Coexistence of Shared-Spectrum Radio Systems through Medium Access Pattern Learning using Artificial Neural Networks

Sebastian Lindner, Leonard Fisser, Andreas Timm-Giel
Institute of Communication Networks (ComNets)
Hamburg University of Technology (TUHH)
Hamburg, Germany
{sebastian.lindner, leonard.fisser, timm-giel}@tuhh.de

***Abstract*—Spectrum scarcity requires novel approaches for sharing frequency resources between different radio systems. Where coordination is not possible, intelligent approaches are needed, allowing a novel ”secondary” system to access unused resources of a legacy (primary) system without requiring modifications of this primary system. Machine Learning is a promising approach to recognize patterns of the primary system and adapt the channel access accordingly. In this contribution we investigate the capability of Feed-Forward Deep Learning and Long Short Term Memory (LSTM) Recurrent Neural Networks (RNNs) to detect communication patterns of the primary user. Therefore we take the example of a new aeronautical system (LDACS) coexisting with three different systems. Firstly the coexistence with the Distance Measurement Equipment (DME) providing a deterministic interference to the secondary user and secondly with two synthetic channel access patterns, realized by a 2-state Markov model, modeling a bursty channel access behavior, as well as through a sequential channel access model.**

It can be shown that the Markov property of a Gilbert-Elliot channel model limits the predictability; nonetheless, we show that the model characteristics can be fully learned, which could leverage the design of interference avoidance systems that make use of this knowledge. The determinism of DME allows an error-free prediction, and it is shown that the reliability of sequential access model prediction depends on the model's parameter. The limits of Feed-Forward Deep Neural Networks are highlighted, and why LSTM RNNs are state-of-the-art models in this problem domain. We show that these models are capable of online learning, as well as of learning correlations over long periods of time.

In the spirit of open science, the implementation files are made available in the conclusion.

***Index Terms*—dynamic spectrum access, reliability, neural networks, machine learning**

I. Introduction

Frequency spectrum is a finite resource, and spectrum scarcity has been a known problem, certainly aggravated by an increasing demand for wireless communication systems. To overcome the scarcity problem, dynamic spectrum sharing has been proposed, where radio nodes employing different technologies intelligently access the spectrum so that little to no interference is caused for the other system. This can be achieved in a mutually-coordinated fashion – for example, 802.11 networks may coexist with LTE Licensed Assisted Access (LTE-LAA) networks, and the two protocol stacks are adapted to achieve fairness while coexisting, as investigated in [1].

Our focus is instead on an uncoordinated coexistence, where a novel system shall operate on a frequency band that is already being utilized by a licensed user. A number of scenarios come to mind where a frequency band has historically been assigned to a use case, but which is (regionally) not fully utilized, or which has seen declined usage over the years as the technology has gone out of fashion. For the latter case, TV whitespaces have been a prime focus of Cognitive Radio (CR) research, which aim to utilize these idle frequency bands as TV is moving from analog to digital, as motivated in [2]. For the prior case, the Distance Measuring Equipment (DME) system for aeronautical communication is of interest. This system from the 1950's is still in use, but the licensing of parts of the radio spectrum for its exclusive use is limiting the development of novel aeronautical communication systems, such as the L-band Digital Aeronautical Communications System (LDACS) that aims to be a digital communication system within the envisioned future infrastructure for aviation. Parts of it are being standardized by the International Civil Aviation Organization (ICAO) at the time of writing, while other parts are under active development. LDACS shall operate in the aeronautical L-band, where DME also operates. To prevent interference on the legacy DME system, the use of DME frequency channels may be regionally prohibited, as discussed in [3].

However, such a general channel blacklisting is costly in terms of efficiency. Therefore it is the main goal of this paper to show that with modern techniques from Machine Learning (ML), it is feasible to identify the *gaps* that the DME system's access on the radio spectrum leaves, which can be utilized for other forms of communication, such as through LDACS. The approaches are general enough to be applied on any shared-spectrum use case.

All of these scenarios can be generalized to the concept of CR networks. In this field, CRs are “cognitive” in the sense that they are capable of learning and adapting to their environment. In a heterogeneous network of both CR and non-CR users, the non-CR users are typically referred to as Primary Users (PUs), while the CR users are Secondary Users (SUs). The SUs may not be able to influence PU behavior, but instead coexist with – and potentially cause interference to – PUs and use the channel when they do not interfere.

When the PU system cannot be changed to coordinate its medium access with the SU, SUs are forced to learn the access patterns of the PUs, and then try to utilize those times for transmission that are predicted to be idle. Such a learning of channel access patterns can be referred to as dynamic spectrum access, and this contribution gives an overview on Artificial Neural Network (ANN)-based approaches to this problem. A focus is put on ANNs, in particular on feed-forward Deep Neural Networks (DNNs), as well as Long Short Term Memorys (LSTMs) networks, a variant of Recurrent Neural Networks (RNNs). An introduction into the workings of these ANNs is given. As a common pitfall of ML methods is its variability, a highlight on reliability analysis is attempted throughout this paper, as well as peeking into what is often referred to as ML's *black box*. Two prominent ANN architectures are studied in this regard, and evaluated on three different channel access models.

The paper is organized as follows. Section II presents related work in a chronological order. Section III defines our system model, and the models that characterize the medium access strategy of the PUs. Section IV investigates feed-forward ANNs and their prediction performance on a Markov channel access model as well as on a sequential channel access model. Section V shows how RNNs are preferable to feed-forward ANNs in the problem at hand, and showcases their performance bounds for the sequential channel access model in as well as the DME channel access model. Section VI concludes the document and gives an outlook into some of the many directions of future research in this area.

II. Related work

In [4] from 2006, a first approach using cyclostationary detection is investigated. The PU channel access is required to be periodic and works optimally for Time-Division Multiple Access (TDMA)-based protocols as these inherently introduce periodicity. Most cellular and WiFi systems are not natively periodic, however it is argued that some periodicity is introduced through higher-layer protocols such as TCP, or from the application layer. The learning technique's task is therefore to reliably identify this "hidden periodicity".

The PU channel access is modelled as a cyclostationary random process R which is stationary over some period τ: $\mathrm{R}(t) = \mathrm{R}(t+\tau)$. Channel access patterns can be described by a Bernoulli random variable $C(t)$.

$$C(t) = \begin{cases} 1 \text{ if channel occupied at time } t, \\ 0 \text{ else.} \end{cases}$$

$C(t)$ can be observed over time to estimate its distribution, and $P(C(t) = 1)$ can be computed, providing the probability of the channel being busy at time index $t \bmod \tau$ when $C(t)$ is discrete.

With this, the optimal SU transmission time for a transmission of l time units can be calculated. The authors show through simulation that the SUs can fully utilize all available bandwidth and cause no interference on PUs when these use a TDMA-based channel access protocol. Performance decreases substantially when Carrier Sense Multiple Access (CSMA) is used by PUs.

In [5] the authors identify two types of traffic patterns that exist in wireless environments: 1) deterministic `on`/`off` patterns, and 2) stochastic patterns that can be described in statistical terms. It becomes the goal of SUs to "minimize interference on PUs by predicting the future idle times and by changing to better channels before the PU appears on the currently used channel." It is acknowledged that a general model is needed that works on any type of traffic pattern. In their system model, a channel prediction technique follows these steps: 1) Gather each channel's access information. 2) Classify the channels' access patterns as "deterministic" or "stochastic". 3) Predict each channels' availability time on each channel, depending on the pattern type. 4) Switch to the most promising channel. The input to the classification is a discrete binary sequence x_n, $n = 1, \ldots, N$ representing whether the channel was idle $x_n = \texttt{off}$ or in use $x_n = \texttt{on}$. Due to suboptimal sensing resolution a lag m is introduced to x_n, which shows a different sequence than what is present in reality. The autocorrelation of the signal with the observed, delayed copy of itself can be computed as $R_{xx}(m) = \sum_{n=0}^{N-m-1} x_n x_{n+m}$. This autocorrelation is found over periods whose length τ correspond to one `on` and one `off` time present during the period. When the separation of the time domain into periods of length τ yields such periods with one `on` and one `off` occurrences relatively constantly, then the traffic is categorized as periodic. For periodic traffic, determining the starting point of the next `off` period is just $T_s = t + T_{\texttt{on}}$, where t corresponds to the starting time of the next period with length τ. And the length of the usable period is $T_l = T_{\texttt{off}} = \tau - T_{\texttt{on}}$.

For stochastic access patterns, when T_s and T_l stem from a probability distribution, then their distribution is the learning target. Having collected the discrete access pattern x_n, the probability that the channel is idle for at least y units of time follows through $P(T_l \geq y) = \frac{|x_k^{\texttt{off}}|}{x_k^*}$ where $x_k^{\texttt{off}}$ refers to that subset of x_n that corresponds to sequences of length k where $x = \texttt{off}$ and x_k^* to the subset of x_n that contains sequences of length k. With this and an a-priori interference tolerance α, a SU could transmit for z units of time where z is found through $P(t \leq z) = 1 - \alpha$; this corresponds to a guarantee of $\alpha\,\%$ not to interfere with a PU.

The majority of work on this problem focuses on ML with its strong suit in time series prediction and pattern learning. We found an early work in this direction in [6] from 2010. There, the channel status of channel i is modeled over T discrete slots with binary values as in Eq. 1.

$$\forall i = 1, \ldots, c : x_i^T = \left[x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(T)} \right], \text{ where}$$
$$\forall j = 1, \ldots, T : x_i^{(j)} = \begin{cases} 0 \text{ if channel } i \text{ sensed idle in slot } j \\ 1 \text{ else} \end{cases} \tag{1}$$

Based on the slot status history, a Multi-Layer Perceptron (MLP) model is trained to predict the status of the next slot for one channel. For a multi-channel system, a predictor is assigned to each channel. Offline training is treated as an advantage over Hidden Markov Model (HMM) models that learn continuously as the computational complexity is focused on this singular training process, and predictions from a trained model are obtained with far less computational effort.

The authors of [7] criticize the popular assumption in CR research that PU activity can be modeled with independent frequency channels. Their example application includes a low-power Wireless Sensor Network (WSN) running IEEE 802.15.4 on the industrial, scientific and medical (ISM) band, which is shared by various wireless technologies. IEEE 802.15.4 foresees multiple frequency channels, which the PUs interfere with in a correlated fashion. Therefore a Gilbert-Elliot model – essentially a Markov chain with *good* and *bad* channel states – is used to model these correlated channels, and in the manner of Reinforcement Learning (RL), a Deep Q-Network (DQN) is employed, so that the RL agent indirectly learns the channel model so that it can maximize its successful transmissions.

Similarly the work in [8] considers a multi-agent framework and compares an actor-critic RL approach to a DQN, random access and the optimal policy. Users can select channels to learn whether they are idle or utilized, and so a partially observable Markov decision process formulates their access strategy. When the channels are independent, a restless multi-armed bandit process is sufficient. The problem is then formulated in a single- and multi-user scenario and the different solutions are compared.

We focus on the earlier-described works, on the "predictability" of different medium access patterns with ANNs, and on the analysis of performance bounds of ANN-based predictors.

III. System Model

We consider a set $\mathcal{C}$ of c orthogonal frequency channels, each with identical bandwidth B. The system is viewed from the perspective of one SU which tries to find a medium access strategy that does not interfere with the PUs. The number of PUs is not known, but the superposition of their accesses on the c frequency channels can be observed through channel sensing. Time is divided into timeslots, and it is assumed that the SU is capable of sensing all c frequency channels during one timeslot.

The sampling of medium accesses by a radio constitutes a time series, and the forecasting of this is thus a form of time series prediction. The channel status history over T timeslots and c channels is modeled as in Equation 1: for $c = 1$ channel this is a binary vector, for $c > 1$ it becomes a matrix. This is a classical problem for supervised learning – as long as there is a pattern inside this series, it can be learned, and ANNs have proven capable of this task. Two main concerns arise: 1) what is the performance bound, or level of reliability, and 2) in many communication networks, especially mobile ones, the environment changes quickly: users join, leave and move; the learning technique must adapt to changes and be applicable in realistic scenarios.

A. Markov Channel Access Model

The paper in [6] considers a Markov chain channel access model for a single channel. This corresponds to a discrete-time Markov chain with two states: idle and busy, as in Fig. 1.

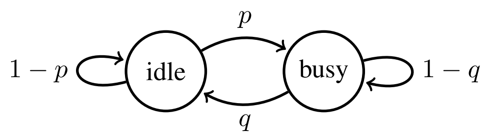


Fig. 1: Markov chain for the medium access pattern of PUs and a single frequency channel.

Let $X \sim \text{Geo}(p)$ be a geometrically distributed random variable. The geometric distribution models the discrete number of timeslots that the channel is *idle* for, and in the same way $Y \sim \text{Geo}(q)$ models the number of timeslots the channel is busy for. The expected numbers of timeslots that the channel is idle or busy for follow as $E[X] = \frac{1}{p}$, $E[Y] = \frac{1}{q}$. "Bursty" traffic is thus modeled. Given $E[X]$, $E[Y]$ for idle and busy times in timeslots, the model's utilization follows as

$$\rho = \frac{E[Y]}{E[Y] + E[X]} = \frac{\text{busy time}}{\text{total time}} \tag{2}$$

Equation 2 follows from the stationary distribution of the two-state Markov chain in Figure 1. Let P be the transition matrix:

$$P = \begin{bmatrix} 1-p & p \\ q & 1-q \end{bmatrix}$$

then the stationary distribution $\pi = [\pi_{\text{idle}}\ \pi_{\text{busy}}]$ follows through Equation 3.

$$\begin{aligned} &\pi \cdot P = \pi \\ &\pi \cdot \mathbf{1} = 1 \\ &\Rightarrow \pi = \begin{bmatrix} \frac{q}{p+q} & \frac{p}{p+q} \end{bmatrix} \end{aligned} \tag{3}$$

Utilization is identical to the stationary busy state distribution as shown in Equation 4.

$$\rho = \frac{E[Y]}{E[Y] + E[X]} = \frac{\frac{1}{q}}{\frac{1}{p} + \frac{1}{q}} = \frac{p}{q+p} = \pi_{\text{busy}} \tag{4}$$

And thus $1 - \rho = \pi_{\text{idle}}$.

B. Sequential Channel Access Model

Let $\mathcal{C}$ be a multitude of $c \overset{e.g.}{=} 16$ frequency channels (as in [7]), then this model foresees a single channel $c_i \in \mathcal{C}$ idle in each time slot. With a probability p, this idle channel moves to the next position $c_i \rightarrow c_{i+1}$. This corresponds to the Markov chain in Fig. 2 with c states. Visiting a state i corresponds to channel i being idle, and all others being busy. Observing the system in this state leads to the observation at time t of the binary vector $\left[x^t_{j<i} = 1,\, x^t_i = 0,\, x^t_{j>i} = 1\right]$.

A basic assumption is that a SU may *transmit* on a particular frequency channel and determine the result of the transmission through some acknowledgement (ACK) / negative acknowledgment (NACK) protocol, and that it may *sense* a frequency channel, which reveals whether it is idle or utilized. An observation over time is exemplified in Fig. 3.

C. Aeronautical L-Band Model

A real-world application was identified during work on the LDACS, which operates on the aeronautical L-band in the range of 1 – 2 GHz. A number of systems operate in the same band. Classical communication system design

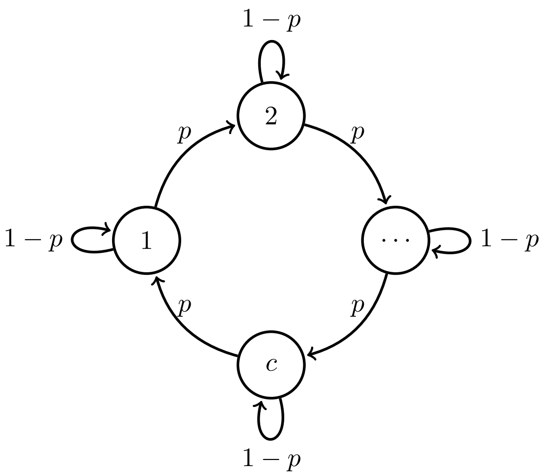


Fig. 2: Markov chain for the sequential medium access pattern of PUs and n frequency channels.

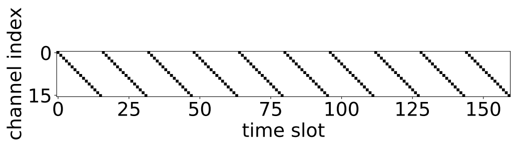


Fig. 3: Exemplary observation of $c = 16$ frequency channels over time, according to a sequential channel access model with switching probability $p = 1$.

dictates exclusive use in a particular frequency channel: a new system will not be allowed to operate if there is a chance that it disturbs legacy systems. However, this type of exclusiveness leads to the spectrum scarcity problem as well as an ineffective utilization of the spectrum.

The dominant legacy system is the DME system. It allows an aircraft to ping a DME ground station on a certain *interrogation frequency*, and to listen for a response on a different *response frequency*, to obtain its geographic position. Both are $1\,\mathrm{MHz}$ wide channels, and the frequency of requests varies within 6-33 ms. The medium access follows a deterministic pattern as shown in Figure 4. It has been modeled in [3] and found as "the main L-band interference source" in [9].

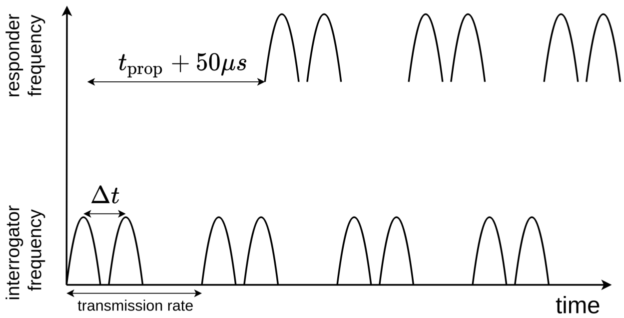


Fig. 4: Depiction of the medium access of the DME system's ping-response mechanism, where t_{prop} is the propagation delay.

IV. Feed-Forward Deep Neural Networks

One of the earliest accomplishments on the problem of PU channel access prediction was made in [6]. The authors constructed a Multi-Layer Perceptron model; a feed-forward DNN in the terminology of the widely popular book in [10]. The term "feed-forward" stems from the data input propagating only from the first to the last layer, i.e. past data inputs have no effect on the current input.

A high-level description of the learning process of a DNN is:

1) An input sequence x is fed into the DNN.
2) The first layer's hypothesis h_{Θ_1} is calculated through its activation function, where Θ_1 denotes the vector of layer weights. A common choice is the Sigmoid function in Equation 5, but other non-linear function can be used instead.
3) The first layer's output is used as input of the second layer, so that successive hypotheses $h_{\Theta_2}(h_{\Theta_1}(x))$ are obtained, finally yielding $h_\Theta(x)$.
4) A *loss function* J – i.e. the Mean Squared Error in Eq. 6 – compares the output to the target label.
5) An *optimizer* adjusts the network weights Θ s.t. $J \rightarrow$ min by computing the current gradient of the loss function J with respect to the current network weights Θ, and updating the weights in the learning rate α-weighted opposite direction; this is called Gradient Descent and is summarized in Eq. 7.

$$h_\Theta(x) = g(\Theta^{\mathrm{T}} x) = \frac{1}{1 + e^{-\Theta^{\mathrm{T}} x}} \tag{5}$$

$$J_{\text{MSE}} = \frac{1}{n} \sum_{i=1}^{n} (Y_i - Y_i')^2 \tag{6}$$

$$\Theta_j = \Theta_j - \alpha \frac{\delta}{\delta \Theta_j} J(\Theta) \tag{7}$$

The network architecture foresees a binary input vector x as in Eq. 1, but for a single channel only. It is fully connected to two hidden layers of 15 and 20 neurons respectively, the *Mean Squared Error* is used as the loss function, and the *Adam optimizer* is responsible for weight updates. The architecture is shown in Figure 5.

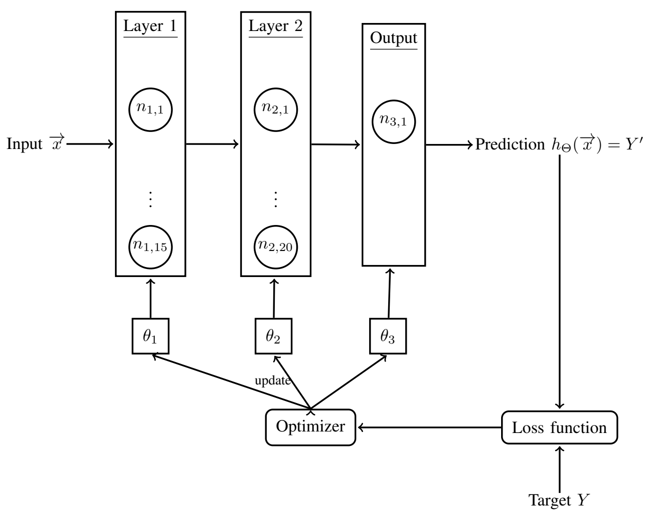


Fig. 5: The MLP network architecture. The input layer is implicit, and only the two hidden layers are shown.

A. Applied on the Markov channel access model

To replicate original results, the Markov channel access model from Section III-A is employed. Idle and busy times are independent of each other, and they themselves follow memoryless Geometric distributions. The resulting Markov chain likewise has the Markov property; and so this is in an interesting choice for a channel access model as there is no pattern to learn, since by definition, only the present state has any effect on future outcomes, knowing all history brings no benefit. Therefore a predictor cannot learn a pattern over the time, and it must fall back to learning the distributions themselves. The hypothesis thus becomes: can a feed-forward DNN accomplish this and learn the characteristics of a Markov channel access model – which essentially is what the authors of [4] and [5] proposed.

An investigation is done through configuring the channel access model with $E[X] = 5$, $E[Y] = 10 \Rightarrow \rho = 0.66$, observing it for 2500 timeslots and feeding these observations into the MLP; once with an input vector of length $T = 1$ and once for $T = 4$, as suggested by the original authors. The accuracy *during training* is shown in Figure 6, where each sample is one input into the MLP, so for $T = 4$ this corresponds to four timeslots – i.e. in both simulations the MLP observed the same number of timeslots.

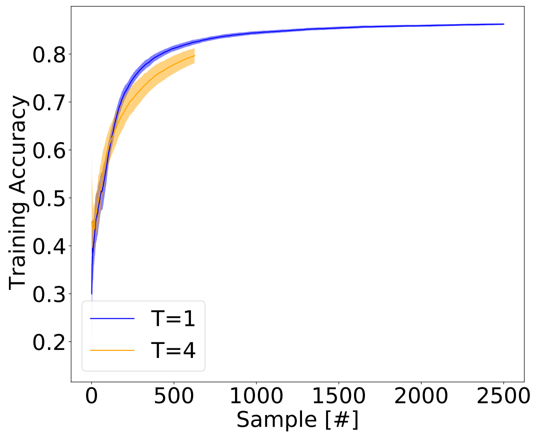


Fig. 6: Multi-Layer Perceptron parametrized as in [6] but with modern Adam optimizer ([11]) applied on a Markov channel access model with utilization $\rho = 0.66$, average number of idle slots $E[X] = 5$ and busy slots $E[Y] = 10$ over 2500 observed timeslots. For both $T = 1$ and $T = 4$ in total 2500 timeslots are considered. 95 %-confidence intervals are obtained through the batch means method.

Fig. 6 shows that using a *single* timeslot as input results in a *better* accuracy than aggregating $T = 4$ timeslots into one sample. However, the original authors chose $T = 4$ over $T = 1$. To investigate, consider the *validation accuracy* (accuracy on *new, unseen* data *after* training) in Figure 7, where the Markov channel model is once parametrized for *short periods* with $E[X] = 2$, $E[Y] = 4$, and once for *long periods* with $E[X] = 23, E[Y] = 46$. A larger sample length T results in worse results. And accuracy on *long* periods is considerably better than on short periods – this coincides with the original authors' findings: there, too, a larger utilization ρ led to better MLP performance. In fact, validation accuracy was found to approximate ρ for simulations on an increasing utilization with $E[X] = 2, E[Y] \in \{4, 8, \dots, 40\}$.

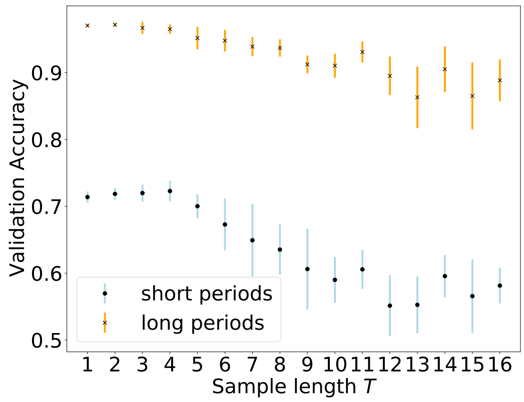


Fig. 7: DNN models with varying input sequence lengths T pitted against a Markov channel model with $E[X] = 2, E[Y] = 4$ (short periods) and with $E[X] = 23, E[Y] = 46$ (long periods).

If the hypothesis were true, then the predictions should converge to the channel model's transition probabilities; for example:

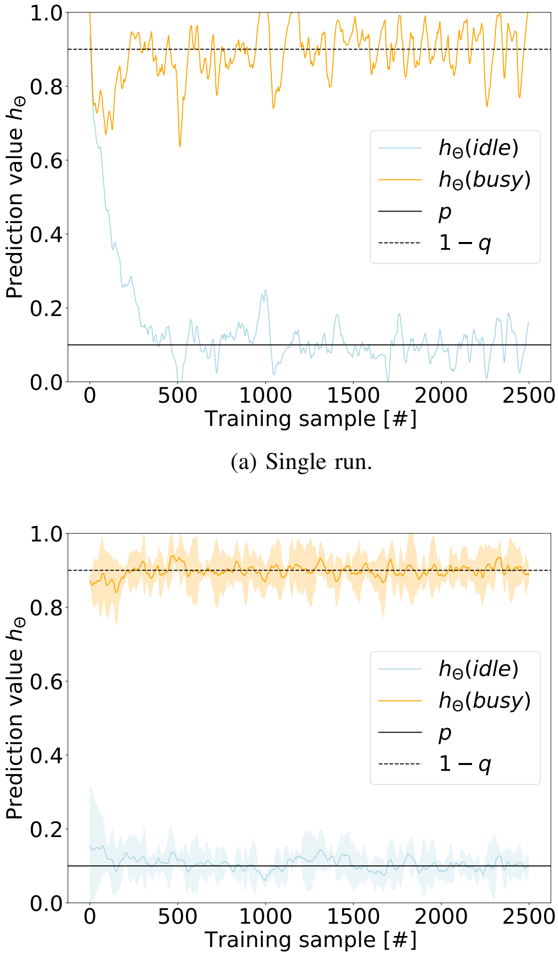
$$p \overset{e.g.}{=} 0.1 \Rightarrow E[X] = \frac{1}{p} = 10$$
$$q \overset{e.g.}{=} 0.1 \Rightarrow E[Y] = \frac{1}{q} = 10$$
$$\pi = \begin{bmatrix} \pi_X & \pi_Y \end{bmatrix} \overset{Eqs.\ 3,4}{=} \begin{bmatrix} 1-\rho & \rho \end{bmatrix} \overset{e.g.}{=} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

On this exemplary Markov channel access model, Fig. 8 depicts a single simulation's evolution of predictions over time on the input of an idle channel ($x = 0$) and on the input of a busy channel ($x = 1$) in Fig. 8a, and averaged over 12 simulations in Fig. 8b. The two prediction values $h_\Theta(x = 0)$, $h_\Theta(x = 1)$ converge to approximate p and $1 - q$ respectively, allowing some variance. The prediction $h_\Theta(x = 0)$ gives the probability, given that the channel is currently idle, it will be busy next. This only happens if the idle state is left, and the busy state entered – with probability p (see Fig. 1). Likewise, for $h_\Theta(x = 1)$ the channel is currently busy, so it remaining busy occurs only if the busy state is *not* left with probability $1 - q$. This is a very interesting result, as knowing p, q fully describes the Markov chain, allowing the calculations of expectation values as well as the stationary distribution.

Finding what the predictions converge towards *does not* explain why a larger utilization ρ leads to a *better* prediction accuracy. To explain, consider what the used binary accuracy is defined as:

$$\begin{aligned} &\text{Let } y^{(\text{pred})} = h_\Theta(\overrightarrow{x}) \in \mathbb{R} \text{ (the predicted value)}, \\ &\quad y^{(\text{true})} \in \{0, 1\} \text{ (the target label)}, \\ \Rightarrow\ \text{accuracy} &= \begin{cases} 1 \text{ if } y^{(\text{true})} = \lfloor y^{(\text{pred})} \rceil \\ 0 \text{ else} \end{cases} \end{aligned} \tag{8}$$

Prediction values are rounded to the nearest integer, and the average over each sample's accuracy gives the final value. Therefore 0.5 is the threshold value at which decisions



(a) Single run.

(b) Averaged over 12 repetitions.

Fig. 8: DNN model prediction values during training on a Markov channel access model with $\rho = 2/3$. $h_\Theta(x)$ stands for the hypothesis of the neural network with neuronal weights Θ on the input sequence x, so the evolution of the hypotheses over the training process is shown here.

are flipped. It follows that the prediction will forecast the channel to be *in the same state* as it currently is in:

$$p < 0.5,\ x = 0: \ h_\Theta(x) \approx p \Rightarrow \lfloor h_\Theta(x) \rceil = 0 = x$$
$$q < 0.5,\ x = 1: \ h_\Theta(x) \approx 1 - q \Rightarrow \lfloor h_\Theta(x) \rceil = 1 = x$$

and when utilization ρ increases, more often a busy timeslot will follow a busy timeslot, therefore this prediction will be correct more often.

As for sample lengths $T > 1$ of the binary input vector, the *input space* increases as 2^T. The MLP must thus map exponentially more inputs to the learned outputs, and so a decreasing performance for a larger T on the same number of considered timeslots is explained in Fig. 7. As there is no pattern to learn, no benefit can be drawn from aggregating multiple timeslots into one input sample.

In conclusion, the MLP model *is capable* of learning Markov chain characteristics through its transition probabilities p, q, which is sufficient to compute expectation values and the steady state distribution. We could show that prediction values for idle and busy observations converge to p and $1 - q$ respectively. However, realistically, the rounded predictions realize a persistent predictor: given input x, output $y = x$. This comes from the choice of the channel access model: the memoryless property of the Markov chain gives the MLP no pattern it can learn. The capability of learning the transition probabilities and with this the expectations and steady state distribution of this process is nonetheless interesting, and could be used for a more sophisticated interference avoidance concept that makes use of fully knowing the channel access model.

B. Applied on the Sequential Access Model

The applicability of this model to the sequential access model is evaluated in this section. A first feasibility test on the simplest sequential channel access model – of $c = 2$ frequency channels and a switching probability $p \in \{0.5, 0.6, \ldots, 1\}$, observations of *one* channel are made – shows validation accuracy is identical to p for $T = 1$. For every percentage point that the switching probability drifts from 1 as $p \to 0.5$, this difference $1 - p$ is *noise*. As the observations get noisier, the accuracy decreases and its variance increases.

Realistically, a SU may have a multitude of $c > 2$ frequency channels to choose from; for example, for $c = 3$ the observations on the first channel would be $x_1^{(j)} = [1, 0, 0, 1, 0, 0, \ldots]$: given input $x = 1$, surely the next timeslot the channel will be idle, but given $x = 0$, what can be a sensible prediction? Half of the time an idle timeslot, and half of the time a busy timeslot will follow, and so $h_\Theta(x = 0) \approx 0.5$. One solution is to increase T s.t. the input space $\{2^T\}$ is large enough to uniquely encode all transitions of the channel access model. As soon as $T \geq c - 1$, the input sequence can encode all transitions; for example see Equation 9, where a $\lightning$ symbol highlights those cases where no clear successor channel state can be learned, and the prediction instead drifts towards an uncertain 0.5.

$$\begin{aligned}
c = 2: &\ \underline{1}\,\underline{0}\,\underline{1}\,\underline{0}\,\underline{1}\,\underline{0} \\
&\overset{T=1}{\Rightarrow} h_\Theta(0) \approx 1,\ h_\Theta(1) \approx 0 \\
c = 3: &\ \underline{10}\,\underline{01}\,\underline{00}\,\underline{10} \\
&\overset{T=1}{\Rightarrow} h_\Theta(0) \approx 0.5\lightning,\ h_\Theta(1) \approx 0 \\
&\overset{T=2}{\Rightarrow} h_\Theta(10) \approx 0,\ h_\Theta(01) \approx 0,\ h_\Theta(00) \approx 1 \\
c = 4: &\ \underline{100}\,\underline{010}\,\underline{001}\,\underline{000} \\
&\overset{T=2}{\Rightarrow} h_\Theta(00) \approx 0.5\lightning,\ h_\Theta(01) \approx 0,\ h_\Theta(10) \approx 0 \\
&\overset{T=3}{\Rightarrow} h_\Theta(100) \approx 0,\ h_\Theta(010) \approx 0,\ h_\Theta(001) \approx 0, \\
&\qquad h_\Theta(000) \approx 1
\end{aligned} \tag{9}$$

In conclusion, the sample length T of a feed-forward DNN is of utmost importance and must be chosen large enough. This is a difficult requirement for realistic deployment, where this parameter may not be available a-priori. When the parameter has been obtained, the predictor is capable of achieving satisfactory performance; this means that the channel access model's switching probability p is approximated, and consequently the smaller the noise $1 - p$, the more accurate the prediction. Finally, even if the exact value of p was known, this couldn't lead to a better prediction: with probability $1 - p$, no switch occurs, and whether this happens in this timeslot or not is up to chance – unfortunately, chance cannot be beaten.

V. Recurrent Neural Networks

The popular choice for time series prediction problems are, at the time of writing, Recurrent Neural Networks; in particular, LSTM networks are usually used. RNNs form a family of ANNs that perform well at processing sequential data, according to [10]. In a nutshell, memory cells are added in order to store correlation on the input data over extended periods of time – the information of the input data no longer propagates only forward.

A prominent article from 2015 on *Deep Learning* in Nature [12] contains Fig. 9, describing a LSTM network. In it, x_t is the input at time step t, s_t the hidden state at the same moment in time, and o_t the corresponding output. The same parameters, matrices U, V, W are used for each time step, and the *state* at each time step *influences* the state and thus the output at the next time step – this is the major distinction from classical feed-forward ANNs, where only the *error* is propagated back through the network to update weights and thus affect the output.

According to [13], the amount of time that information is kept in memory is not set a-priori, but is part of the learning process. Even dependencies over large distances in the time domain can now be modeled.

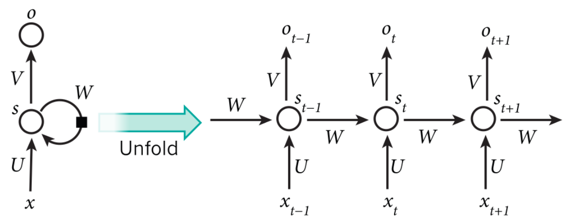


Fig. 9: A Recurrent Neural Network contains self-loops, and its architecture may be *unfolded* into a sequence of layers of a feed-forward ANN; from [12].

A. Applied on the Sequential Access Model

The LSTM model is designed to accept input matrices of dimension $T \times c$, where T again denotes the number of timeslots that are aggregated into one sample, and a binary observation on each of $c \overset{e.g.}{=} 16$ frequency channels is made per timeslot. The LSTM layer has 200 neurons and is directly connected to the output layer of c neurons, so that one prediction per frequency channel is obtained.

To observe the learning process, Fig. 10 depicts the loss during training for every sample of single timeslots $T = 1$. The pattern has periodicity $c = 16$, and it can be seen that for early samples, when a pattern has not been completely fully, the next sample increases the loss, and when a pattern sequence has been processed, the loss decreases rapidly. After a number of pattern periods, the loss converges towards zero.

To compare, consider that randomly guessing the single idle frequency channel succeeds with $p = \frac{1}{c} \overset{e.g.}{=} 0.0625$. Before any learning has taken place, the RNN is no better than randomly guessing, but after ten pattern periods it is close to perfect prediction accuracy *during training*.

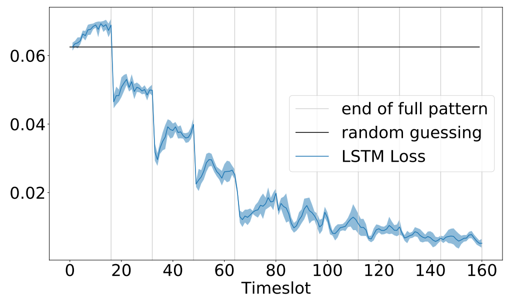


Fig. 10: LSTM network loss during training on data obtained from observing the channel for 10 full pattern sequences through $c = 16$ frequency channels in the sequential access model. Observing one full sequence is shown with vertical lines, and a comparison to randomly guessing the correct idle channel is given through the horizontal line.

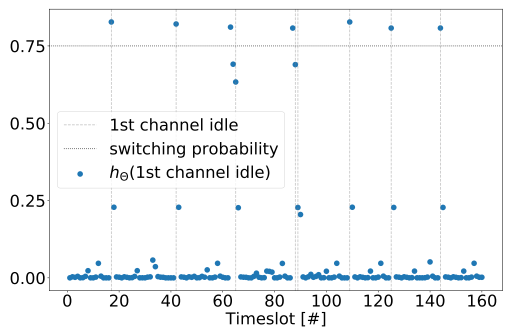


Fig. 11: Trained LSTM model predictions on whether the first channel is idle. The 1st channel being in fact idle is highlighted through vertical, dashed lines. The sequential channel access model's switching probability is shown through a horizontal line.

To gain insight into what the RNN learns, a model is trained on 1000 timeslots, and its raw prediction values on whether the *first* frequency channel is idle is plotted for $10c$ timeslots in Fig. 11. We have used a softmax activation function in the output layer to transform the prediction values to a probability distribution, and a dense layer of 150 neurons in-between LSTM and output layer – the evaluation in Fig. 10 converges with this setup as well, but requires significantly more timeslots to do so, so it was easier to show the trend with a simpler model before.

It can be seen that the prediction value is $h_\Theta(\text{1st channel idle}) \approx p$ when this channel is due for being idle. Apparently, the model has learned the pattern and "counts" the number of timeslots until this channel should be idle again. In this case $p = 0.75 < 1$, so there is a 25 % chance of a channel being idle for more than one consecutive timeslot. In Fig. 11, the first channel is idle for two consecutive timeslots once, and during the cycle before, an earlier channel is idle for several timeslots before, where the first channel is anticipated as idle too soon. It being idle for the first time is always correctly anticipated. Clearly, for as long as it is idle until it has become busy again, this likelihood of it remaining idle is reflected in a prediction $p < h_\Theta(\text{1st channel idle}) < 0$; the model has learned to account for this, too.

B. Applied on the Distance Measuring Equipment Model

Now we apply the learning model on the practical, realistic DME channel access model. We set $c = 2$ for the DME interrogation and response channels, as a regional DME operation will only occupy these two channels. The learning model is as before, but no softmax transformation is required, because both channels may be busy or both idle. Our implementation simulates a PU DME interrogator aircraft and a SU aircraft. The two are equidistant at $d = 150\,\mathrm{km}$ on opposite sides of a DME ground station. Therefore a radio signal from the ground station requires $0.5\,\mathrm{ms}$ of propagation delay to reach either PU or SU, and a PU signal requires $1\,\mathrm{ms}$ to reach the SU. Timeslots are discretized to $1\,\mathrm{ms}$ so that both interrogation and response signals arrive at the SU within the same timeslot (but on different frequency channels).

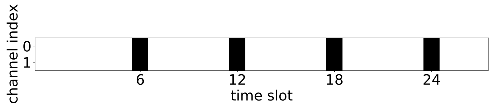
1) Long-term memory: The advantage of LSTM models is their capability of learning correlations over extended periods of time, as discussed. With this, the input sample length is no longer an all-important a-priori parameter. Now, instead, a realistic application may observe the channel each timeslot and directly feed this observation into the RNN. However, we found that the sample length still affects the learning time, which is because the argument made in Sec. IV-B still holds: if the sample length T is large enough to uniquely encode all possible channel model transitions, then each sample lets the model learn the pattern individually. The added memory of a stateful LSTM model *can* identify the correlation *between* input samples. An increased learning time shows, however, that more data is required until the pattern has been learned. If the sample length allows the encoding of all possible channel model transitions, then this still shortens the convergence time significantly.

The case for LSTM models with relatively large sample length T can be discussed further. Feed-forward MLPs that accept $c \times T$ time series inputs flattens this into a $cT \times 1$ vector, which in turn requires a large number of neurons in the first layer – the model's complexity increases with T. This is not the case for LSTM networks, where T can freely match the requirements of the system it is deployed in, such as when channel sensing is feasible, and ideally it should be large to speed up the learning process. If time allows, the sample length can also be set to a small value (e.g. a single timeslot), and training data presented to the model through several epochs – we showed results only for a single epoch.

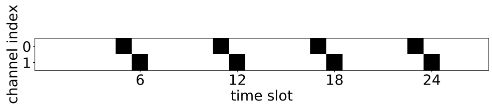
The effect is that for $T > 1$, the LSTM layer *unfolds* to allow the correlation in time over a sample, which adds computational complexity to the process. A feed-forward MLP has, for $T > 1$, additional computational complexity as well as an increased memory footprint due to more neurons.

2) Online learning: A key feature for realistic applicability is being able to adapt to changes; to continuously learn. Users will enter and leave the radio range of a particular SU, aircraft are highly mobile and when, for example, a mountain peak is passed, even sudden changes in the radio environment are possible.

Without loss of generality the DME interrogation rate is set to $D = \frac{1}{6\,\mathrm{ms}}$. Users are initially positioned as described earlier, which results in a channel observation pattern as in Fig. 12a, and after a change in positions, the pattern changes to Fig. 12b.



(a) PU and SU are equidistant on opposite sides of a DME ground station.



(b) PU and SU have moved, the PU is much closer to the ground station, the SU much further away.

Fig. 12: DME channel access patterns from the perspective of the SU.

The first $2500\,\mathrm{ms}$ are spent observing the initial pattern, the following $2500\,\mathrm{ms}$ the later pattern is observed; this corresponds to a sudden change in the radio environment. Fig. 13 clearly shows the drop in accuracy, when suddenly the learning target has changed. On first glance the drop does not seem dramatic, but it drops to $1 - \frac{1}{6} = 0.83$, which is the worst case, as the channel is usually idle (still correctly predicted), and then the single timeslot out of every six that it is not idle, it is not correctly predicted – a precision of zero is achieved.

It then shows a very quick recovery – apparently, a shift in the pattern does not require re-learning from scratch, but an adaption succeeds in much less time than the initial learning time. Right after adapting, the additional data is then used to improve accuracy past the maximum of before the change in positions.

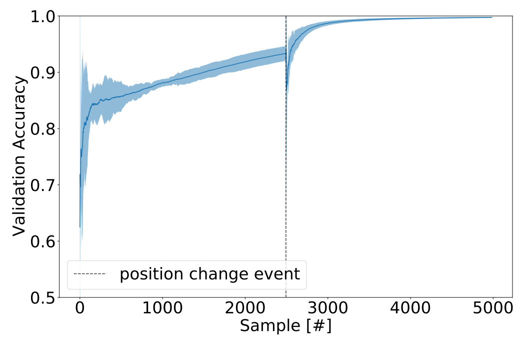


Fig. 13: Training accuracy over time, where at half the time the positions suddenly change, and with it the training target.

VI. Conclusion

We could show that the PU channel access model plays a vital role. A requirement for the application of Artificial Neural Networks is the presence of a pattern in the time series problem at hand. If the channel access model is memoryless, like the Markov model, then the MLP could be shown to approximate the model's transition probabilities instead, which fully describes the model. This could enable sophisticated interference avoidance concepts that make use of the model knowledge.

The sequential channel access model inherently presents a predictable pattern, and we could show that both MLPs and RNN models are capable of this learning task. A distinction can be made, where a feed-forward MLP requires the input sample length to be of adequate size for the pattern to be learned. This limits their applicability to those cases where the required length is known a-priori or can be quickly determined. Alternatively, Recurrent Neural Network – Long Short Term Memory ANN in particular – can circumvent this necessity through their built-in memory, through which they can recognize correlations over long periods of time. LSTM ANNs allow a free configuration of the sample length. A tradeoff must be found, where a large sample length adds computational complexity during learning, but boosts learning time if all channel model transitions are encoded. Accuracy can also be gained from training on the same data over several epochs, but this translates directly to more time spent training and may be problematic in time-critical applications.

Future research directions are manifold. We have assumed perfect channel sensing on *all* available frequency channels. Realistically, a user may only be able to choose a subset, perhaps a single frequency channel to sense or to try and transmit. Such a channel access model is thus *partially observable*, and when picking *which* frequency channel to try next becomes an action, then the whole problem transitions from Supervised Learning into Reinforcement Learning. In fact, Deep Reinforcement Learning has been applied to this problem in [7] and [8], for example. It has also been approached from the perspective of the Multi-Armed Bandit Problem in [14], and formulated in Game Theory terms in [15]. These approaches should likewise be analyzed for their reliability, potentially combined with the benefits of LSTM models, and finally these analysis-focused results should be engineered into a feasible medium access protocol, either directly or through an abstraction into RL, Game Theory or other. While the capability of finding idle gaps in the medium have been shown *for a single user*, such a protocol should take into account the coexistence with other SUs and achieve global fairness. Implementation files are made available in [16].

Acknowledgment

This work was partially funded by the German Federal Ministry for Economic Affairs and Energy (BMWi) as part of the *IntAirNet* project with reference number 20V1708F.

References

[1] Morteza Mehrnoush et al. "On the Fairness of Wi-Fi and LTE-LAA Coexistence". In: *IEEE Transactions on Cognitive Communications and Networking* 4.4 (Dec. 2018), pp. 735–748. DOI: 10.1109/tccn.2018.2867032.

[2] Maziar Nekovee. "A Survey of Cognitive Radio Access to TV White Spaces". In: *International Journal of Digital Multimedia Broadcasting* 2010 (2010), pp. 1–11. DOI: 10.1155/2010/236568.

[3] Ulrich Epple, Felix Hoffmann, and Michael Schnell. "Modeling DME Interference Impact on LDACS1". In: *2012 Integrated Communications, Navigation and Surveillance Conference*. IEEE, 2012, G7–1.

[4] T. Charles Clancy and Brenton D. Walker. "Predictive Dynamic Spectrum Access". In: *SDR Forum Technical Conference*. IEEE, 2006.

[5] Marko Höyhtyä, Sofie Pollin, and Aarne Mämmelä. *Performance Improvement with Predictive Channel Selection for Cognitive Radios*. Tech. rep. Center. 2008.

[6] Vamsi Krishna Tumuluru, Ping Wang, and Dusit Niyato. "A Neural Network Based Spectrum Prediction Scheme for Cognitive Radio". In: *IEEE International Conference on Communications*. IEEE, 2010. DOI: 10.1109/ICC.2010.5502348.

[7] Shangxing Wang et al. "Deep Reinforcement Learning for Dynamic Multichannel Access in Wireless Networks". In: *IEEE Transactions on Cognitive Communications and Networking* 4.2 (2018), pp. 257–265. DOI: 10.1109/TCCN.2018.2809722.

[8] Chen Zhong et al. "A Deep Actor-Critic Reinforcement Learning Framework for Dynamic Multichannel Access". In: *IEEE Transactions on Cognitive Communications and Networking* 5.4 (Dec. 2019), pp. 1125–1139. ISSN: 2372-2045. DOI: 10.1109/TCCN.2019.2952909.

[9] Ulrich Epple et al. "Overview of Interference Situation and Mitigation Techniques for LDACS". In: *IEEE/AIAA 30th Digital Avionics Systems Conference*. IEEE, 2011.

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[11] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[12] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep Learning". en. In: *Nature* 521.7553 (May 2015), pp. 436–444. ISSN: 1476-4687. DOI: 10.1038/nature14539.

[13] Y. Bengio, P. Simard, and P. Frasconi. "Learning Long-Term Dependencies with Gradient Descent Is Difficult". In: *IEEE Transactions on Neural Networks* 5.2 (Mar. 1994), pp. 157–166. ISSN: 1941-0093. DOI: 10.1109/72.279181.

[14] Keqin Liu and Qing Zhao. "A Restless Bandit Formulation of Opportunistic Access: Indexablity and Index Policy". In: *2008 5th IEEE Annual Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops*. June 2008, pp. 1–5. DOI: 10.1109/SAHCNW.2008.12.

[15] Kobi Cohen, Angelia Nedić, and Rayadurgam Srikant. "Distributed Learning Algorithms for Spectrum Sharing in Spatial Random Access Wireless Networks". In: *IEEE Transactions on Automatic Control* 62.6 (2016), pp. 2854–2869.

[16] Sebastian Lindner. *Coexistence of Shared-Spectrum Radio Systems through Medium Access Pattern Learning Using Artificial Neural Networks Code Release*. https://doi.org/10.5281/zenodo.3971740. 2020.