (iii) Hybrid protocols: Some protocols use partially slotted transmission. Control signalling makes use of synchronised time slots, while data transmission may use random access protocols without time synchronisation. Alternatively, random access protocols are extended with a slotted structure, so transmissions can only start at specific fixed times. All these techniques focus on optimising the technology's own performance, without taking the other's performance into account. To the best of our knowledge, there is no protocol that is able to react to cross-technology behaviour to avoid cross-technology interference.

Optimising the spectrum for cross-technology environments is complex because of the diversity of technologies, the wide variety of different applications and the mobility of nodes. This variety and mobility make the problem very dynamic and hard to solve with naive approaches.

In this work, we consider a multi-channel access problem where $N$ nodes have access to $C$ channels in an environment where other networks (possibly using other wireless technologies) use the same portion of the spectrum. Nodes at different positions have different views of the spectrum. We assume that the MAC protocol used at the own network is a slotted MAC protocol, such as Multiple Frequencies Time Division Multiple Access (MF-TDMA) or slotted-CSMA where the receiver decides when and on which channel to perform transmissions. This could be achieved through a (centralised or decentralised) scheduler, which is not part of the scope of this paper. Nevertheless, our solution is compatible with those (centralised and decentralised) schedulers by providing additional information to them. Our proposed solution considers $S \times C$ slots, sorting them in increasing order of predicted noise level. Then, the used MAC algorithm could select slots with less predicted noise first. Note that we define noise as spectrum usage by other networks. We call this set of unknown networks the Interfering Networks Cluster (INC). To optimise the spectrum, we assume that we want to optimise the overall throughput of all networks in the environment, both the own network as well as the networks in the INC. In other words, we optimise the overall throughput by predicting the behaviour of the INC and change our own behaviour by rescheduling slots if necessary. In this paper, we focus on the prediction of the behaviour of the INC.

In general, the problem can be formulated as a partially observable stochastic game, which is a natural extension of a Markov Decision Processes (MDPs) to multi-agent scenarios. Because of the large number of actions, the number of permutations of $S \times C$ slots is $(S \times C)!$, the action space should be considered continuous and solving this problem with Reinforcement Learning (RL) is hard [4]. We investigate the use of deep multi-agent supervised online learning as a way to enable learning in an unknown environment by predicting the spectrum usage (at every node) for upcoming slots. We define a loss function that allows optimising the prediction based on partially observable data. Our online approach makes the proposed algorithm robust against unknown behaviour and new environments. With this approach, we use techniques of RL and supervised learning.

To the best of our knowledge, this paper is the first attempt to provide a Multi-Agent Machine Learning (MAML) solution that predicts the spectrum usage to optimise spectrum sharing and increases the overall throughput of all networks in the environment. We can use our approaches with different unknown sources, and evaluate the algorithm in a realistic simulator with multiple topologies and scenarios. This work focuses on the prediction of the spectrum usage of the INC in a multi-agent partially observable environment while using the spectrum itself, by training a DNN. We run simulations to compare our algorithm with traditional slot selection algorithms. Also, this algorithm was implemented in the SCATTER radio [5], a wireless radio developed by the two times prize winner SCATTER team that reached 6th place in the DARPA Spectrum Collaboration Challenge (SC2).[1] In this competition, teams were challenged to build a collaborative wireless radio to improve the Quality of Service (QoS) of all the networks in the same collision domain [6], [7].

The remainder of this paper is structured as follows: we first discuss the related work and state of the art in section II. Secondly, we mathematically describe the problem in section III. In section IV, we discuss our proposed algorithm and architecture, while in section V, we discuss the implementation in the simulator and the SCATTER wireless radio system. The results are shown in section VI where we compare the algorithm with other techniques in simulation and a real setup by using the DARPA SC2 testbed. Finally, we conclude this work in section VIII.

## II. RELATED WORK

Spectrum and channel management have been widely studied. Traditional channel management algorithms focus on avoiding collisions within one network. Nowadays, a lot of different algorithms aiming at avoiding collisions or prioritising different users exist. In recent years, Machine Learning (ML) approaches to optimising certain network management problems have gained popularity. In this section, state of the art concerning collision avoidance techniques, CR networks and ML in wireless networks are discussed.

### A. COLLISION AVOIDING TECHNIQUES

As described previously, three main approaches to collision avoidance exist. (i) Random Access protocols: The simplest (and still widely used) Collision Avoiding technique is ALOHA [8]. With this generic and easy-to-use approach, nodes use the medium at any time and retry after a random back-off time if transmission fails. Nowadays, a lot of Random Access protocols are extended with CSMA [9]. This technique listens to the medium and only starts transmitting if the medium appears free. It sends the packet after a random interval, starting from the moment it no longer notices any activity on the medium. This can be extended

[1] https://www.spectrumcollaborationchallenge.com

with an RTS/CTS (Request To Send/Clear To Send) flow control mechanism. This method tries to avoid collisions by creating coordinated access to the medium using control signals between nodes. RTS/CTS communication cannot be used between different networks or across network technologies, while carrier sensing strategies also listen to other network technologies. It is known that some CSMA networks (such as Wi-Fi) may face starvation if other network technologies are using the same spectrum with no listen before talking mechanisms [10]–[12]. (ii) Time-slotted protocols: Another way of using the spectrum is by defining fixed time slots. This is called Time Division Multiple Access (TDMA), or MF-TDMA if we also subdivide the available bandwidth into different channels. These MAC protocols need network-wide synchronisation so that every node can start listening and receiving precisely at the intended time, as defined in the MF-TDMA schedule. Over the last years, we have seen more and more technologies using TDMA schemes, especially in sensor networks [13] and Vehicular Ad hoc Networks (VANETs) (such as STDMA, SOFTMAC, and TC-MAC.) [14]. Also, MF-TDMA MAC protocols are used more frequently in newer technologies such as 6TiSCH (Time Synchronized Channel Hopping) [15]. (iii) Hybrid protocols: Some protocols use a partially slotted transmission. Control signalling can make use of synchronised time slots, while data transmission may use random access protocols without time synchronisation. Other protocols use techniques of Random Access protocols, but all the transmissions start on predefined slots (e.g. Slotted ALOHA).

### B. COGNITIVE RADIO NETWORKS

The most natural solution to solve the spectrum scarcity problem is to increase the number of frequency bands. CR allows users (denominated Secondary Users (SUs)) to use other frequency bands (e.g. licensed frequency bands) as long as they are free and not used by Primary Users (PUs) (the actual users of the frequency band) [16]. Different techniques, such as Spectrum Decision, Spectrum Sharing and Spectrum Mobility, have been proposed. Most of these algorithms also use Spectrum Sensing where the SU senses the spectrum to decide if it is free and could be used for its own traffic.

Much work was done in the last years to support CR Networks (CRNs). Not only at the physical layer where spectrum sensing techniques are implemented, but also on the MAC, where sensing scheduling schemes, sensing-access tradeoff design, spectrum-aware access MAC, and CR MAC protocols are proposed [17]. These MAC protocols optimise the use of the spectrum by using the spectrum sensing performed by the physical layer to avoid the PU. Most of these proposed algorithms still try to optimise their own traffic, as long they are not interfering with the PU. On the network layer tomography, spectrum-aware routing, and QoS control could help. This is addressed in the literature to increase the performance of current CRNs [17]. This network layer is outside the scope of this work.

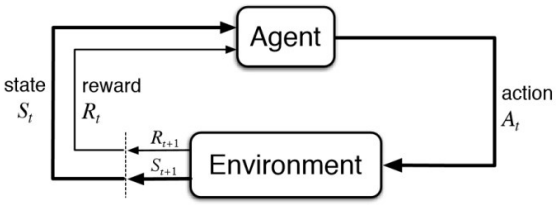



**FIGURE 1.** Reinforcement learning.

In contrast with CRNs, we use another technique to support QoS and optimise spectrum usage. We assume that all networks in the environment have equal priority, and no additional spectrum can be used. We don't use the terms PU and SU but own network (i.e., the network we can control) and the INC, the set of interfering networks.

### C. MACHINE LEARNING FOR WIRELESS NETWORK MANAGEMENT

ML in Wireless Networks is a growing research domain. Many decision-making problems in wireless networks, such as vertical handovers in heterogeneous networks [18] and power allocation in energy harvesting communication systems [19], are already using ML techniques. These optimisation algorithms can typically be described as an MDP. In contrast, the multi-channel access problem is usually described as a Partially Observable Markov Decision Process (POMDP) or, in a multi-agent setup, as a partially observable stochastic game.

If the model can be described as an MDP or a POMDP, RL is a widely used technique to derive a policy. With RL, an agent acts in an environment. By experimenting in the environment, the agent receives rewards or penalties and learns to optimise the reward in different games as visualised in Figure 1 [20]. Different techniques are described in literature to solve RL problems, by using simple algorithms as Q-learning, or more complex solutions to solve more complex problems by using DNNs [21], even in multi-agent environments [22].

ALOHA-QIR, proposed by Chu *et al.* is an algorithm that uses Q-Learning to improve slotted ALOHA [23]. With this approach, they can double the maximum throughput of Slotted ALOHA while increasing the energy-efficiency. ALOHA-QIR creates a kind of superframe that consists of $S$ slots on top of the structure of naive slotted ALOHA. Each slot in the superframe represents a score in the Q-Table. The reward used in the Q-Learning algorithm of this approach is based on whether the transmission was successful or resulted in a collision or failure. Combining ALOHA and Q-Learning results in a light-weight and fitting protocol for most Wireless Sensor Networks (WSNs). One of the disadvantages of this approach is its poor scalability. As the number of nodes and the amount of data grows, the ALOHA-QIR algorithm does not scale because it can only use a single channel. A more dynamic approach is proposed by Wang *et al.* [24].

They modeled the Dynamic Multi-channel Access problem as a POMDP. At the start of every time slot, a user selects one channel to sense the spectrum and transmits a packet. They search for a policy by using a Deep Q-Network (DQN), proposed by Google Deepmind [25]. By classifying each channel in one of two states, *good* or *bad*, they are able to learn a near-optimal policy by using online learning. However, this approach will not work in realistic and complicated scenarios with multiple users and simultaneous transmissions because it does not scale very well [24]. Also, the employed action-space is relatively small. If the action space becomes large when selecting multiple timeslots, a redesign of the DQN is necessary. In contrast, our solution optimises the multi-agent and multi-channel problem using partially observable supervised learning. This provides better scalability in terms of nodes and size of the superframe, both in number of channels as number of time slots.

## III. PROBLEM FORMULATION

In this section, we state the problem formally and mathematically. Used symbols are described in Table 1 and defined later in this section.

In this work, we consider a network of $N$ nodes, called the own network, and a second (unknown) set of networks, called the Interfering Networks Cluster (INC). We assume that the nodes of our own network have access to $C$ channels, and use a slotted MAC protocol. This MAC protocol could be an MF-TDMA protocol, or a slotted random access protocol. We define a superframe with $S$ time slots. When using an MF-TDMA protocol, this superframe is equal to the MF-TDMA superframe. Otherwise, we define a superframe length $S$. This artificial superframe could be used together with a slotted random access protocol.

We define $t$ as a global reference of time. $t$ is the number of slots that have passed since the start of the first node in the environment. Because we have a slotted MAC protocol, all nodes in our own network start a slot at exactly the same time. This is a general property of a slotted MAC protocol. Every node $n$ executes an action $A^n_{t,c} \in \{\text{Idle}, TX, RX\}$ at time $t$ for every channel $c$. Furthermore, we denote $TX^n_{t,c}$ and $RX^n_{t,c}$ as:

$$TX^n_{t,c} = \begin{cases} 1 & \text{if } A^n_{t,c} = TX \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$RX^n_{t,c} = \begin{cases} 1 & \text{if } A^n_{t,c} = RX \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Each node executes exactly $C$ actions at a specific time $t$, one action for each channel at every time slot. Note that *Idle* is part of the action space.

$$\forall n \in [1, N], \quad \forall c \in [1, C], \ \forall t : TX^n_{t,c} + RX^n_{t,c} \leq 1 \tag{3}$$

The INC is completely unknown. As a result, there is an unknown number of nodes in the INC, using unknown protocols, and even noise generators, jammers, could be part of the INC. We define $P_{t,c}$ as the usage of channel $c$ at time $t$ by the INC, where $P_{t,c} = 1$ if a source in the INC used channel $c$ at time $t$, and $P_{t,c} = 0$ otherwise. Due to fading, power control, and the Modulation and Coding Scheme (MCS) setting, the interference of a generated signal is different for different nodes at different positions. We define $P^n_{t,c}$ as the usage of channel $c$ at time $t$ by the INC with energy above the noise threshold in the interpretation of node $n$. $P^n_{t,c} = 1$ if and only if the INC produces energy that, at node $n$ at time $t$ on channel $c$ is above the noise threshold.

**TABLE 1. Overview of used symbols.**

| Symbol | Description |
|---|---|
| $N$ | Number of nodes in the own network. $N \in \mathbb{N}$ |
| $C$ | Number of channels used in the own network. $C \in \mathbb{N}$ |
| $S$ | Number of time slots used in the (artificial) super frame $S \in \mathbb{N}$ |
| $t$ | Global reference of time started at the start of the first radio in the environment $t \in \mathbb{N}$ |
| $n$ | Node number $n \in [1, N]$ |
| $c$ | Channel number $c \in [1, C]$ |
| $H$ | Number of history frames in a sample. $H \in \mathbb{N}$ |
| $A^n_{t,c}$ | Action of node $n$ at time $t$ on channel $c$ |
| $TX^n_{t,c}$ | 1 if $A^n_{t,c} = TX$, 0 otherwise |
| $RX^n_{t,c}$ | 1 if $A^n_{t,c} = RX$, 0 otherwise |
| $P_{t,c}$ | 1 if the INC used channel $c$ at time slot $t$, 0 otherwise |
| $P^n_{t,c}$ | 1 if node $n$ of the own network received energy on channel $c$ from a source inside the INC, 0 otherwise |
| $\gamma_{t,c}$ | 1 if there could be a successful transmission within our own network, 0 otherwise |
| $\gamma'_{t,c}$ | 1 if there could be a successful transmission within the INC, 0 otherwise |
| $\hat{\Gamma}$ | The maximum network packet throughput of our own network |
| $\hat{\Gamma}'$ | The maximum network packet throughput of the INC |
| $\Gamma$ | The actual network packet throughput of our own network |
| $\Gamma'$ | The actual network packet throughput of the INC |
| $\alpha$ | Relative priority between the own network and the INC. $\alpha \in [0, 1]$. Where 0 indicates own priority and 1 indicates INC priority. |
| $\digamma(\alpha)$ | The overall network throughput of both the own network and the INC |
| $\omega_{t,c}$ | 1 if there is a potential collision between the INC and the own network |
| $\Omega$ | Maximum number of collisions between the own network and the INC |
| $O^n_{t,c}$ | Amount of energy detected by $n$ at time $t$ on channel $c$ |
| $TX_{t,c}$ | 1 if a node in the own network is executing action $TX$ at time $t$ on channel $c$, 0 otherwise |
| $\xi^n_{t,c}$ | 1 if node $n$ received a packet correctly on time $t$ on channel $c$, 0 otherwise |
| $p^n_{t,c}$ | Output of the prediction unit of node $n$ for time slot $t$ at channel $c$. $p^n_{t,c} \in [0, 1]$ |
| $\pi^n_t(O^n, TX)$ | Policy predicted by the prediction unit of node $n$ at time $t$ based on observation $O^n$ and schedule $TX$. $\pi^n_t(O^n, TX) : \mathbb{R}^{H\times S\times C}, \{0, 1\}^{H\times S\times C} \mapsto \mathbb{R}^{S\times C}$ |
| $\pi^{*n}_t(O^n, TX)$ | Theoretical optimal policy of node $n$ at time $t$ based on observation $O^n$ and schedule $TX$. $\pi^{*n}_t(O^n, TX) : \mathbb{R}^{H\times S\times C}, \{0, 1\}^{H\times S\times C} \mapsto \mathbb{R}^{S\times C}$ |
| $\phi$ | Noise threshold |

As described above, the goal of this work is not to design a new MAC protocol or algorithm but rather to improve the available information as an enabler to use the spectrum smarter and more efficiently. For the remainder of this paper, we assume that the used MAC protocol avoids all collisions in the own network and for every transmission, there is at least one node listening (this could be realised by a centralised MF-TDMA algorithm for example). Communication between two different nodes in our own network is successful if and only if at a given time slot $t$ at least one node is in an *RX* state and only received a message from exactly one other node of the own network at a given channel $c$. As described above, we assume that the used MAC protocol avoid collisions within our own network:

$$\forall c \in [1, C], \forall t : \sum_{n \in [1,N]} TX^n_{t,c} \leq 1 \tag{4}$$

Based on this assumption, we define that communication of the own network at time $t$ on channel $c$ is successful if one own node is transmitting on channel $c$ where the destination node has no interference of the INC. We use $\gamma_{t,c}$ to determine if an own packet could be delivered successfully. Note that there is always a chance that the communication fails because of the environment (e.g. fading, etc.).

$$\gamma_{t,c} = \begin{cases} 1 & \text{if } \sum_{n \in [1,N]} TX^n_{t,c} = 1 \wedge \\ & \quad \sum_{n \in [1,N]} RX^n_{t,c}(1 - P^n_{t,c}) = 0 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Because the INC is unknown, there is, from an external point of view, no way to know if the observed INC spectrum usage indicates successful packet transmission. We merely assume that communication in the INC is successful if $\gamma'_{t,c} = 1$, where:

$$\gamma'_{t,c} = \begin{cases} 1 & \text{if } \sum_{n \in [1,N]} TX^n_{t,c} = 0 \wedge P_{t,c} = 1 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

We define the maximum network packet throughput, $\hat{\Gamma}$, as the sum of all successful transmissions:

$$\hat{\Gamma} = \sum_t \sum_{c \in [1,C]} \gamma_{t,c} \tag{7}$$

In the same way, we define the maximum network packet throughput of the INC:

$$\hat{\Gamma}' = \sum_t \sum_{c \in [1,C]} \gamma'_{t,c} \tag{8}$$

We define $\Gamma$ and $\Gamma'$ as the actual packet throughput for the own network and INC respectively.

The objective of this algorithm is to maximise the overall network throughput of all the networks, own network and the networks of the INC, by choosing the actions $A$ as optimally as possible for all channel-timeslot pairs for all nodes $n$ in the own network. We define the overall network throughput of superframe as follows:

$$F(\alpha) = \alpha\Gamma + (1 - \alpha)\Gamma' \text{ with } \alpha \in [0, 1] \tag{9}$$

where $\alpha$ denotes the relative priority between the own network and the INC, where 0 indicates priority for only the own network, while 1 indicates priority of the INC network.

Note that to increase the throughput $\Gamma$ and $\Gamma'$, we can either (i) increase the number of transmissions (*TX*/*RX* actions), (ii) generate more data, or (iii) decrease the number of collisions between the INC and the own network. We assume that we have no control about the data generation and can only control nodes in the own network. The only way to increase the throughput is by reducing collisions.

We define $\omega_{t,c}$ to be 1 if there was a potential collision between the INC and the own network the slot at time $t$ on channel $c$, and $\Omega$ as the total number of potential collisions.

$$\omega_{t,c} = \begin{cases} 0 & \text{if } \sum_{n \in [1,N]} TX^n_{t,c} + P_{t,c} \leq 1 \\ 1 & \text{otherwise} \end{cases} \tag{10}$$

$$\Omega = \sum_t \sum_{c \in [1,C]} \omega_{t,c} \tag{11}$$

As we assume there are no collisions in the own network (Equation 4), the only way to maximise $F(\alpha)$ is to minimise $\Omega$.

Furthermore, we assume that every node $n$ can measure energy on all $C$ channels. Based on new hardware radio technologies (such as Software Defined Radios (SDRs) [26]) supporting simultaneous energy measurement over a broad bandwidth is possible [27]. We define $o^n_{t,c} \in [0, 1]$ as the observation made at time $t$ on channel $c$ for node $n$. We define that $o^n_{t,c} = 1$ if the amount of energy detected on channel $c$ on time $t$ for node $n$ is so high that it was not possible the receive any (other) packet. Note that all energy is detected, including energy generated by the node itself or neighbouring nodes of the own network.

We define $TX_{t,c}$ as the global *TX* action in the own network. Where $TX_{t,c} = 1$ if and only of at least one node of the own network is transmitting data at timestamp $t$ on channel $c$:

$$TX_{t,c} = \bigvee_{n \in [1,N]} TX^n_{t,c} \tag{12}$$

Finally, we define $\xi^n_{t,c}$ as the global indicator of successful packet delivery within our network. $\xi^n_{t,c} = 1$ if and only if $RX^n_{t,c} = 1$ and node $n$ received a packet correctly at time $t$ on channel $c$.

Each node also contains some network information and statistics. In this work we assume we have access to the following node and network information: (i) each node knows if a slot is allocated inside the network or by using Equation 12 if $TX_{t,c} = 1$. In most MF-TDMA protocols such information is (partially) known at the nodes and for CSMA networks this could be implemented by the use of RTS/CTS strategies. (ii) Each node knows if packets are received correctly: $\xi^n_{t,c} = 1$

## IV. FRAMEWORK

### A. ARCHITECTURE

As illustrated in Figure 2, the proposed approach consists out of five main components: the spectrum monitor,

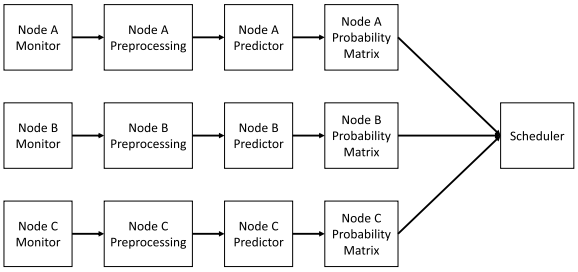



**FIGURE 2. Architecture of the algorithm for an own network of three nodes.**

preprocessing unit, predictor unit, probability matrix on each node and the overall (centralised or decentralised) scheduler. The first component of our architecture is the spectrum monitor. Each node captures the energy on the overall spectrum [28]. This monitor information is forwarded to the next component, the preprocessing unit. The preprocessing unit of node $n$ creates the correct observation values $O^n_{t,c}$, together with the network schedule $TX_{t,c}$, the node schedule $A^n_{t,c}$, and success information $\xi^n_{t,c}$ for node $n$, for every timestamp $t$ in the last (artificial) superframe on each channel $c$.

For every (artificial) superframe, the predictor unit predicts the upcoming spectrum usage generated by the INC. We define $p^n_{t,c}$ as the result of the prediction unit where $p^n_{t,c} \in [0, 1]$ indicates if the slot is predicted as highly used (value close to 0) or is predicted as free (value close to 1). These $p^n_{t,c}$ values form a probability matrix. These probability matrices are used by the scheduler to select slots expected to be free, as to avoid collisions. Note that the scheduler could be a (centralised or decentralised) MF-TDMA scheduler, or a slotted CSMA Access Point (AP) that defines when nodes could transmit. This could be realised in a MF-TDMA network where a sender node transmits a request for slots to the receiver, to which the receiver replies with the best available slots based on the prediction.

Note that there are clear differences with our previous work described in [29]. In [29], the observation $O^n_{t,c}$ was defined as the energy detected at node $n$ at time $t$ at channel $c$ produced by the INC. This definition is unobtainable in realistic scenarios because there is no way of subtracting the energy produced by the own network from the observation. Only in simulators it was possible to obtain this value. In this work, we extended the framework, algorithm and ML model in such a way that the observation is realistic; $O^n_{t,c}$ is the overall energy (i.e., own plus INC energies) detected at node $n$ at time $t$ at channel $c$.

### B. PREDICTION UNIT

As described above, the predictor unit predicts the used slots in the upcoming superframe used by the INC. These predictions could be used by the scheduler to take an action to select slots. We define the prediction as $\pi^n_t(O^n, TX) : \mathbb{R}^{H\times S\times C}, \{0, 1\}^{H\times S\times C} \mapsto \mathbb{R}^{S\times C}$, the policy predicted by the prediction unit of node $n$ at time $t$ based on observation matrix $O^n$ and schedule matrix $TX$. $H$ defines a number of history frames. Additionally, we define $\pi^{*n}_t(O^n, TX)$ as the theoretically optimal policy.

If we want to model this problem as an RL problem, we need to define the actions. An action could be defined as selecting $m$ 'free' slots to the correct destination, where $m$ is the number of slots required to deliver the data. In our approach, we assume that $m$ is unknown and that the scheduler takes care of it. Therefore, we describe our action as sorting all the slots of a superframe. If we want to describe this problem discretely, our action is sorting $S \times C$ slots, where the first slot of the sequence is the best slot to use. As such, there are $(S \times C)!$ different actions in this discrete case. The scheduler decides how many slots are needed, and can just select slots based on the order. Intuitively, the interesting action space seems smaller, because we are only interested in the $m$ best slots. However, because each node is generating a prediction and only free slots can be selected, it is possible that non-optimal slots will be selected. If the number of slots that need to be selected grows, reducing the action space is less interesting.

Another way of representing the action space defined previously is by using a continuous space and representing our problem as a stochastic game with a continuous action space. As an advantage, this would make labelling during online playing easier. On the other hand, this type of game is very complex and hard to solve. It is still a domain of interest as no successful attempts have been made so far to derive learning dynamics for this setting [4].

On the other hand, there is a lot of information we could teach the system based on energy observations and successful transmissions. This inspired us to combine supervised learning with RL techniques to solve this problem. We describe three main components: the State, the Label and Loss function, and the Neural Network (NN):

#### 1) STATE

The input features represent the state of the environment in the last $H$ superframes. We describe the state as $[O^n, TX]$, with observation matrix $O^n \in \mathbb{R}^{H\times S\times C}$ and network schedule information $TX \in \{0, 1\}^{H\times S\times C}$. Each element of $O^n$ corresponds to a value $o^n_{t,c}$, and element of $TX$ corresponds to a value $TX_{t,c}$. We use this state representation to indicate when observations could be used to recognise patterns. If $TX_{t,c} = 1$ we know that a node is using the spectrum at time $t$ in channel $c$. This implies that the observation $O^n_{t,c}$ could be based on our own spectrum usage. $TX$ helps to indicate our own possible spectrum usage.

#### 2) LABEL AND LOSS

To construct the label, we define slots as *good*, *bad*, or *unknown*. After defining $\phi$ as the noise threshold, *Good* and *Bad* slots are defined as:

$$Good^n_{t,c} = O^n_{t,c} < \phi \vee \xi^n_{t,c} \tag{13}$$

**TABLE 2. Hyperparameters NN used in the predictor unit.**

| Hyperparameter | Value |
|---|---|
| Minibatch size | 256 |
| Optimiser | RMSProp |
| RMSProp momentum | 0 |
| RMSProp $\epsilon$ | $10^{-10}$ |
| RMSProp decay | 0.9 |
| Learning rate | $10^{-4}$ |
| Experience replay buffer size ($M$) | 20,000 |
| $\delta$ | 10 |

$$Bad^n_{t,c} = \neg Good^n_{t,c} \wedge \left(\neg TX_{t,c} \vee RX^n_{t,c}\right) \quad (14)$$

$$Unknown^n_{t,c} = \neg\left(Good^n_{t,c} \vee Bad^n_{t,c}\right) \quad (15)$$

In Equation 13, we define good slots as slots where the amount of energy detected is lower than the noise threshold, along with those containing a successful transmission. Otherwise, as stated in Equation 14, if a slot is not good and not part of the overall schedule or part of the own node receiving schedule, we define a slot as being *bad*. We define all slots in the superframe that are not *good* nor *bad* as *unknown*, as shown in Equation 15.

We define the loss in Equation 16 as the cross entropy loss of the *good* and the *bad* labels, divided by the number of slots we could update. Note that the loss of *unknown* slots is 0 and the NN will try to maintain the output value of these slots. The divisor normalises the loss, ensuring the NN will not simply attempt to label as many slots as possible as *unknown*.

$$\mathcal{L} = -\frac{\sum_{t,c} Good^n_{t,c} \log\left(\pi^n_{t,c}\right) + Bad^n_{t,c} \log\left(1-\pi^n_{t,c}\right)}{\sum_{t,c} Good^n_{t,c} + Bad^n_{t,c}} \quad (16)$$

#### 3) NEURAL NETWORK

In this work, we use a DNN to optimise the prediction. As shown in Figure 3, we use 8 fully connected layers with for each layer $S \times C \times \lfloor \frac{(8-l)H}{8} \rfloor \times 2$ neurons, where $l$ is the index of the layer, with $l \in \mathbb{N} : 0 \leq l < 8$. We use a swish activation function [30] on each layer except for the last (output) layer, where a softmax activation is applied on the last dimension. The softmax activation function ensures that each cell in the output matrix, which represents the prediction, is a value between 0 and 1, predicting the probability of the slot being used by the INC.

Because data generation, schedule policies, network topology, and the environment could change drastically, online learning is necessary. It is almost impossible to train the predictor unit offline on a comprehensive distribution. Therefore we apply online learning with experience replay. After every $\delta$ steps we execute a training step where we randomly select samples from the replay buffer. To optimise the weights in the NN we use an RMSProp optimiser. All hyperparameter values used in this work are defined in Table 2.

### C. MULTI-AGENT EXAMPLE

Each node in the network runs the entire framework. Each node constructs its own state by using its own RF information and network information. Each node uses its own prediction unit and each node executes its own online learning process. Each node has its own prediction and this has a direct influence on the slot selection. Note that selecting a slot is actually (partly) a network decision. At least two nodes should agree on allocating a slot. The advantage of using this framework is that this negotiation to select slots is done by the (already existing) slot scheduling algorithm that is independent of the framework. The resulting action taken by the scheduler is part of the state and has an influence on the behaviour of every node. All nodes are working almost independently while they optimise a network objective, increase the spectrum efficiency, or as discussed in section III, decrease the inter-network collisions.

As an example, suppose node A wants to transmit a packet to node B in the own network. The two nodes have a separate observation and different influences of other networks in the wireless spectrum. If A wants to select a slot to B, node A needs to negotiate with node B. Node B can indicate which slots are better to receive the message based on the information of the prediction. The negotiation and the slot allocation process are done by the scheduler. If a slot is selected, it can be used in the next superframe. Once the superframe is finished, each node labels each slot as *Good*, *Bad* or *Unknown* by using Equation 13, Equation 14 and Equation 15. This label is based on the local information of the node. The sample (i.e., the state and the label) will be stored in the own replay experience buffer of each node. Each node uses its own replay buffer to retrain the network in an online way.

## V. EXPERIMENTAL SETUPS

To evaluate the proposed algorithm, we performed simulations and executed a real experiment using the testbed built especially for the DARPA SC2 competition [6].

### A. SIMULATION SETUP

To evaluate the framework in simulation, we used an MF-TDMA discrete event simulator written in Python based on the 6Time Synchronized Channel Hopping (TiSCH) simulator of Palattella *et al.* [31], combined with RF-data we monitored during the SC2 competition. During this part of the competition, 19 different teams were involved. In previous work [29], we already modified the simulator to make it possible to have two separate networks, where one of them does not use the MF-TDMA schedule, but only generates spectrum usage.

As shown in Figure 4, we simulated two different network topologies: (i) a ring network, Figure 4a, and (ii) an AP topology, Figure 4b. In both topologies, there are 6 nodes of the own network (blue), and two nodes that represent the INC network (grey). In the ring network, we use an MF-TDMA network, with each node generating data to be forwarded to a neighbour. In the AP network, we simulate a network in infrastructure mode with one central AP and 5 clients. The 5 clients generate data while the AP replies to any successfully delivered message. With this topology,

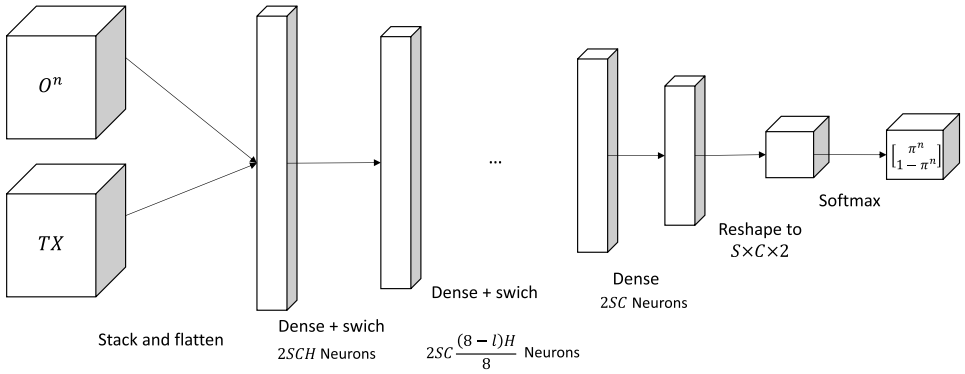
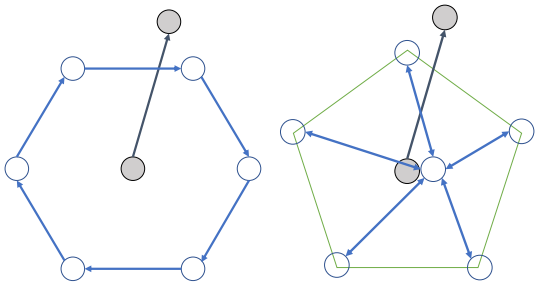



**FIGURE 3.** Visualisation of the DNN.



(a) Ring topology (b) AP topology

**FIGURE 4.** Topologies and routing of the simulated test network with one unknown network (grey) representing the INC, and 6 own nodes (blue).

we use an MF-TDMA network and a CSMA network with an RTS/CTS control mechanism. In both setups, traffic is generated with a uniform random probability. We assume that a scheduler can schedule the slots in the superframe at the start of that superframe. In all setups, we use a superframe of 20 time slots and 4 channels.

### B. DARPA SC2 COMPETITION EXPERIMENTAL SETUP

In 2017, the Defense Advanced Research Projects Agency (DARPA) organised a three-year competition aimed at optimising spectrum usage by using Artificial Intelligence (AI) in real networks. This competition was called the Spectrum Collaboration Challenge (SC2) [6]. We participated in this competition as team SCATTER and reached sixth position. In the SC2 competition, different teams played in the same environment with 10 nodes each. All teams received traffic they needed to deliver. If the QoS requirements of a given flow are achieved for at least 10 seconds, a flow is called stable and every next second, for which the flow is still stable, the team received a number of points for the given flow. To ensure collaboration: if not all teams were able to achieve some minimal score, each team received the score of the worst-performing team. Because, for each scenario, many matches were played, collaboration was required, especially because the available spectrum was limited and incumbents or jammers could be active in the same environment. Each team was able to use their preferred technologies on the physical layer, MAC layer and network layer. The SC2 competition made use of a custom testbed called Colosseum. This testbed exists out of 128 wireless nodes, combined with an RF-simulator.

The proposed algorithm was implemented using Tensorflow,[2] while the preprocessing was implemented using Cupy.[3] A distributed scheduler is used. If a new slot needs to be selected, the sender sends a request to the receiver for slot allocation. The sender proposes a number of slots, while the receiver selects one of them, based on the value of their local prediction [5]. Other nodes can overhear this communication, as long as there is no interference. Only if they do receive these messages on the broadcast control channel, do they know that a slot is used by the own network.

In the experiments executed on the SC2 Colosseum, we played against two randomly picked other teams. The scenario used in our experiments was called *Alleys of Austin*. In this scenario, each team has 10 nodes (nine ground soldiers and one helicopter) moving around in Austin, Texas. There are three stages, each having progressively more data to be delivered, while the teams move closer and closer to each other.

To train the algorithm, we first played 30 matches with three teams. During these matches, we pre-trained the DNN by using the hyperparameters described in Table 2. To make this possible, we enabled a preprocessing step. During this preprocessing step, we mapped all the data to a 44 by 28 slot frame. This is achieved by using a bilinear interpolation.

During the competition, it was not possible to use the replay buffer across different matches. No data could be exchanged and each radio's state was reset after every match. The duration of every game was between 3.5 and 15 minutes, depending on the selected scenario. To optimise the learning process, we (i) increased the learning rate during the competition to $10^{-3}$, (ii) decreased the replay buffer size

[2]https://www.tensorflow.org
[3]https://cupy.chainer.org

to a maximum capacity of 5000 examples and (iii) increased the minibatch size to 512. This improves the prediction as quickly as possible for the current (unknown) match. Note that it was not feasible to share information about successfully received packets with all nodes, as was assumed to be possible in the simulations. In this setup, more slots are annotated as *unknown* due to missing information.

## VI. RESULTS DESCRIPTION

### A. SIMULATION RESULTS

In order to discuss our results fairly, we introduce four alternative approaches: (i) *Regular* is the approach that is used in most systems today. In order to reach a high throughput, the approach uses all free slots in the internal schedule. In other words, it uses only a traditional MF-TDMA schedule algorithm or CSMA strategy without having collisions in the own network. (ii) *No Sending* is an extreme approach where the own network will not send any packets at all. This is an extreme case, but shows us the optimal performance of the INC. (iii) *Optimal* is the hypothetical best solution to solve the problem when $\alpha = 0.5$. In this approach, the own network knows the behaviour of the INC up front, which is not possible in reality. Lastly, (iv) *DQN*, a strategy based on the work of Wang *et al.* [24]. Here, each node maintains its own DQN. As this did not scale to a superframe of 20 time slots, we instead used a superframe of size 1 for this approach. This reduced the complexity of the problem drastically from $4^{20}$ to 4 actions. We compare these strategies with our approach, which we call the *Prediction* approach. We trained our models on 22 scenarios for 10 episodes, while we evaluate all approaches on 36 unknown other scenarios.

#### 1) MF-TDMA RING TOPOLOGY

In Figure 5, we show the average Overall Throughput $F(\alpha)$ of a superframe. First of all, we see that *No Sending*'s performance decreases as $\alpha$ grows, since the own network is not using any slots at all. On the other hand, the *Regular* strategy's performance improves as $\alpha$ is growing. It makes sense that the *Regular* approach outperforms all other approaches if $\alpha$ is close to one, as it is the most aggressive approach. The *DQN* approach clearly underperforms, while our proposed *Prediction* approach is always performing at least as good as the *Regular* approach (when $\alpha < 0.8$) or the *Optimal* approach (when $\alpha > 0.8$). Note that the optimal strategy tries to optimise the problem for $\alpha = 0.5$, as explained in section V, meaning it may be outperformed by other approaches for different values of $\alpha$. In Figure 8b, it is shown that the *Prediction* approach could reduce the average number of collisions for each superframe by 30% in comparison to the *Regular* scheduler. For throughput in Figure 6, packet success ratio in Figure 7, and missed opportunities in Figure 8a, our *Prediction* approach is always in between the performance of the *Regular* scheduler and the *Optimal* approach. Note that in comparison to the *Regular* scheduler, our *Prediction* approach reduces the throughput slightly, as the *Prediction* approach is less aggressive. This leads to fewer collisions, but also some more missed opportunities if the module generates a false positive. Note that slots that are predicted with a *free* probability close to 0 are never selected by the scheduler. This results in fewer slots and lower throughput. The same is represented in the packet loss, illustrated in Figure 9. It is clear that the average loss, of the own network and the INC, for the *Prediction* approach is better than for the other approaches. However, the loss of the own network is a slightly higher for the *Prediction* approach in comparison to the *Regular* strategy. We can apply the same reasoning as with the throughput, where sending opportunities are not taken if the predicted slot value is too low, which makes the strategy less aggressive.

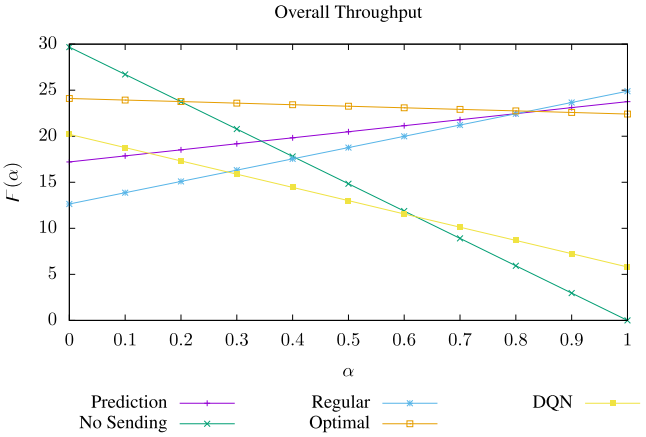
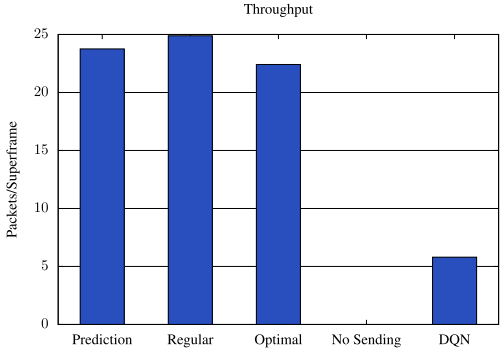



**FIGURE 5.** Average overall throughput $F(\alpha)$ of a superframe in the MF-TDMA mesh network and INC for different scheduling approaches.

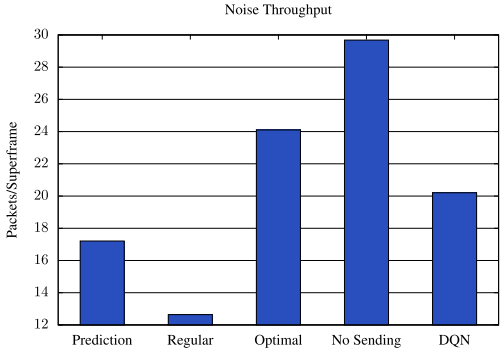
#### 2) MF-TDMA AP TOPOLOGY

If we change our topology from a full mesh to an AP approach, the throughput of our own network drops, due to the schedule of the AP reducing the potential for parallel communication using different channels. As the AP is positioned in the centre of the collision domain, it experiences a lot of interference. For this setup, Figure 10 shows that as long as $\alpha < 0.7$, the prediction approach outperforms Regular scheduling. More importantly, we also reduced the number of collisions by a factor of 1.3, as shown in Figure 13b, while the throughput of the own network is nearly unchanged and the throughput of the INC has increased by a factor of 1.17, as shown in Figure 11a and Figure 11b respectively. The INC packet success ratio (shown in Figure 12b) is 10% higher, because the predictor avoided upcoming collisions. The average number of missed opportunities per superframe with the *Prediction* approach is less than 7% higher than with the *Regular* scheduler. The average packet loss drops from 41% for the *Regular* strategy to 38% for the *Prediction* approach as shown in Figure 14.

To speed up the testing process, we did not evaluate the DQN approach, as it is clearly not feasible to run it in real cases and was previously outperformed by our approach.

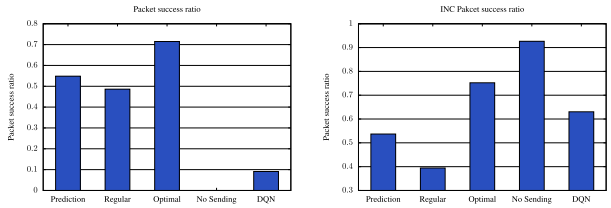



(a) $\Gamma$ of the own network



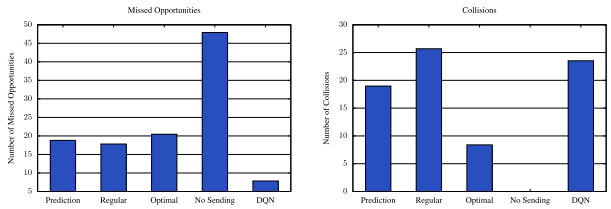

(b) $\Gamma'$ of the INC network

**FIGURE 6.** Average throughput per superframe for different scheduling approaches in an MF-TDMA mesh network.




(a) Own network (b) INC network

**FIGURE 7.** Average Packet Success Ratio per superframe for different scheduling approaches in an MF-TDMA Mesh network.




(a) Missed Opportunities (b) Collisions

**FIGURE 8.** Average number of missed opportunities and collisions for the own network in an MF-TDMA mesh scenario.

### 3) CSMA AP TOPOLOGY

In the CSMA setup, no slot is scheduled at the beginning of every frame. Nodes can send an RTS message to the AP from the moment they have a data frame ready. The AP can reply with a CTS at any time. Once that is received, the node can send the message. In comparison to the MF-TDMA AP setup, this will reduce the queue length of all nodes as transmission is no longer necessarily delayed until at least the start of the next superframe.

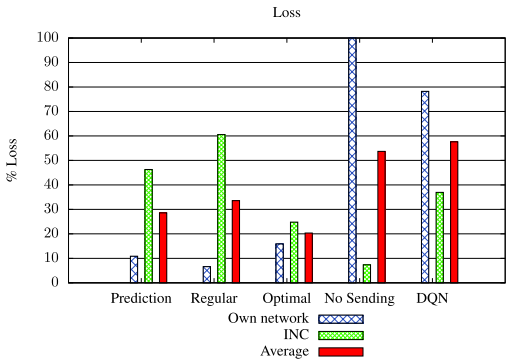



**FIGURE 9.** Packet loss of own network, INC and average of both in an MF-TDMA mesh scenario.

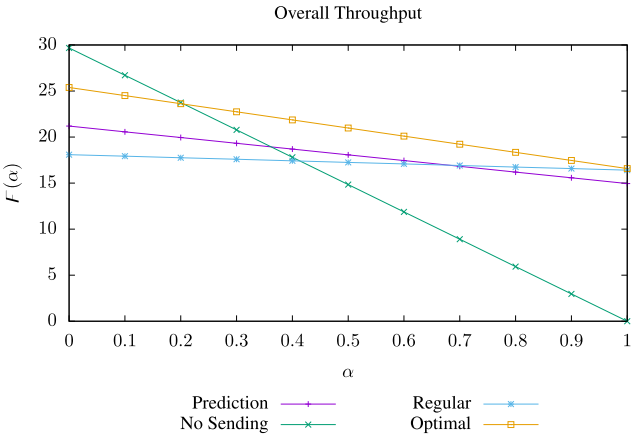
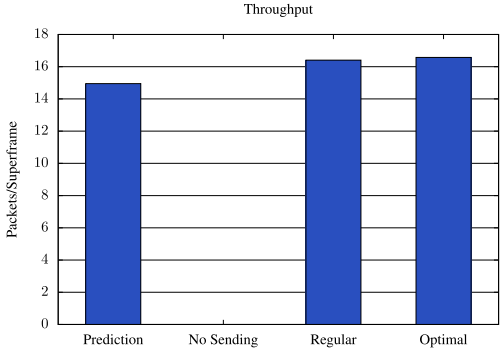
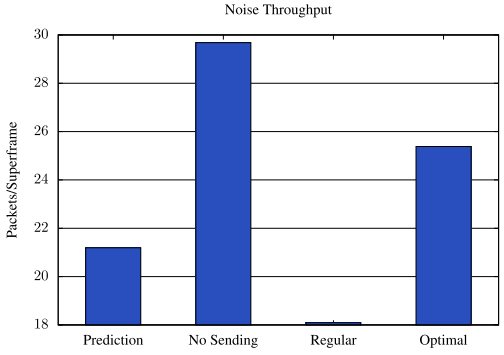



**FIGURE 10.** Average overall throughput $F(\alpha)$ of a superframe in the MF-TDMA AP network and INC for different scheduling approaches.

In Figure 15, it is shown that the overall throughput of our *Prediction* approach is always between the *Regular* scheduler and the theoretical *Optimal*. Figure 16b and Figure 17b clearly show that our proposed *Prediction* approach is more aggressive in the CSMA setup in comparison to the MF-TDMA setup. This is probably because the data transmissions start immediately, which makes it harder to learn a pattern, because more frames are labelled as unknown. In the MF-TDMA setup, only a few packets of the own network are scheduled in the first few frames. In this setup, the number of missed opportunities is similar for the *Prediction* approach and for the *Regular* scheduler, as shown in Figure 18a, as new data can be scheduled immediately, instead of having to wait for the next superframe. On the other hand, as shown in Figure 18b, we see that we reduced the number of collisions in comparison to the *Regular* scheduler by 20% on average for each superframe. This results in an

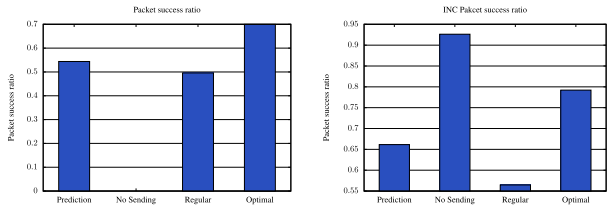



(a) Γ of the own network



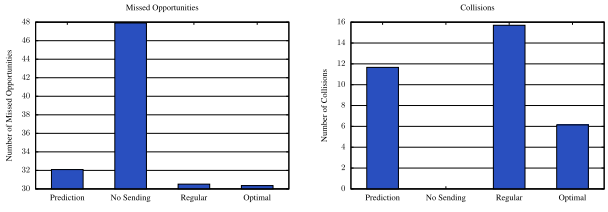

(b) Γ′ of the INC network

**FIGURE 11.** Average throughput per superframe for different scheduling approaches in an MF-TDMA AP scenario.




(a) Own network (b) INC network

**FIGURE 12.** Average packet success ratio per superframe for different scheduling approaches in an MF-TDMA AP scenario.




(a) Missed Opportunities (b) Collisions

**FIGURE 13.** Average number of missed opportunities and collisions for the own network in an MF-TDMA AP network.

almost equal loss for the own network by using the *Regular* strategy or the *Prediction* approach, but the loss of the INC drops from 34% to 30%, as shown in Figure 19.

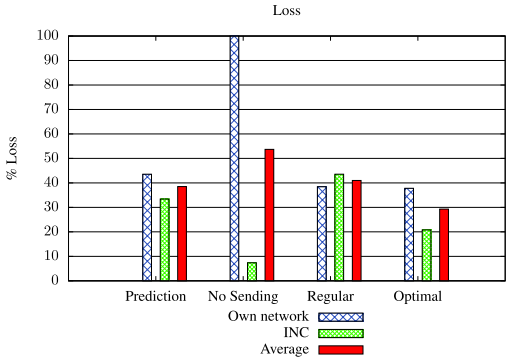



**FIGURE 14.** Packet loss of own network, INC and average of both in an MF-TDMA AP scenario.

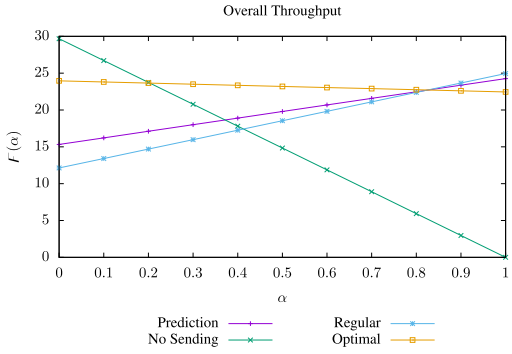



**FIGURE 15.** Average overall throughput $\Gamma(\alpha)$ of a superframe in the CSMA AP network and INC for different scheduling approaches.

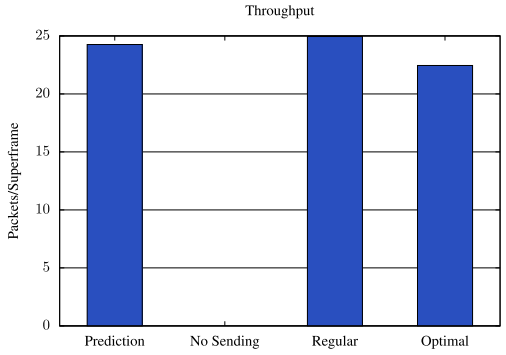
**TABLE 3.** Experimental results.

| Parameter | EWMA algorithm | Prediction algorithm |
|---|---|---|
| Throughput | 20 Mb/sec | 22 Mb/sec |
| Packet success Ratio | 91.8% | 92% |
| # of move operations | 1473.7 | 1352.5 |
| Individual score | 33188.1 | 36005.35 |
| Game score | 143.0 | 151.3 |

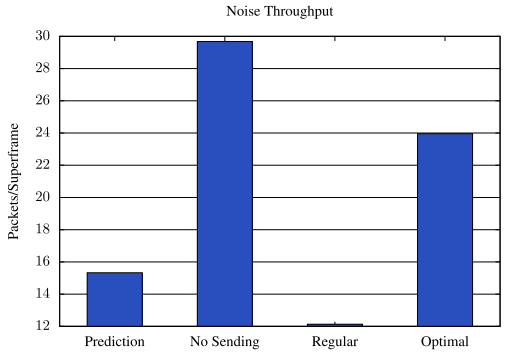
### B. EXPERIMENTAL RESULTS

As mentioned previously, the algorithm was also implemented in the SCATTER radio [5] . To compare our algorithm in the entire system, we compared against an Exponentially Weighted Moving Average (EWMA) collision avoidance slot selection algorithm. This algorithm simply selects new slots based on a historical moving average of the detected energy given a specific slot. After the pre-training, as mentioned before, we executed two times 16 matches, where we played against two other randomly selected competitors during the final year of the SC2 competition.

In Table 3, we show that the average throughput of the own network increases from 20Mb/sec to 22 Mb/sec. The collision avoidance rate is represented by the game score of a match (this value depends on how well other teams are consistently maintaining traffic) which increases by almost 10 points on average. The stability of the own network is measured by

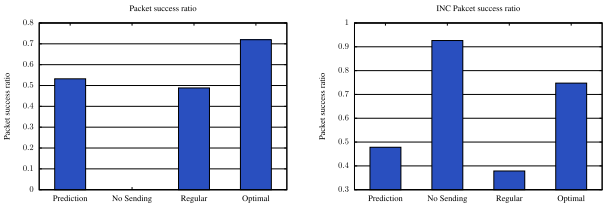



(a) $\Gamma$ of the own network



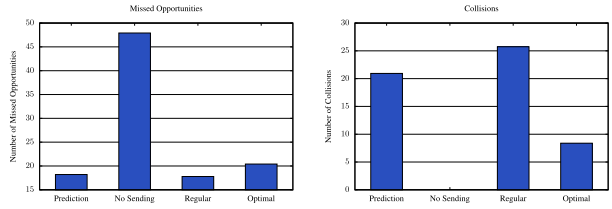

(b) $\Gamma'$ of the INC network

**FIGURE 16.** Average throughput per superframe for different scheduling approaches in a CSMA AP scenario.




(a) Own network (b) INC network

**FIGURE 17.** Average packet success ratio per superframe for different scheduling approaches in an a CSMA AP network.




(a) Missed Opportunities (b) Collisions

**FIGURE 18.** Average number of missed opportunities and collisions for the own network in a CSMA AP scenario.

the Individual Score. Given the rules of SC2, our system starts scoring after achieving the QoS requirements of a given traffic flow for at least 10 seconds, with a boost of almost 3000 points. The number of move operations decreased,

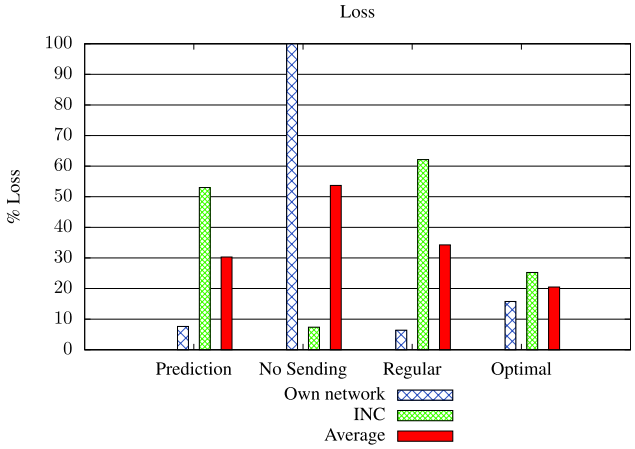



**FIGURE 19.** Packet loss of own network, INC and average of both in an CSMA AP scenario.

meaning better slots were found at first instance, considerably reducing the number of suboptimal decisions. Note that MCS and gain control adapters are still enabled in the system and try to optimise the packet success ratio given the channel conditions. This fact explains why the packet success ratio is almost the same for both the EWMA algorithm and prediction algorithm.

## VII. ALGORITHM HARDWARE REQUIREMENTS

Within the context of the SC2 competition, full-blown servers were used to execute the algorithms. If we want to deploy the proposed algorithm on real devices, nodes need to be equipped with enough memory. They need to have memory available to store the model and the replay buffer. The number of elements of the replay buffer is defined as $2MHSC$. In our example, we used 32-bit floats (4 bytes), a replay buffer size ($M$) of 20,000 (as defined in Table 2), $S$ and $C$ are 20 and 4 respectively, and a history ($H$) of 50 superframes. This would mean that we used 610MiB of memory for the replay buffer. Note that we can quickly reduce the size if we store fewer elements or reduce the history. In addition, for the NN proposed in this work, the number of neurons to store in the network is defined in Equation 17.

$$\frac{29}{2}S^2C^2H^2 + 23SCH \tag{17}$$

For our example, the network needs 885MiB of memory. Note that this can be reduced by decreasing the size of the NN. Alternatively, a Convolutional Neural Network (CNN) capable of producing useful predictions may be more memory-efficient.

The computation effort needed to run the proposed algorithm could be offloaded to AI-enabled chipsets or hardware acceleration (like GPUs or FPGAs). These chipsets are quickly dropping in cost and are becoming commonplace in smartphones.

During the DARPA SC2, the nodes were built to have more than enough resources available. The constraints to deploy the

algorithm on real devices was out of the scope of this work and will be provided in future work.

## VIII. CONCLUSION

In this paper, we presented a multi-agent spectrum prediction approach to optimise the wireless spectrum for a multichannel slotted wireless network and a set of other (unknown) networks. By using a DNN, we are able to predict the upcoming superframe. Based on this information a (receiver-side) scheduler can avoid collisions with other neighbouring networks. These neighbouring networks could use different and unknown technologies. The DNN is trained in an online way, using techniques of both RL and supervised learning. Because of the continuous action space and the multi-agent environment, we model the problem as a partially observable stochastic game with continuous action space. As deriving a policy with RL in this case is unfeasible, we create partially observable labels to optimise a prediction by using online supervised learning.

We showed that by using a prediction approach, in simulation we were able to reduce the number of inter-network collisions by 30% in comparison to commonly used schedulers. We showed that we increased the overall throughput in a variety of topologies and settings. Additionally, the algorithm was implemented in the SCATTER radio built to participate in the SC2 competition. We showed that, in comparison with a EWMA slot selection algorithm, our proposed algorithm increased our own throughput from 20Mb/sec to 22Mb/sec and increased the score of the game, which indicates higher throughput by the other teams as well.

## REFERENCES

[1] 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, no. 13, pp. 2127–2159, Sep. 2006.

[2] Forecast, Global Mobile Data Traffic, "Cisco visual networking index: Global mobile data traffic forecast update, 2018–2023," Cisco Syst., San Jose, CA, USA, White Paper, 2020.

[3] C. Cormio and K. R. Chowdhury, "A survey on MAC protocols for cognitive radio networks," *Ad Hoc Netw.*, vol. 7, no. 7, pp. 1315–1329, Sep. 2009.

[4] D. Bloembergen, K. Tuyls, D. Hennes, and M. Kaisers, "Evolutionary dynamics of multi-agent learning: A survey," *J. Artif. Intell. Res.*, vol. 53, pp. 659–697, Aug. 2015.

[5] S. D. Giannoulis, C. Donato, R. Mennes, F. A. P. de Figueiredo, I. Jabandzic, Y. De Bock, M. Camelo, J. Struye, P. Maddala, M. Mehari, A. Shahid, D. Stojadinovic, M. Claeys, F. Mahfoudhi, W. Liu, I. Seskar, S. Latré, and I. Moerman, "Dynamic and collaborative spectrum sharing: The SCATTER approach," in *Proc. IEEE Int. Symp. Dyn. Spectr. Access Netw. (DySPAN)*, Newark, NJ, USA, Nov. 2019, pp. 1–6.

[6] P. Tilghman, "Will rule the airwaves: A DARPA grand challenge seeks autonomous radios to manage the wireless spectrum," *IEEE Spectr.*, vol. 56, no. 6, pp. 28–33, Jun. 2019.

[7] *Darpa Spectrum Collaboration Challenge*. Accessed: Oct. 23, 2019. [Online]. Available: https://spectrumcollaborationchallenge.com/

[8] F. F. Kuo, "The aloha system," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 4, no. 1, pp. 5–8, 1974.

[9] S.-Y. Yun, Y. Yi, J. Shin, and D. Y. Eun, "Optimal CSMA: A survey," in *Proc. IEEE Int. Conf. Commun. Syst. (ICCS)*, Nov. 2012, pp. 199–204.

[10] C. Cano and D. J. Leith, "Coexistence of WiFi and LTE in unlicensed bands: A proportional fair allocation scheme," in *Proc. IEEE Int. Conf. Commun. Workshop (ICCW)*, Jun. 2015, pp. 2288–2293.

[11] A. M. Cavalcante, E. Almeida, R. D. Vieira, S. Choudhury, E. Tuomaala, K. Doppler, F. Chaves, R. C. D. Paiva, and F. Abinader, "Performance evaluation of LTE and Wi-Fi coexistence in unlicensed bands," in *Proc. IEEE 77th Veh. Technol. Conf. (VTC Spring)*, Jun. 2013, pp. 1–6.

[12] V. Maglogiannis, D. Naudts, A. Shahid, S. Giannoulis, E. Laermans, and I. Moerman, "Cooperation techniques between LTE in unlicensed spectrum and Wi-Fi towards fair spectral efficiency," *Sensors*, vol. 17, no. 9, p. 1994, Aug. 2017.

[13] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC protocols for wireless sensor networks: A survey," *IEEE Commun. Mag.*, vol. 44, no. 4, pp. 115–121, Apr. 2006.

[14] M. Hadded, P. Muhlethaler, A. Laouiti, R. Zagrouba, and L. A. Saidane, "TDMA-based MAC protocols for vehicular ad hoc networks: A survey, qualitative analysis, and open research issues," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2461–2492, 4th Quart., 2015.

[15] D. Dujovne, T. Watteyne, X. Vilajosana, and P. Thubert, "6TiSCH: Deterministic IP-enabled industrial Internet (of Things)," *IEEE Commun. Mag.*, vol. 52, no. 12, pp. 36–41, Dec. 2014.

[16] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, "A survey on spectrum management in cognitive radio networks," *IEEE Commun. Mag.*, vol. 46, no. 4, pp. 40–48, Apr. 2008.

[17] Y.-C. Liang, K.-C. Chen, G. Y. Li, and P. Mahonen, "Cognitive radio networking and communications: An overview," *IEEE Trans. Veh. Technol.*, vol. 60, no. 7, pp. 3386–3407, Sep. 2011.

[18] E. Stevens-Navarro, Y. Lin, and V. W. S. Wong, "An MDP-based vertical handoff decision algorithm for heterogeneous wireless networks," *IEEE Trans. Veh. Technol.*, vol. 57, no. 2, pp. 1243–1254, Mar. 2008.

[19] P. Sakulkar and B. Krishnamachari, "Online learning of power allocation policies in energy harvesting communications," in *Proc. Int. Conf. Signal Process. Commun. (SPCOM)*, Jun. 2016, pp. 1–5.

[20] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, vol. 2, no. 4. Cambridge, MA, USA: MIT Press, 1998.

[21] A. Gosavi, "Reinforcement learning: A tutorial survey and recent advances," *Informs J. Comput.*, vol. 21, no. 2, pp. 178–192, May 2009.

[22] Y. Shoham, R. Powers, and T. Grenager, "Multi-agent reinforcement learning: A critical survey," *Web Manuscript*, 2003.

[23] Y. Chu, P. D. Mitchell, and D. Grace, "ALOHA and Q-learning based medium access control for wireless sensor networks," in *Proc. Int. Symp. Wireless Commun. Syst. (ISWCS)*, Aug. 2012, pp. 511–515.

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

[25] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, and G. Ostrovski, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[26] Z. Chen, N. Guo, and R. C. Qiu, "Demonstration of real-time spectrum sensing for cognitive radio," in *Proc. Mil. Commun. Conf. (MILCOM)*, 2010, pp. 323–328.

[27] F. A. P. de Figueiredo, D. Stojadinovic, P. Maddala, R. Mennes, I. Jabandžić, X. Jiao, and I. Moerman, "Scatter phy: A physical layer for the DARPA spectrum collaboration challenge," in *Proc. IEEE Int. Symp. Dyn. Spectr. Access Netw. (DySPAN)*, Newark, NY, USA, Nov. 2019, pp. 1–6.

[28] F. A. De Figueiredo, X. Jiao, W. Liu, R. Mennes, I. Jabandžić, and I. Moerman, "A spectrum sharing framework for intelligent next generation wireless networks," *IEEE Access*, vol. 6, pp. 60704–60735, 2018.

[29] R. Mennes, M. Claeys, F. A. P. De Figueiredo, I. Jabandžić, I. Moerman, and S. Latré, "Deep learning-based spectrum prediction collision avoidance for hybrid wireless environments," *IEEE Access*, vol. 7, pp. 45818–45830, 2019.

[30] P. Ramachandran, B. Zoph, and Q. V. Le, "Swish: A self-gated activation function," 2017, *arXiv:1710.05941*. https://arxiv.org/abs/1710.05941

[31] M. R. Palattella, T. Watteyne, Q. Wang, K. Muraoka, N. Accettura, D. Dujovne, L. A. Grieco, and T. Engel, "On-the-fly bandwidth reservation for 6TiSCH wireless industrial networks," *IEEE Sensors J.*, vol. 16, no. 2, pp. 550–560, Jan. 2016.

**RUBEN MENNES** (Student Member, IEEE) received the master's degree in computer science from the University of Antwerp, Belgium, in June 2016. In August 2016, he joined the Department of Mathematics and Computer Science, University of Antwerp - imec, where he is active as a Ph.D. Student. The focus of his research is mainly on collaborative wireless networks and smart wireless networks, the cross-domain between wireless network technologies, and machine learning. He was part of the SCATTER team of the DARPA SC2 competition, where he focused on the structure, design, implementation and research of the decision making and optimisation engine.

**FELIPE A. P. DE FIGUEIREDO** (Student Member, IEEE) received the B.S. and M.S. degrees in telecommunications from the Instituto Nacional de Telecomunicações (INATEL), Minas Gerais, Brazil, in 2004 and 2011, respectively. He is currently pursuing the Ph.D. degree with the Internet Technology and Data Science Laboratory part of the Department of Information Technology, Ghent University - imec, Gent, Belgium. He has been working with Research and Development of telecommunications systems for more than ten years. His research interests include digital signal processing, digital communications, mobile communications, MIMO, multicarrier modulations, and FPGA development.

**STEVEN LATRÉ** (Member, IEEE) received the Master of Science degree in computer science, the Ph.D. degree in computer science engineering, and the Ph.D. degree in autonomic network management from Ghent University, Belgium, in 2006 and 2011, respectively. He is currently an Associate Professor with the University of Antwerp - imec, Belgium. He is leading the IDLab research group, University of Antwerp, consisting of more than 80 researchers. He is author or coauthor of more than 100 articles published in international journals or in the proceedings of international conferences. His personal research expertise focuses on deep reinforcement learning and wireless network management. He is a member of the Young Academy of Belgium. He was a recipient of the IEEE COMSOC Award for Best Ph.D. in network and service management, in 2012, the IEEE NOMS Young Professional Award, in 2014, and the IEEE COMSOC Young Professional Award, in 2015.