# SPECTRUM ANALYSIS AND PREDICTION USING LONG SHORT TERM MEMORY NEURAL NETWORKS AND COGNITIVE RADIOS

Jorge Luis Hernandez Villapol

Thesis Prepared for the Degree of

MASTER OF SCIENCE

UNIVERSITY OF NORTH TEXAS

December 2017

APPROVED:

Kamesh Namuduri, Major Professor
Murali Varanasi, Committee Member
Bill Buckets, Committee Member
Shengli Fu, Chair of the Department of Electrical Engineering
Costas Tsatsoulis, Dean of the College of Engineering
Victor Prybutok, Dean of the Toulouse Graduate School

Hernandez Villapol, Jorge Luis. *Spectrum Analysis and Prediction Using Long Short Term Memory Neural Networks and Cognitive Radios*. Master of Science (Electrical Engineering), December 2017, 53 pp., 2 tables, 27 figures, 31 numbered references.

One statement that we can make with absolute certainty in our current time is that wireless communication is now the standard and the de-facto type of communication. Cognitive radios are able to interpret the frequency spectrum and adapt. The aim of this work is to be able to predict whether a frequency channel is going to be busy or free in a specific time located in the future. To do this, the problem is modeled as a time series problem where each usage of a channel is treated as a sequence of busy and free slots in a fixed time frame. For this time series problem, the method being implemented is one of the latest, state-of-the-art, technique in machine learning for time series and sequence prediction: long short-term memory neural networks, or LSTMs.

ACKNOWLEDGEMENTS

To my mother and my father, the sources of my inspiration and success.

Special thanks to my major advisor Dr. Kamesh Namuduri for taking me under his wing and guide me into this wonderful field.

Finally, a great thank you to my friends of the Autonomous Lab: Roya, Kunal, Ramanpreet, Oghenetega and others who made my time in the lab and my UNT experience an amazing time of my life.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

# CHAPTER 1

# INTRODUCTION

## 1.1 Motivation and Previous Work

One statement that we can make with absolute certainty in our current time is that Wireless Communication is now the standard and the de-facto type of communication. People and customers no longer desire but instead demand and need Wireless Communication between the ever-growing number of devices in their life. From smartphones, alarm clocks, wearables to Bluetooth connectivity in smart cars, the ability to communicate devices without a physical connection is present.

However, the ability to send information over the air is not without limitation. From the beginning of the history of Wireless Communication there was a limitation in how much data we could send over a spectrum band. The Nyquist-Shannon sampling theorem [1] states that there is a limit frequency that we can use as a maximum frequency to be able fully decode the transmitted signal. This frequency, known as Nyquist Rate, is the first hurdle that early analog devices had to surpass to use higher frequencies and lower bandwidths, which are necessary to make Wireless communication viable.

The new digital devices are able to easily, and in a cost-effective way, beat this Nyquist Rate. The number of devices using Wireless Communications increased rapidly and it seemed the sky was the limit and they were correct. With still a limited number of channels in the frequency spectrum, restrictions and bands were placed in place with the intention to regulate this natural resource [2] [3]. Apart from the government restricted, the emergency bands and the commercial bands, there is a “free” band which the new devices can use to communicate. With the

substantial number of devices getting thrown into that free band, then arises a new problem of overpopulating that frequency.

Here is where a new generation of devices is born, Cognitive Radios, which are able to interpret the frequency spectrum and adapt. These devices do not come without issues, which will be explained later in chapter 2; the main issue that this thesis work is aimed to solve is known as the Hidden Primary User. While other research work focus on a cooperative approach or the design of better detection techniques, this paper will focus on the prediction of future usage of the frequency spectrum.

With the ability to predict usage of the frequency bands we will be able to find underutilized zones, commonly known as spectrum-holes or white spaces, see Figure 1-1. These pockets of free spectrum could be used by a cognitive radio, therefore increasing the total number devices that every specific band can support.

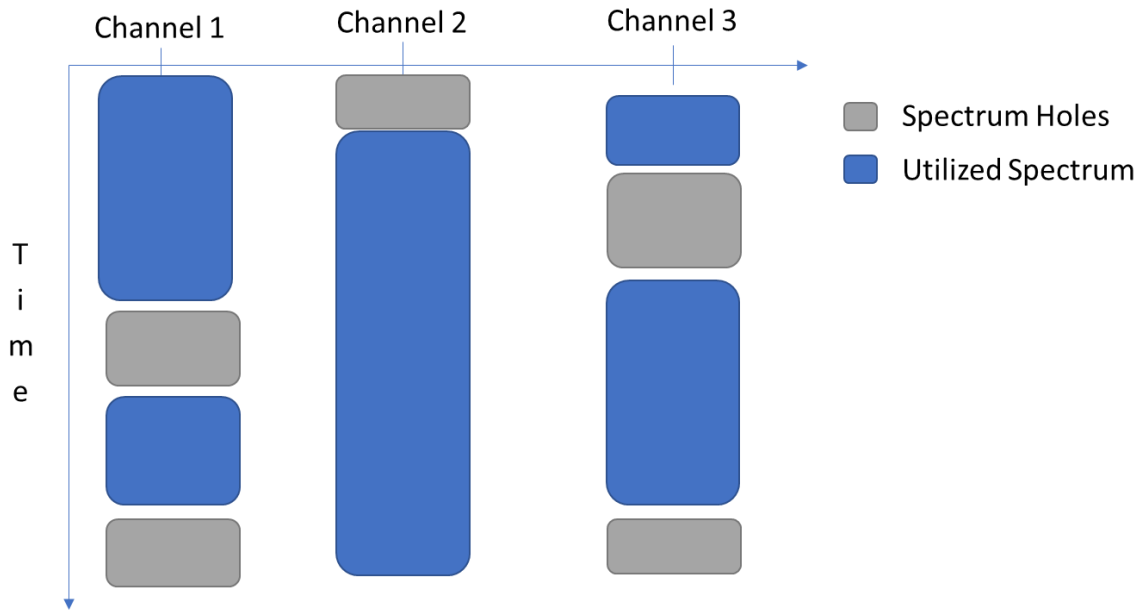


Figure 1-1 White Spaces in Spectrum

## 1.2 Objective and Significance of the Research

The aim of this work is to be able to predict whether a frequency channel is going to be busy or free in a specific time located in the future. To do this, the problem is modeled as a time series problem where each usage of a channel is treated as a sequence of busy and free slots in a fixed time frame.

For this time series problem, the method being implemented is one of the latest, state-of-the-art, technique in Machine Learning for time series and sequence prediction: Long Short-Term Memory Neural Networks, or LSTMs. The thesis work is implemented using Python 3 to simulate the network activity plus the LSTM networks are designed using the popular Python Library Tensorflow in addition with the Advanced Programing Interface (API), Keras.

## 1.3 Overview of the Chapters

This thesis provides a viable solution to the spectrum analysis and prediction using state-of-the-art machine learning techniques. The dynamic allocation problem is considered trivial when we have an exact prediction of the spectrum usage and a simple solution is provided. This work is divided into 4 chapters.

Chapter 2 provides background information and previous work results on the field of cognitive radios. Cognitive Radios are based on the software-defined radio, which is explained in Chapter 2. The typical challenges of cognitive radios are described in this chapter as well, along with current solutions found in the research field at the time of this writing. Finally, the chapter also provides a brief explaining of LSTM networks and its usage in Time Series prediction.

Chapter 3 presents the python framework designed for this thesis work. It contains the detailed methods of each of the classes present in the framework and the design requirements.

Chapter 4 explains in detail the experiment and presents the result obtained in this thesis. It also contains the simulation and results of previous work and how it compares with the solution presented in this paper. Finally, it presents a conclusion and ideas for future work.

CHAPTER 2

# COGNITIVE RADIOS: COMPREHENSIVE BACKGROUND

## 2.1 Spectrum Etiquette

As we mentioned in chapter 1, the early hurdle of limited frequency bands available for analog devices forced to impose restrictions on the Radio Frequency Spectrum (RF Spectrum). Currently the European Union (EU), NATO, and the US military have agreed on the following set of EU-NATO-US ECM frequency bands for electromagnetic frequencies.

| FREQUENCY RANGE | WAVELENGTH | EU/NATO/US ECM BAND |
|---|---|---|
| **TO 250 MHZ** | to 1.2 m | A band |
| **250-500 MHZ** | 1.2 m to 600 cm | B band |
| **500 MHZ-1 GHZ** | 600 cm to 300 cm | C band |
| **1-2 GHZ** | 300 cm to 150 cm | D band |
| **2-3 GHZ** | 150 cm to 100 cm | E band |
| **3-4 GHZ** | 100 cm to 75 cm | F band |
| **4-6 GHZ** | 75 cm to 5 cm | G band |
| **6-8 GHZ** | 5 cm to 3.75 cm | H band |
| **8-10 GHZ** | 3.75 cm to 3 cm | I band |
| **10-20 GHZ** | 3 cm to 1.5 cm | J band |
| **20-40 GHZ** | 1.5 cm to 750 mm | K band |
| **40-60 GHZ** | 750 mm to 500mm | L band |
| **60-100 GHZ** | 500 mm to 300mm | M band |

Table 2.1 EU-NATO-US-ECM Frequency Bands

With a few of these bands reserved for emergency situations and military use, the rest were assigned to commercial use. Mobile Phone Carriers, TV Stations and even conventional Radio stations are required to lease a specific frequency band in order to be able to use it. These loans are also bound by geographical location. Finally, the Wi-Fi frequency band (2.4Ghz and 5Ghz) is the only band where new digital devices can communicate freely.

Since the Wi-Fi is the only band that allows these devices, the band itself is overcrowded with devices. An immense amount of research is being devoted to networking to take the most of this frequency band. In addition, with the design requirement of having limited range, these radios are coexisting. However, the rest of the RF spectrum is being underutilized.

Cognitive Radios bring up the idea of sharing the restricted frequency bands. The radios that obtained the lease for the band are known as Primary Users (PU) while the new cognitive radios sharing the band are known as Secondary Users (SU). By defining a very simple Spectrum Etiquette, SU cannot interfere with the signal from the PU in any point in time. Therefore, the ability to detect and predict PU usage of the spectrum becomes a fundamental task of Cognitive Radios.

## 2.2 Software Defined Radios

While Cognitive Radios do, in a high level, the same function as conventional radios, transmit and receive information, Cognitive Radios possess an essential difference: They need to be able to change the transmitting and/or the receiving frequency. Conventional Radios do have the ability to manually change the receiving frequency, which allows us to listen to multiple commercial signals present in the RF Spectrum. However, Conventional Radios are not designed

to manually change the transmitter frequency. These radios which are used as transmitters are carefully tuned to broadcast in a single frequency in the most cost-effective way.

Software Defined Radios (SDR) are radios whose specifications such as Sample Rate, transmitter frequency and receiver frequency are controlled by software, making these types of radios ideal for Cognitive Radios.

Conventional hardware components like filters, amplifiers, modulators/demodulators, etc. in a traditional radio are replaced by software-based functions [4]. The user has the ability to implement any type of wireless communication configuration just by choosing the correct specific software. SDRs are an inexpensive solution for the conventional radios as it eliminates the need to build and tune new circuits for different radio applications. The benefit of this technology is multi-functionality by reusing the same hardware platform [5].

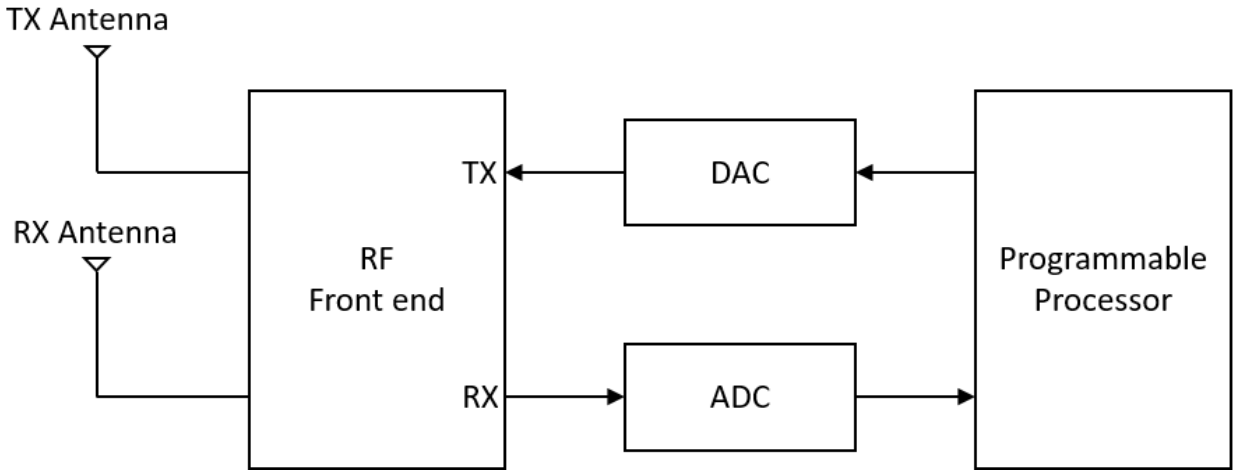


Figure 2-1 Block Diagram of SDR

SDRs are able to transmit and receive signals at different frequencies as they have access to broad spectrum (e.g. the commercial SDR: bladeRF has frequency range 300 MHz to 3.8 GHz, other commercial radio USRP covers frequency range from DC to 6 GHz) [4], [6], [7].

The user can use an Integrated Development Environment (IDE) such as GNU Radio to develop complex signal processing pipelines. Software Defined Radios consist of an array of Field Programmable Gate Arrays (FPGA), General Purpose Processors, Digital Signal Processors and other specific programmable processors as shown in Figure 2.1. For the radio developer, SDR is the tool devised to provide a cost-effective solution to universal wireless communications. Some examples currently include bladeRF, rtl-sdr, SDRPlay and Ettus research radios (USRP N210, USRP E310, etc.) [6], [7].

Cognitive Radio Systems autonomously make the most use of available unoccupied spectrum to provide new paths to spectrum access resulting in uninterrupted communication. Cognitive Radio Systems are aware of license holders, Primary Users, and other secondary users who are opportunistically looking for a chance to access the spectrum when primary users are idle. The main idea behind CR system is to reuse the spectrum or share the spectrum, allowing more secondary systems to communicate over the allocated or licensed channel when it is free. The principal problem of the CR system is that they might increase the interference of secondary users with the primary users and worsen the quality of service of primary systems. Hence, the main design requirement is to avoid interference with the primary users at all cost. In order to achieve this goal, the CR system needs to be able to continuously monitor the spectrum and detect when a certain frequency is not being used by the primary user, not only that but the CR also needs to manage its own signal by allocating in a dynamic way on an available frequency.

The different tasks of Cognitive Radios are divided into three parts, as shown in Figure 2-2 below. The importance of spectrum sensing lies within the basic idea of cognitive radio. To detect the presence or absence of primary users, the spectrum sensing is required. As any other

conventional radio, the ability to decode and demodulate the received signal, signal decoding must be the part of cognitive radio. Finally, Dynamic Spectrum Access (DSA) is the result of cognitive radio as depending on the first two functions, and cognitive radio makes the decision and changes the RF parameters to transmit the data on the free channel.

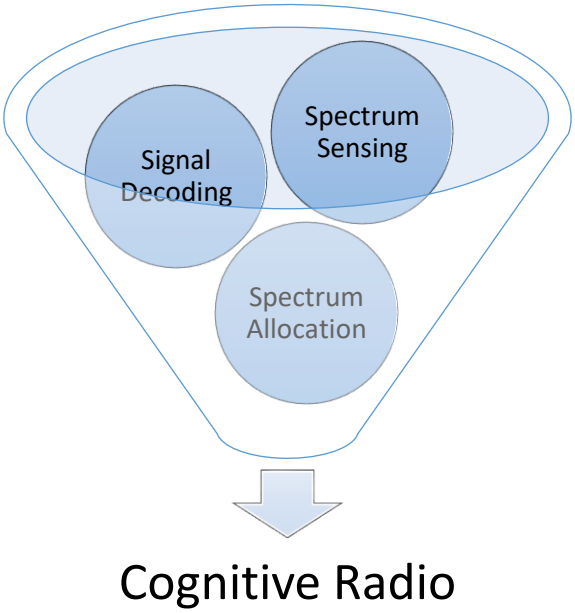


Figure 2-2 Functional Division of CR System

## 2.3 Spectrum Sensing

As mentioned before, the ability to sense the spectrum is one of the basic components of the cognitive radios. The problem of spectrum sensing is not trivial. The spectrum sensing is one of the most essential components of cognitive radio system as it must be able to detect weak primary user signals as well. The secondary users need to frequently sense the spectrum in search of the spectrum holes or the existence of primary users. As soon as the primary users are detected, the secondary users must reduce the transmission power and vacate the channel

within a certain amount of time to avoid the interference with the primary users. The spectrum sensing involves the following subtasks:

- Find the spectrum holes
- Find the presence of primary user
- Estimate the direction of incoming interferences
- Classify Signals

Signal to Noise Ratio (SNR), noise level and multipath interference are some of the issues encountered when analyzing the spectrum. Even in the event of the primary user's transmitter being in a close distance to the cognitive radios receiver, the transmitted signal can go well below -20dB. In this case the Cognitive Radio System still needs to detect the signal, a failure to detect the primary user's signal could cause a secondary user to interfere with a primary user. The decay of the primary user signal power can be due to many factors, such as multipath fading. However, one unlikely scenario can be due to the variation of the noise level. Simpler spectrum sensing techniques could not deal with the variation of the noise level, since they require a calibration stage where a threshold is chosen to detect signals. If the noise level increases, due to natural reasons or multiple devices doing interference, then the signal detection fails.

In the literature can be found a few spectrum sensing techniques, most of which can be divided into 3 types:

a) Blind Methods: do not require any information about the source signal. Methods such as eigenvalue-based sensing or covariance-based sensing are blind methods.
b) Noise Methods: only require information about the noise variance. Energy Detection is the primary example of these type of methods.

c) Informed Methods: These methods do require complete information about the source signal. Matched filtering or cyclostationary detection are informed methods.

Table 2.1 shows the brief analysis of the mainly used techniques for spectrum sensing. Each cognitive radio system uses a spectrum sensing technique within its signal range and decides on which channel is available. Energy detection is the classic and simplest spectrum sensing method. It detects users by measuring the signal power over a frequency range.
The signal is detected by performing a power spectral density analysis, or PSD.
Afterwards, the magnitude square is taken to remove imaginary elements and perform a simpler comparison. An optional step for the energy detection algorithm is to perform an average of *n* PSD samples or using a low pass filter to reduce false positives due to a noisy environment. Finally, in order to detect a signal, the magnitude of the power signal is compared to a previously calibrated threshold. If the power signal is greater than the threshold then the algorithm indicates that a user is present over the analyzed range frequency, otherwise the frequency band is free. A brief block diagram for energy detection is shown in Figure 2.3

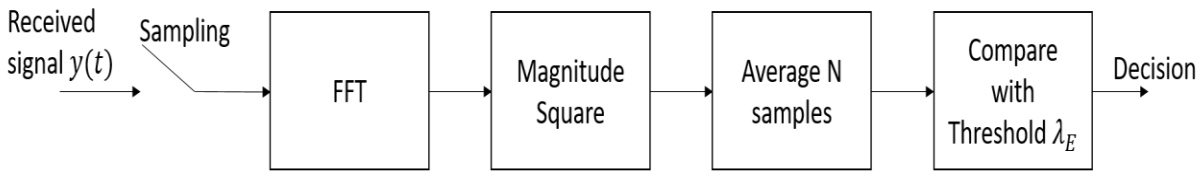


Figure 2-3 Schematic Representation of Energy Detection

The cyclostationary method exploits the periodicity in the target signal. A cyclic correlation filter is designed in order to match the desired signal. This method usually outperforms the energy detection method since it does not only detect the presence of a primary user but also is able to differentiate between primary users. The downside of the cyclostationary method is that in order to correctly identify the correct primary user, information about the target signal is required.

The wave-form method is very similar to the cyclostationary method. It exploits the periodicity of the desired signal. However, while the cyclostationary method takes the information of the whole signal as input, the wave-form method is only looking for specific parts of the message within the signal. Preambles, midambles, access codes and other common patterns are used in communications systems as milestones for synchronization and decoding reasons. These chunks of data are usually fixed and located on certain parts of the message and therefore are also present on the target signal. If this information is available, a correlation filter is implemented to search for those chunks of data within the spectrum, whenever they are found. The wave-form algorithm is certain that the primary user is present in the channel.

The last method listed on Table 2.1 is the Matched Filtering. This method needs the exact form of the desired signal and as the name indicates looks for an exact match in the spectrum. Although this method yields the best accuracy over the other methods, it also requires the major amount of compute time, due to its complexity, and requires the absolute knowledge about the target signal.

Table 2.1 Common Spectrum Sensing Techniques

| Method | Complexity | Accuracy | Pros | Cons |
|---|---|---|---|---|
| **Energy Detection** | Low | Low on low SNR areas | Fast and Simple | Weak to noise |
| **Cyclostationary** | Mid | Mid | Can detect and classify signals | Weak to noise |
| **Waveform-Based** | Low | Mid | Simple | Must know preamble |
| **Matched Filtering** | High | High | Best possible accuracy | Requires prior knowledge about the signal |

## 2.4 Cooperative Sensing

In addition to the problems mentioned above, noise variation, multipath fading and multidevice interference, another big issue with spectrum sensing is formally known as the Hidden Primary User problem. This problem arises because of inability to detect a primary user due to the location of the radios. In some scenarios the cognitive radio is located where the primary user signal does not reach; nevertheless, the primary user is still present. Most of the methods mentioned before have a challenging task when the hidden primary user problem arises. If the primary user signals do not reach the cognitive radio system, none of the methods listed above will work, due to the fact that all of them search the spectrum in hope to locate the target signal.

The research community has taken two approaches to this problem, this thesis work will explore a third approach. The first approach is to improve the detection algorithm methods; optional steps like adding low pass filters are known tricks to improve the performance on the detection algorithms in low SNR scenarios. The second approach is by creating a network of cognitive radio devices in order to form a cooperative sensing technique. Cooperative sensing decreases the probability of false alarm and probability of misdetection, which are major problems in local sensing [8], [9].

With the idea of mesh networks of cognitive radio devices sensing the spectrum on various locations the hidden primary user can be overcome. However, having a network of cognitive devices generates new variables and issues to take into consideration. The primary issue of this approach is to converge into a coherent solution. This issue is trivial when all the cognitive devices arrive to the same conclusion, whether or not a primary user is present in a frequency band. But, when there are some opposing conclusions, an extra step or algorithm is needed to achieve the correct decision.

From simple voting systems to machine learning techniques, reaching a consensus between the nodes in the network is a whole problem on its own. A simpler voting system can be implemented to reach consensus, where each cognitive device votes for whether or not the space is available, and the option with the bigger number of votes wins. Another simpler system is used by taking a weighted average, in which votes coming from better located cognitive devices will carry more weight than cognitive devices' poorer locations. This issue can also be solved using Machine Learning, a meta-learner or super learner as described in [8], can be used. Each prediction of the cognitive device is taken as an input for the stacked model and trained with a

known result. Finally, the super learner model can make accurate decisions based on what each cognitive radio reported.

The second but lesser problem with the cooperative spectrum sensing technique is that communication between the nodes is required. This means that a pre-defined frequency channel needs to be free in order to allow the cognitive devices to communicate. The requirement of using an extra channel just for synchronization purposes goes against the idea to maximize the spectrum usage. One last aspect worth mentioning of the cooperative sensing technique is that each cognitive radio device is free to use the spectrum sensing techniques that it pleases. This way a cooperative sensing technique not only gives a superior spectrum analysis against the hidden primary user problem, but it also achieves better performance on detecting primary user as it combines all the spectrum sensing techniques mentioned above.

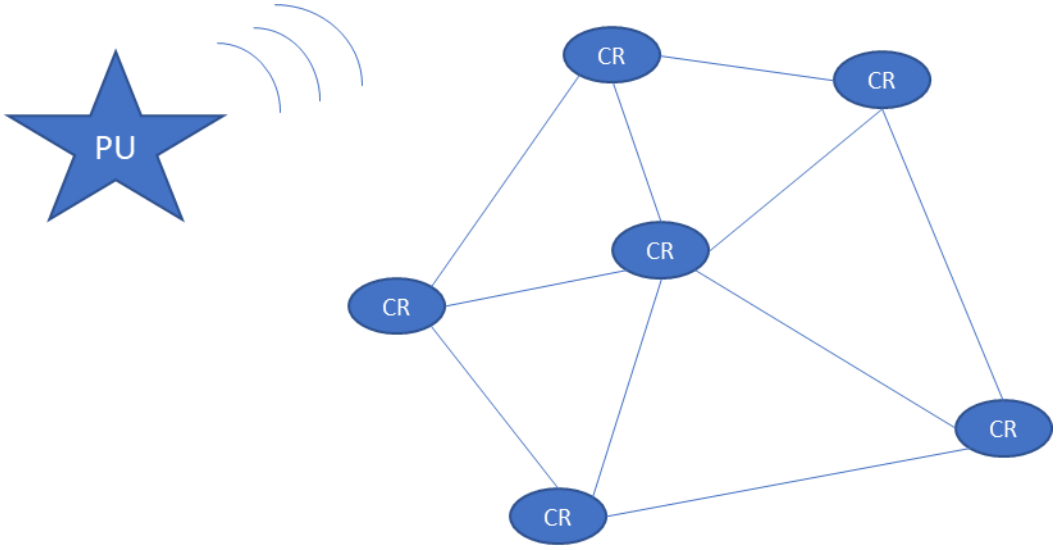


Figure 2-4 CR Network in the Presence of a PU

## 2.5 Dynamic Spectrum Access

Another fundamental task of the Cognitive Radio system is how to allocate the transmitted signal into an available frequency slot. Since Software Defined Radios can change its radio parameters like the center frequency, bandwidth and transmitter power. It is possible for the cognitive radio to jump or switch between different frequency channels. When the cognitive radio looks for an opportunity to use a certain frequency band, it is called opportunistic spectrum access. Hence, the accuracy of the spectrum sensing technique that is being used by the cognitive device has a direct impact in the performance of the spectrum access policy.

As well as the spectrum sensing, there are some DSA models that can be used: interweave, underlay and overlay [10], [11]. The interweave model is the primary model used for dynamic spectrum access and in this thesis work, as it switches to another frequency whenever a primary user is detected within the same band. In this model the primary user has unrestricted and privileged access to the frequency and no other signal is allowed when the primary user is an active state. On the other hand, the overlay model allows the secondary user to use the same frequency channel as the primary user at the same time as long as it does not interfere with the performance or quality of the primary user signal. Finally, in the underlay model the secondary users are allowed to transmit over the spectrum even in the presence of primary users. This is usually achieved by transmitting with ultra-wide band technology (UWB). With this technology the transmitted power of the secondary users is distributed over a large frequency range and do not interfere with the narrower primary user's signal.

## 2.6 Time Series Predictions Using Machine Learning

As mentioned before, the spectrum usage prediction problem can be modeled as a time series problem. Figure 2.5 shows how can the spectrum usage can be converted into a time-based sequence. Time series is a well-studied problem in the literature and multiple solutions have been published in order to deal with this specific type of problem. Even today new techniques are being developed to get better performance in time series prediction. For this thesis work, the main technique used is the Recurrent Neural Networks.

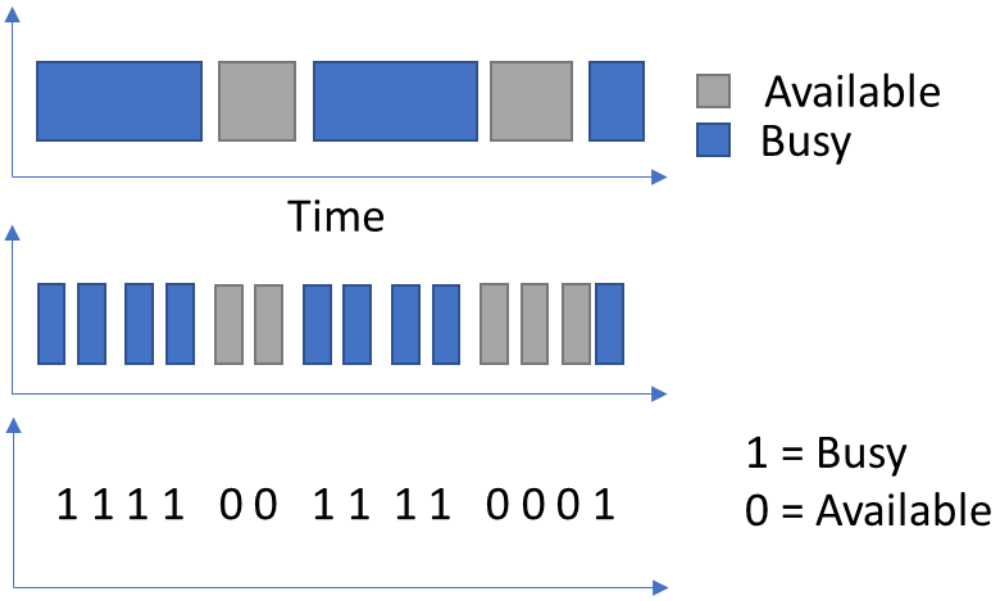


Figure 2-5 Spectrum Usage as Time Series Problem

As shown above, the spectrum behavior can be modeled into a sequence of zeros and ones. For this paper 1 means that the channel is being used by a primary user while 0 means that the channel is available to be used. As any time series problem, it is important to have samples spaced into a fixed time axis. This method does not possess a thread to spectrum sensing since all the algorithms perform a periodic check on the spectrum.

Multiple examples of time series problems solved by Machine Learning can be found in the literature. From the stock market prediction to speech recognition software, all of these time-based problems are being solved using machine learning.

The traditional time series prediction technique is the Autoregressive integrated moving average, also known as ARIMA models [12] [13]. The ARIMA model is the final combination of 3 basic models: The Autoregressive model or *AR*, the integrated model or *I* and the moving average model (MA). The Autoregressive part of the ARIMA models try to predict the future values of a time series by a statistical model based on past input values. On the other hand of the equation, the moving average model tries to predict future values by the average of past output values.

$$X_t = \sum \alpha_i X_{t-i}$$

AR Models

$$X_t = \sum \beta_i \varepsilon_i$$

MA Models

$$\left(1 - \sum_p \alpha_i L^i\right)(1-L)^d X_t = \left(1 + \sum_q \beta_k L^k\right)\epsilon_t$$

ARIMA MODEL

Another classical method for dealing with time series problems is Hidden Markov Models [14] [15]. This technique is a stochastic model that consists of the change of the system state every time step or with every input. Since many time series problems do not have a clear source nor deterministic equation to rule their behavior, hidden Markov models provide the best approximation from all the Markov based models. With simpler Markov models the probability

distribution of the states is known. However, in hidden Markov models (HMM) these states are not known and are trained using supervised learning techniques to emulate the behavior of the desired target signal.

In every Markov model the output is given by the combination of the input, if any, and the probability density function given by the current state. These probability functions are unique for each state and not only dictate the output of the system but also the state transitions. Figure 2.6 provides a simple example of a 3 state HMM with two possible outputs (1 or 0). The transition probabilities as in any hidden Markov Models are unknown.

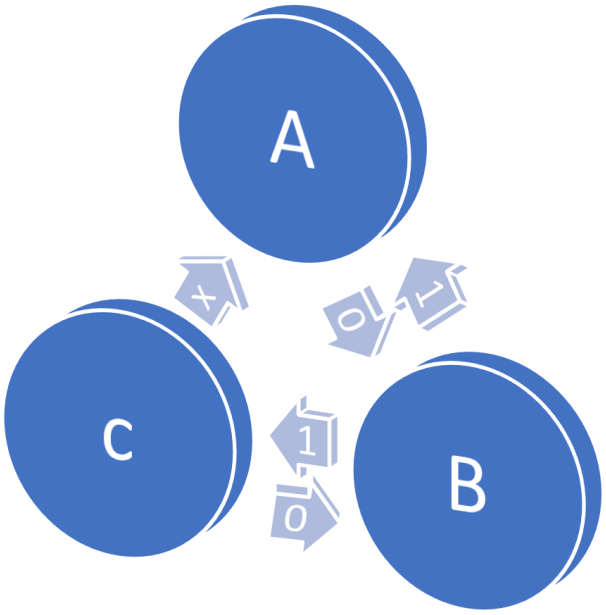


Figure 2-6 Hidden Markov Models with 3 States

Finally, another classical time series solution is Recurrent Neural Networks [16]. Since their creation neural networks have been known as black boxes capable of mimicking any mathematical function, linear or non-linear. Recurrent neural networks go a step beyond vanilla neural networks and incorporate one or multiple feedback loops. Thus, adding the temporal

component to neural networks. At any time step the perceptron in the recurrent neural network not only has the input and the bias component, present at every neural network, but it also takes as input the previous output of the same perceptron. This way, the neurons learn to recognize sequences, seasonality and other time-related behaviors.

## 2.7 Long Short Term Memory Neural Networks (LSTM)

As mentioned above, the Recurrent neural networks are a special configuration of vanilla neural networks. Figure 2.7 presents what could be known as a vanilla recurrent neural network with a feedback loop in the hidden layer unrolled over time. It is shown that: at the time $t_1$ the inputs are the output of the state $A_0$ and the input $x_1$.

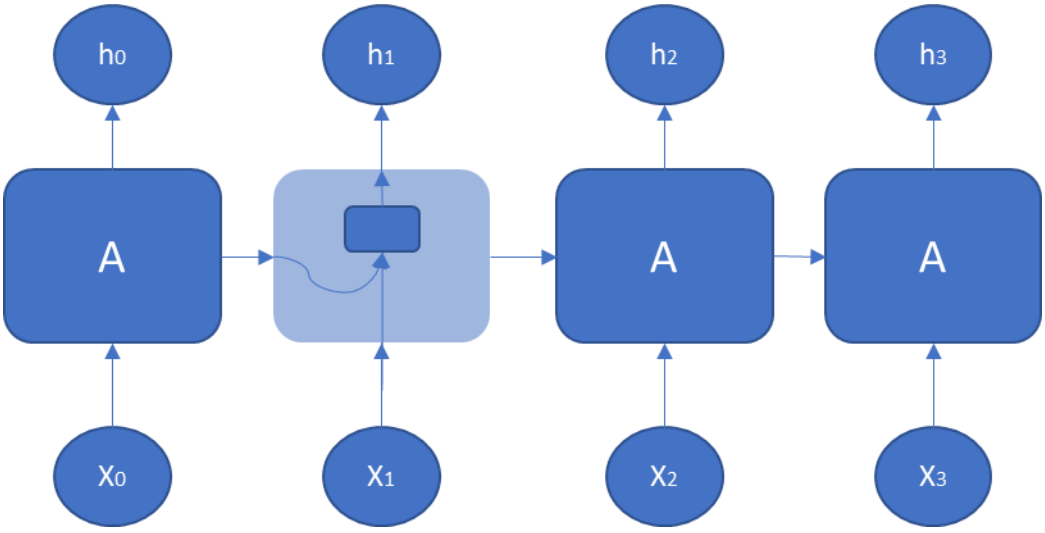


Figure 2-7 Unrolled Vanilla Recurrent Neural Network

Inside each perceptron of the neural networks resides an activation function, depending on the range of the target value, this function could be either a sigmoid or a hyperbolic tangent function. Usually hidden perceptrons are stacked in layers. The number of layers and the number of perceptrons per layer is one of the most complicated problems found in neural networks.

Another fundamental characteristic about Recurrent Neural Networks (RNNs) is that they possess the ability to be Multiple Input Multiple Output (MIMO). Figure 2.8 shows the multiple types of networks configurations that are possible with RNNs. In the example of one to many of many to one, it is possible to use one single input to produce a sequence or with a sequence as an input it is possible to predict the next element of the sequence.

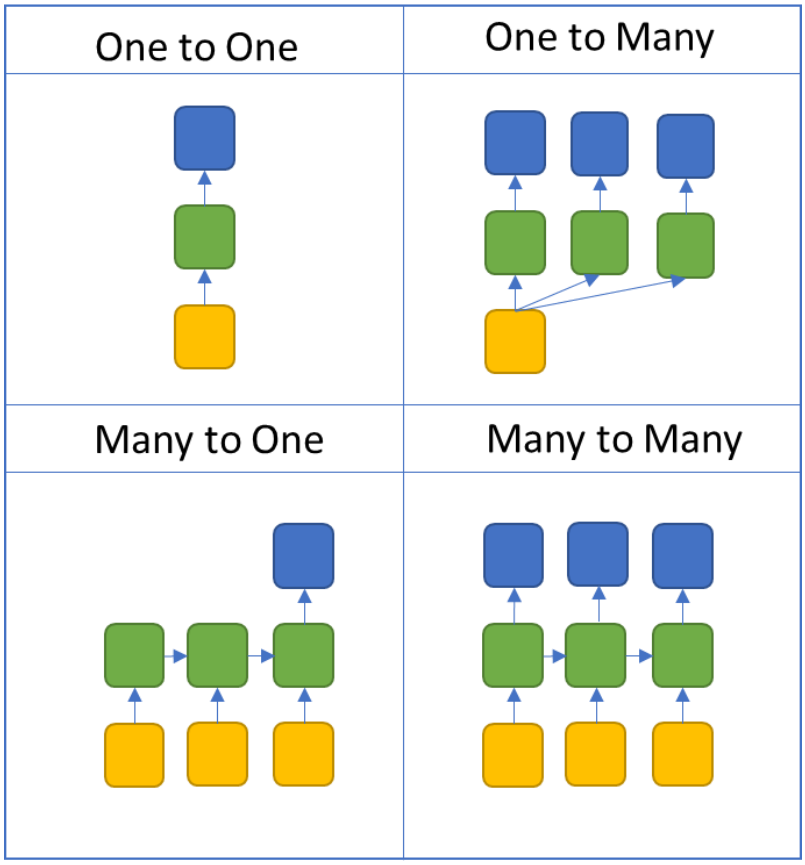


Figure 2-8 Recurrent Neural Networks Configuration Types

One of the main problems with Recurrent neural networks is that there is no option to select which information to remember, because all the information is recorded in the same way. Hence, some notable events do not have the impact in the learning process that they should. With this idea on mind, Long Short Term Memory networks add multiple gates and operations within the regular neural network cell.

These new operations are called gates in the LSTM architecture. The first gate is commonly known as the forget gate, because its function is to select which of the data the network forgets at a specific point time step. The second gate of the LSTM is known as the input gate layer, which works exactly as a regular RNN. Finally, the last gate is called the candidate gate. This gate prepares a set of candidate values to be added to the final state of the cell. The whole LSTM cell is shown in the figure below along with the characteristic equation of each of the gates.

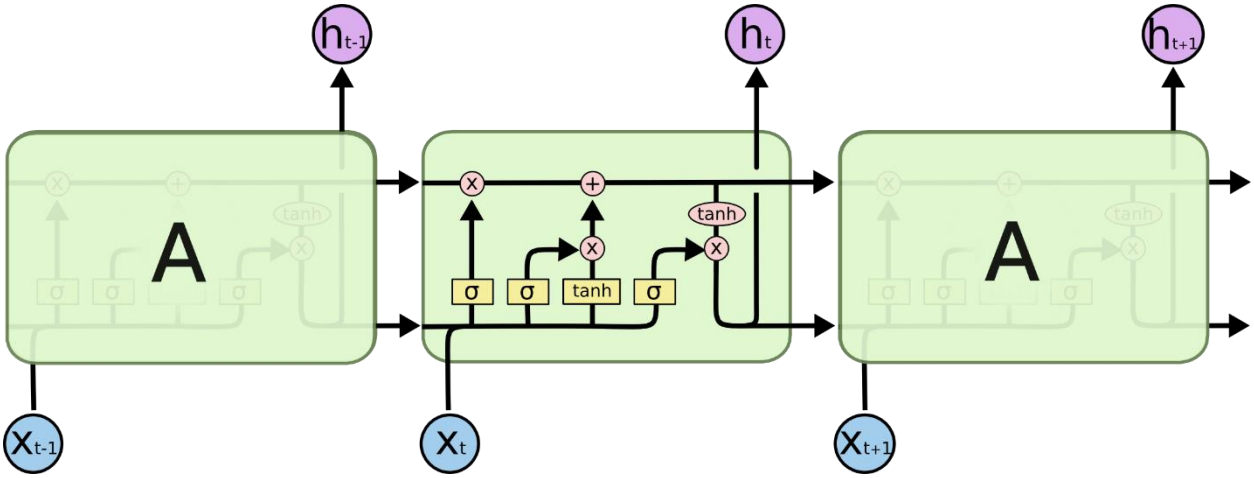


Figure 2-9 LSTM Cell [17]

Forget Gate: $f_t = \sigma\left(W_f[h_{t-1}, x_t] + b_f\right)$

Input gate Layer: $i_t = \sigma(W_i[h_{t-1}, x_t + b_i)$

Candidate Gate: $\tilde{C} = \tanh(W_c[h_{t-1}, x_t] + b_C)$

It is important to notice that the LSTM cell has two inputs and two outputs. As input it takes the input of the system and the previous state, just like the feedback loop of the vanilla RNN. As output it has the output of the system and the current state of the cell.

Cell State $C_t = f_t * C_{t-1} + i_t * \tilde{C}$

$$o_t = \sigma(W_o[H_{t-1}, x_t] + b_o)$$

Output $h_t = o_t * \tanh(C_t)$

## 2.8 Tensorflow Framework

Tensorflow is an open source library for numerical computations, originally designed by the Google Brain Team within the Google's Machine Intelligence research organization. With the main focus being to use Tensorflow as an Advance Programing Interface (API) to build and train machine learning applications, especially Deep Learning applications, the Google Brain Team decided to open source the code of Tensorflow as it was found to be useful in a wide variety of solutions.

At the time of this writing Tensorflow is used as a backend for the machine learning of most of Google's predictive services and many other technology industries. Tensorflow works by allowing the user to program computational graphs that are later compiled and executed at an efficient speed. As the name indicates, the basic units inside Tensorflow are called tensors. These tensors are mathematical manifestation of a vector. With tensors the user can create mathematical expression that will be executed during runtime. This ability to easily program complex mathematical expressions is the core advantage of using Tensorflow. The library also comes with pre-defined estimators and optimizers, making possible the execution of the training process of the neural network.

Among the many features of the Tensorflow library is a module for recurrent neural networks and LSTMs. This feature allows the user to build a completely connected LSTM cell in a single line of code without the requirement of building all the gates equations stated above. By

concatenating the LSTM cells with the weights update function and the preferred optimizer, the user can design with incredible amount of detail its own neural network application.

## 2.9 Keras API

Even though Tensorflow provides many high-level features like the LSTM cell, the user still needs to provide all the network connections along with the tensors initialization, making a tremendous amount of possible human errors while assembling the neural network application. The Keras API provides a higher-level interface that allows the user to build a deep neural network in a much simpler and user-friendly manner while still using Tensorflow as the backend. The only notable downside of the Keras API is that it removes some customization capabilities only available in the Tensorflow API.

While Tensorflow uses graphs as the overall model, Keras defines Sequential models that can be built simply by adding layers of operations. Sequential models and layers are the basic units inside the Keras API. Multiple types of layers are already defined in the library: Dense layers are the vanilla neural network perceptron while a special LSTM cell is already coded inside Keras. Figure 2.10 provides an example on how LSTM networks are constructed with this library.

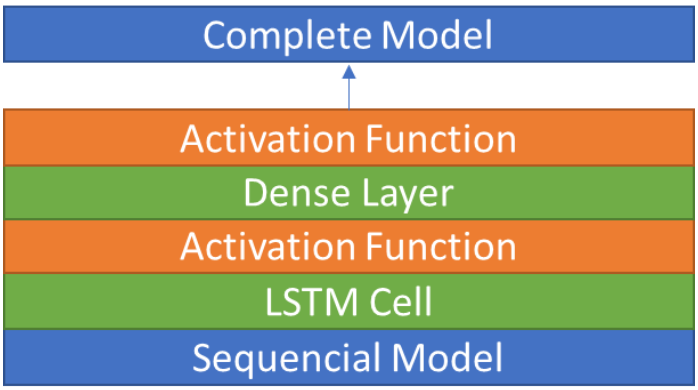


Figure 2-10 LSTM Network in Keras

After all the layers are added to the sequential model, the user needs to compile the model by calling the method compile. This will lock and build the model with all the specifications provided. If any change is desired after the method compile is called a new sequential model must be constructed. Finally, for the model to be trained, a fit method must be called using as a parameter the desired optimizer function. Among the optimizers included in Keras are [18]:

- Stochastic gradient descent
- RMSProp optimizer
- Adagrad optimizer
- Adadelta optimizer
- Adam optimizer
- Adamax optimizer
- Nesterov Adam optimizer

Another interesting tool present in the Keras API are the function callbacks. These functions are user defined function designed to monitor and take action during the training stage. One example of these callbacks function is the Early Stopping function. This function stops the

training after the selected metric stops improving during the training phase. Early Stopping is one of the fundamental methods to avoid overfitting problems with neural networks.

# CHAPTER 3

# EXPERIMENTAL FRAMEWORK DESIGN

## 3.1 Core Simulation Framework

For this thesis work a simulation based framework was constructed. Everything was coded using python and the intention behind this was to present a ready to use simulation framework to design and test cognitive radio systems and their interactions with a pre-established network of Conventional Radios. The framework also allows the user to verify the cognitive radio interaction with other cognitive devices. It also allows the user to quickly prototype and test cognitive radio algorithms for opportunistic spectrum access and look and evaluate the performance of the system within a controlled environment. Another excellent feature of the python based design is that allows us to reuse the code and implement directly on physical software defined radio devices [5].

In the design, a simulation class was created in order to be the referee or overseer of the RF spectrum. All radios, conventional and cognitive, must register to the simulation class and possess a generate method for the simulation to take place without errors. All the generate methods receive as an input the current status of the RF spectrum, the spectrum sensing is considered accurate, and the method must return the next state of the RF spectrum with the inclusion of the radio activity. Hence, conventional radios mark their behavior in the generate method while cognitive radio studies the current state and make optimal predictions for the next state.

The init method present on the simulation class instantiates a new simulation environment. The input parameters allow the user to specify the details of this environment. The input parameters are:

- Number of Channels: The number of channels refers to the number of frequency bands to be present in the experiment.
- Granularity: Refers to the minimum time step in the experiment. This could be seconds, minutes or hours.
- Cycles: Cycles refers to the total number of time steps that the simulation will last.

The Add Radio and Add Cognitive Radio methods allow the radios to register with the simulation class. The only difference between these two methods is that for Conventional Radios, a channel is required. The channel indicates which channel is reserved for that particular primary user.

After all the radios are registered, the run simulation method begins the simulation by calling all the generate methods from the registered radios. After every radio reports its activity in the spectrum the simulation frameworks begin to evaluate the spectrum, looking for clashes or interference between radios and computing the network efficiency. Network Efficiency is defined by the percentage of used time slots over the total slots available. A confidence metric is computed as well, which in the network is defined as the number of non-colliding blocks over the total blocks present in the network.

## 3.2 Conventional Radio

The conventional radio class represents the primary users in the simulation framework. The initialization of this class has two input parameters: the first one is the period of the transmitted signal. As primary user, all conventional radios are designed to have a deterministic period. For the especial case of the negative period value, it means that the radio transmits all the time on the licensed channel. The second parameter is the total amount of cycles that the simulation will last. The generate method included in the conventional radio class reports the periodic activity on the licensed channel to the simulation class.

## 3.3 Cognitive Radio v1

The Cognitive Radio v1 is the class designed to simulate the results from the previous work iteration of the cognitive radios research in the Autonomous Lab at UNT [19]. The init method for this cognitive radio takes the total number of channels as input. With the total number of channels to list are created, the free channels and the busy channels. These lists are updated at each time step during the simulation run.

The generate method for this class takes the current state and evaluates each time step, computing the free available channels and the busy channels. Then, the cognitive radio v1 will transmit in the next available frequency slot until the next round of the simulation is delivered. After all the time steps are analyzed, the class reports to the simulation frame the cognitive radio activity.

## 3.4 Cognitive Radio v2

The Cognitive Radio v2 is the current thesis work presented in this paper. Cognitive Radio v2 uses a LSTM neural network for each frequency channel in order to predict the spectrum activity in the next time slot. This way it can safely report activity based on future predictions, avoiding collisions in the next time frame. The behavior of this cognitive device is different from the rest of the devices. While the other radios start reporting activity from the very first time slot, this cognitive device first hears and learns from the environment without predicting nor acting. After enough data is collected, the cognitive device starts by separating the data into a validation set and a training set.

As well as the other radios coded in the framework, the init method provides an instance of the Cognitive Radio v2 class. The init method has the following parameters:

- Spectrum: this refers to the current state of the RF spectrum
- Train: The train parameters indicates the percentage of the information that is going to be used as training data, by default this variable is 0.7
- Lookback: Lookback refers to how many time steps to the past are going to be used as input to the LSTM network.

On the other hand, the generate method does not take any input parameters. However, multiple steps are taken before the networks generate the radio activity on the RF spectrum. The first step taken is to transform the current state information of the spectrum to a binary state. Every other user in the network is considered a primary user, even other secondary users. Therefore, if there is at least one signal present on the frequency band, that band is considered

busy in that time period. If no signal is detected on the frequency band, then the channel is considered available. An example of this data transformation is shown on Figure 2.5.

After the spectrum information is transformed into binary data, a shifted copy is created. The shift moves the spectrum one time slot into the future, and this copy of the data will work as the target data. Then the data is split into training and test data. As this is a time series problem, the data is split in a time-sensitive way. The amount of training data is given by the train parameter defined in the init method, e.g. if train is equal to 0.7, then the first 70% of the data is taken as the training set while the later 30% of the data is taken as the test set.

Once the training set is created, it needs to be reshaped to fit into the LSTM cells. The LSTM cells takes 3-dimentional data as input. For LSTM the input format is the following:

$$[samples, timesteps, features]$$

Where *samples* refers to the number of sequences present in the training set, *timesteps* refers to how many elements the input sequence has and finally the *features* refers to how many columns or features the input contains. This features dimension allows the user to use multiple sequences to generate one single sequence.

Take the sequence 1 through 10 as example, with a lookback of 3 timesteps. The desired shape of the data to feed to the LSTM cell is [7,3,1]. Figure 3.1 shows the detailed process on how the data is transformed to get the desired sequences.

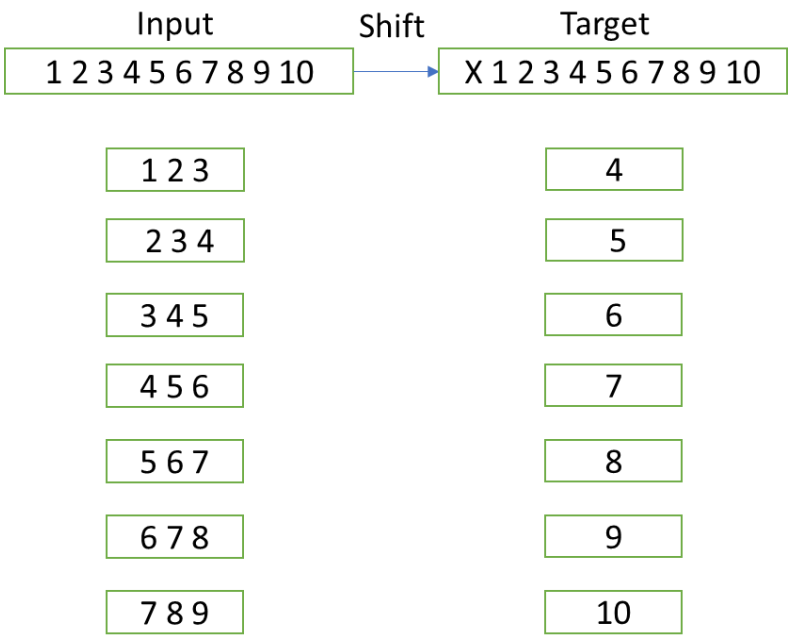


Figure 3-1 Data Reshape for Lookback = 3

After the data is split and in the correct format, it is time to define the LSTM network. Figure 3.2 shows the architecture of the LSTM network used in this thesis work. Five LSTM cells in a single layer is enough to decode simpler periodic sequences as the ones present in the primary users. One dense layer, or vanilla perceptron layer, is placed just after the LSTM cell with a Sigmoid activation function to produce the probabilities of the two classes present in the data (busy channel or free). Finally, the model is compiled using the binary cross entropy loss function, the adam optimizer and the metric to optimize is the accuracy of the LSTM network.

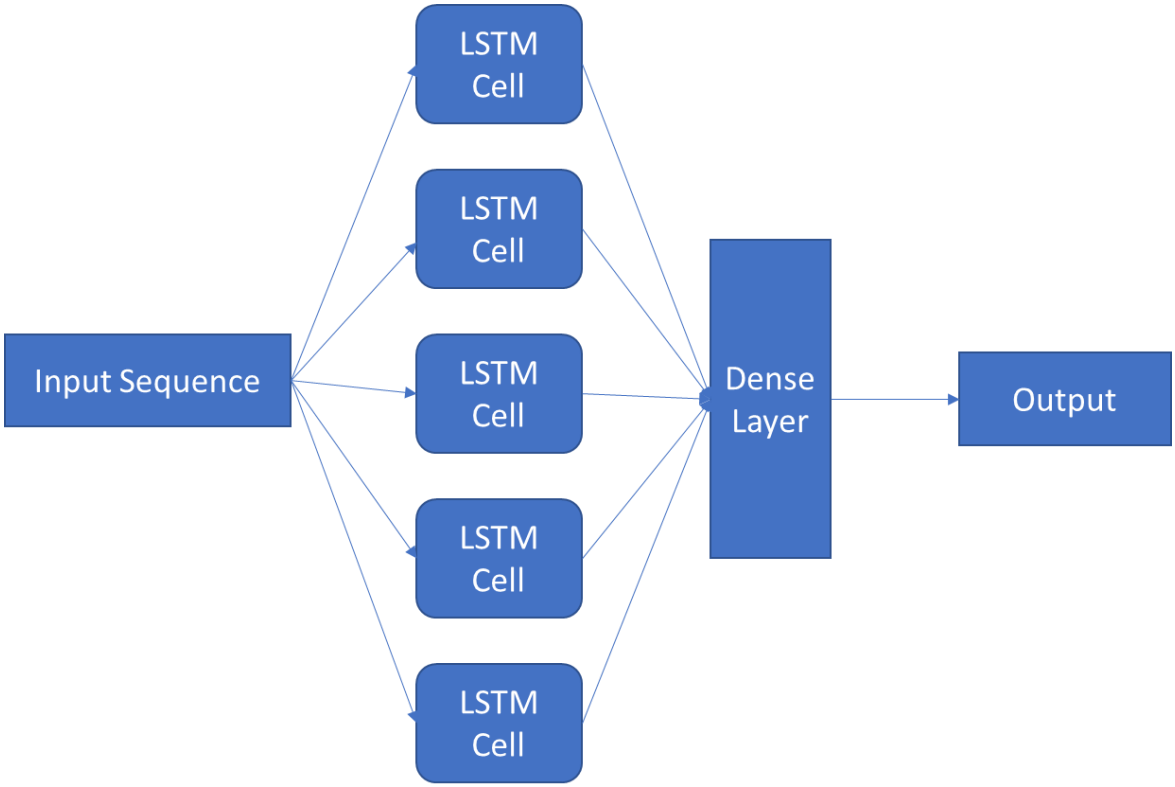


Figure 3-2 LSTM Model Network

After the model is constructed the network is trained for 10 epochs, with a batch size of 1. This means that the network weights and the loss function are updated after each sample. Finally, when the network is trained predictions are made over the test set and the predicted state activity is saved. With the predicted spectrum activity, a simple allocation technique is used and the radio activity is reported back to the simulation class. This process is repeated for each frequency channel.

# CHAPTER 4

# EXPERIMENTAL RESULTS AND CONCLUSION

## 4.1 Simulation and Experiment Overview

The objective of this thesis work is to simulate a Cognitive Radio device that is able to predict the Radio Frequency Spectrum usage by using Machine Learning methods and LSTM neural networks. For this objective, the simulation framework was designed in Python language to simulate and evaluate the performance of the cognitive device and its interaction with an established conventional radio network. In addition to conventional radio networks, this work evaluates the performance of the cognitive device with the presence of other cognitive devices and finally with the presence of a random radio source.

The cognitive radio device powered by Long Short Term Memory neural networks was designed to follow the design requirements listed below:

- Every other radio present in the spectrum is to be considered a primary user.
- Collision or interference with a primary user is forbidden.
- A training period is to be decided by the user. In this period, no activity is reported to the simulation core class. Meanwhile, all the network activity present in the RF spectrum is recorded to be used as training data.
- After training is complete, the cognitive device will predict and report activity in the next time slot. In each time slot, the current state is used to make future predictions.

With these design requirements, the Cognitive Radio device with LSTM neural network was designed. It is important to notice that the radio device will only start transmitting information after the training period is over. Another fundamental fact is that the future predictions are based on detected energy activity on the network and not from past predictions.

Finally, in order to get the response of the cognitive radio device on possibly the worst-case scenario, the cognitive radio device is required to transmit on all time slots after the training period is over. The cognitive radio will not produce activity only when there are no predicted available frequency bands in the next time slot.

## 4.2 Experiment Parameters

With two cognitive radio solutions explored in this work, all the experiments performed have the same runtime parameters. All experiments consist of a simulation of 15 days' worth of information with 10 available frequency bands and the simulation core class performs an evaluation with a granularity of 1 hour. This means that each time slot refers to 1 hour of network activity.

All conventional radios have a periodic behavior with a maximum period of 12 hours. This means that, for conventional radios, all days are the same. In the simulation every frequency band has an assigned primary user, a conventional radio with a random periodic behavior. Broadcasting stations might be also present in the simulation. These broadcasting stations are a special case of conventional radios with the small difference that they are constantly using the frequency band.

The performance of the cognitive radio models and the overall performance of the network are measured by two custom metrics: Network Confidence and Spectral Frequency

Efficiency. The Network Confidence metric refers to how confident the simulation class is of the cognitive radios' ability to avoid interference with primary users. The Network Confidence formula is shown below:

$$Network\ Confidence = \frac{\#\ non - coliding\ blocks}{\#\ of\ total\ blocks}$$

On the other hand, the Spectral Frequency Efficiency is the metric of usage on the RF spectrum by the network. The formula is shown below:

$$Efficiency = \frac{\#\ used\ blocks}{\#\ available\ blocks + \#\ used\ blocks}$$

Each experiment is done with 10 Monte Carlo walks where the behavior of the conventional radios are picked at random. The following list summarizes all the experiments performed.

- Experiment A: Baseline Activity with no cognitive radio devices present in the network
- Experiment B: Network Activity with a single Cognitive Radio v1 present in the network
- Experiment C: Network Activity with a single Cognitive Radio v2 present in the network
- Experiment D: Network Activity with both Cognitive Radio v1 and Cognitive Radio v2 present in the network
- Extra Experiment: Network Activity with a single Cognitive Radio v2 and a true random source present in the network

4.3 Experiment A: No Cognitive Device

This experiment serves as the baseline to the rest of the experiments. Only primary users are present in the spectrum and the network confidence is 100%, since there are no collisions on the spectrum. Figure 4.1 shows the waterfall type diagram of the spectrum activity.

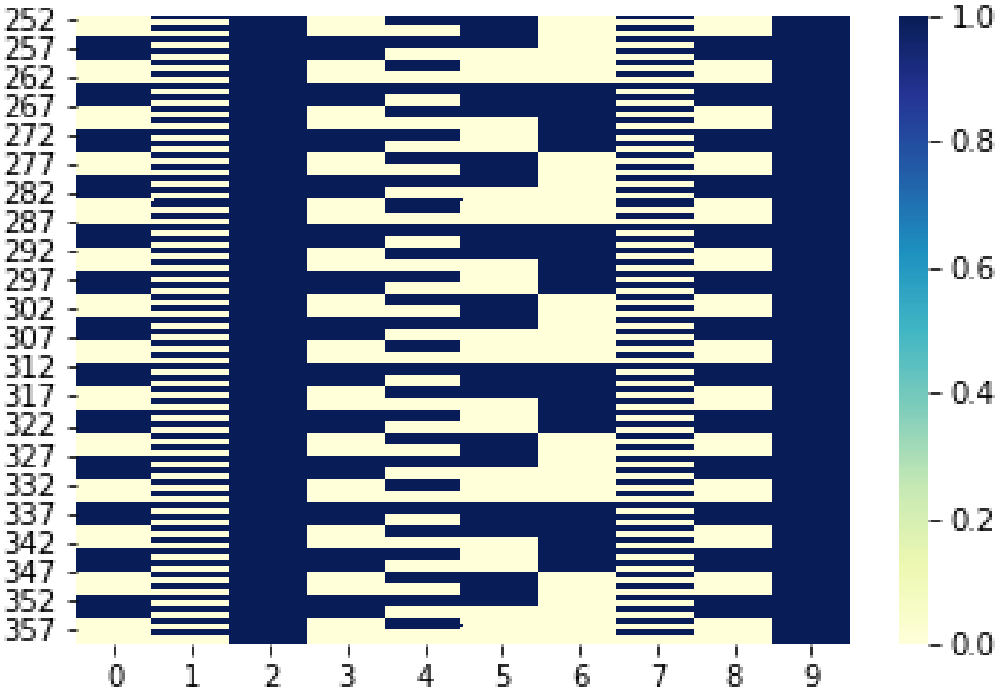


Figure 4-1 Experiment A: Spectrum Activity

As we can see in the figure, each primary user has a periodic random behavior. Two broadcasting stations are located in channel 2 and channel 9 as we can see continuous use of those channels.

4.4 Experiment B: Cognitive Radio v1

This experiment is the simulated result of the cognitive radio previously built on [19]. Figure 4.2 presents the same waterfall type diagram of the spectrum activity:

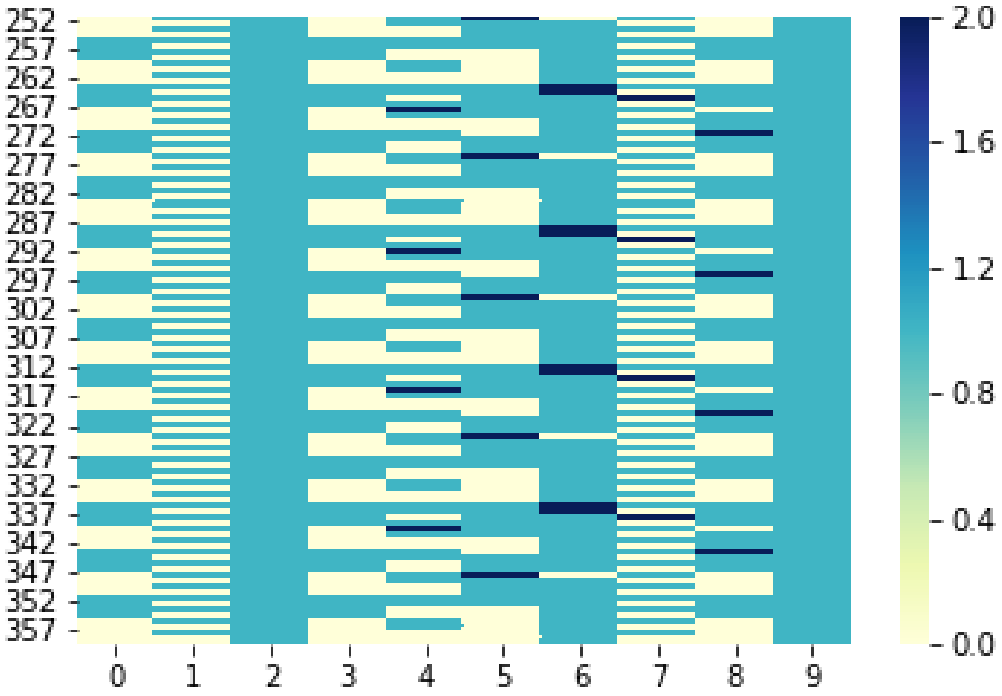


Figure 4-2 Experiment B: Spectrum Activity

In this experiment we can visually see some of the collisions happening in the spectrum. When the Cognitive Radio v1 detects the presence of the primary user in the channel, it jumps to a different available frequency band. However, since the simulation class takes the report of each radio once every hour, we can see the collision before the cognitive radio changes channels. On Figure 4.3 is shown the Network Spectral Efficiency and the Network Confidence for the 10 Monte Carlo runs with the Cognitive Radio v1 device.

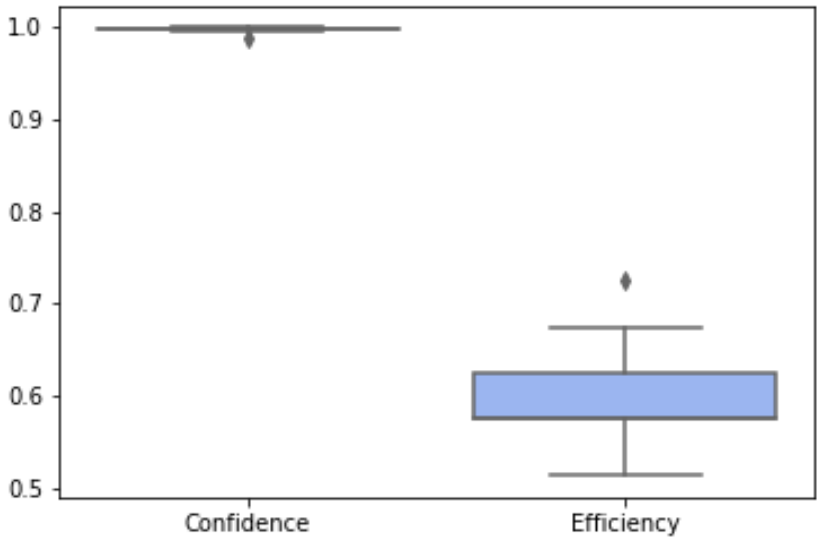


Figure 4-3 Experiment B: Confidence and Efficiency Plot

As we can see, even though we are able to see the collision on the network activity graph, we still get around 97% of Network Confidence. This means that interference with primary users only occurs around 3% of the time. On the Efficiency side we see a grand variation from 50% to even 70%. This is mostly due to the range of primary users' behaviors.

## 4.5 Experiment C: Cognitive Radio v2

For this experiment, Figure 4.4 shows the network activity. We can see that there are still a few collisions with the primary users but considerably less than the previous cognitive radio device. It is important to notice that this network activity and the metrics are only computed on the test data, after the training phase is completed.

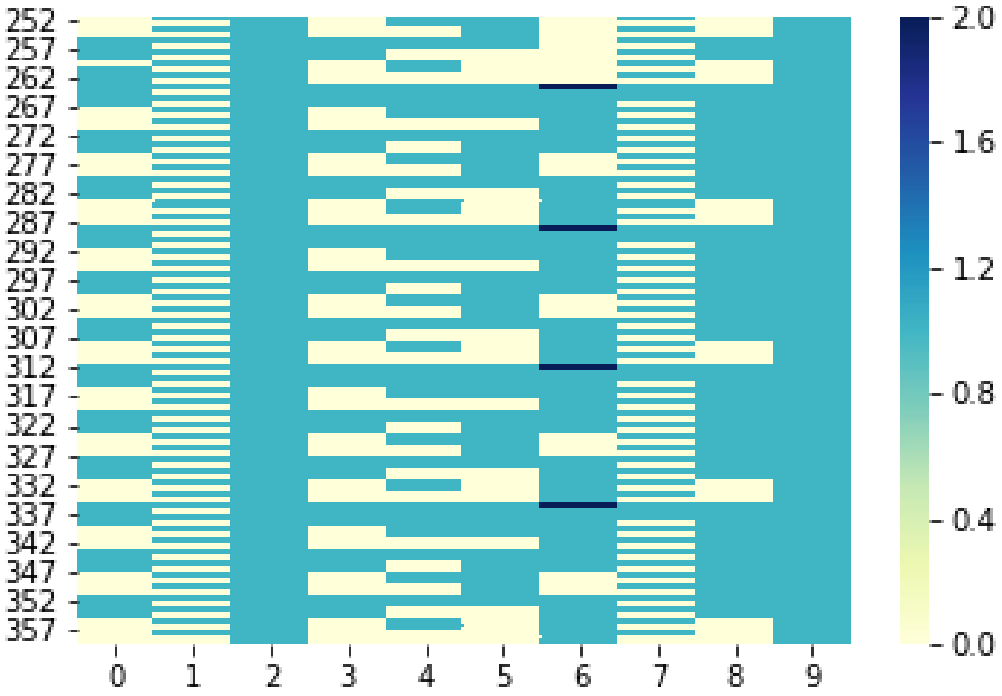


Figure 4-4 Experiment C: Spectrum Activity

Figure 4.5 shows the accuracy of the LSTM models per epoch. As expected the training accuracy is slightly higher than the test accuracy. However, we can see that they both follow the same pattern and reach almost 100% by epoch 7. We can also notice how the learning flattens after epoch 9, which leads to the conclusion of the network not learning much after epoch 9, hence no need to increase the number of epochs. On the other hand, Figure 4.6 shows the binary cross-entropy loss per epoch. As well as the accuracy graph, both training and test values follow the same pattern, which is a good indicator to avoid overfitting the network.

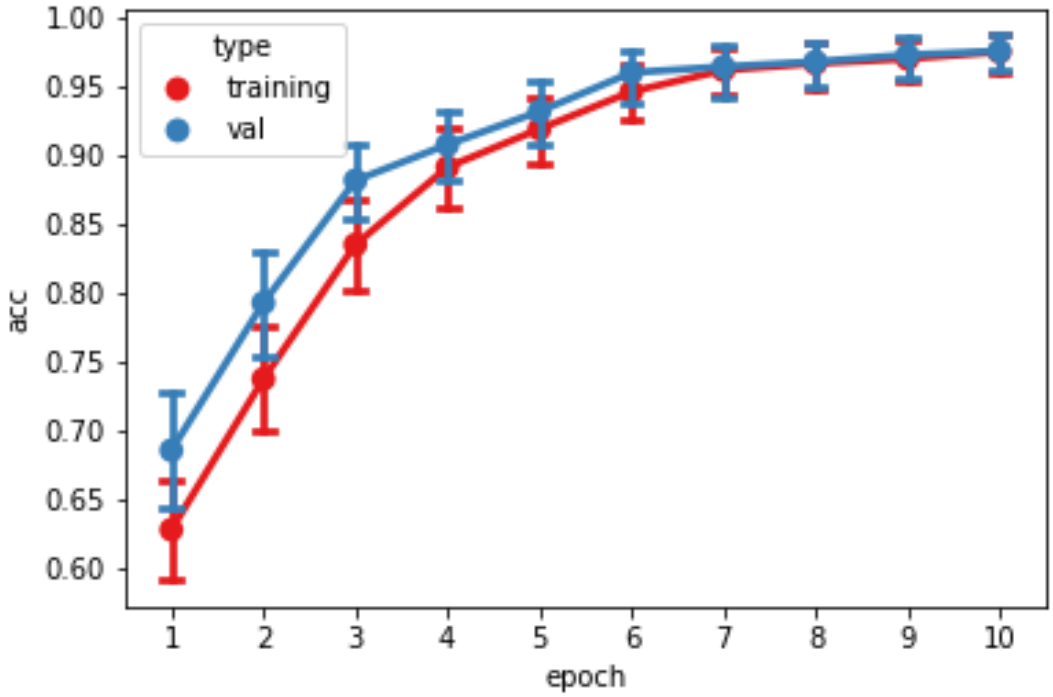


Figure 4-5 Experiment C: Training and Test Accuracy

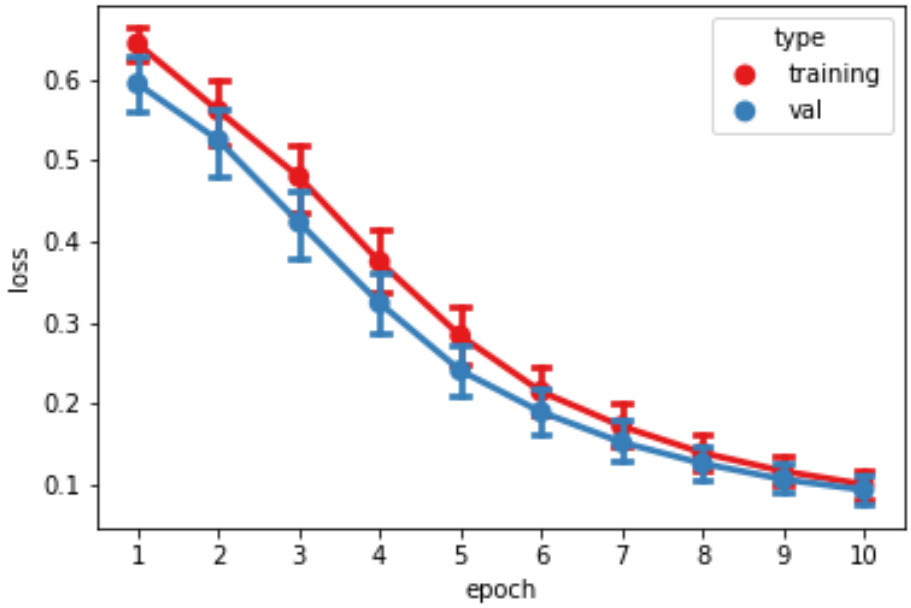


Figure 4-6 Experiment C: Training and Test Loss

Figure 4.7 shows the Efficiency and Network Confidence metrics. We can notice how the Confidence is over 99% and the spectral efficiency slightly increased in comparison with experiment B.

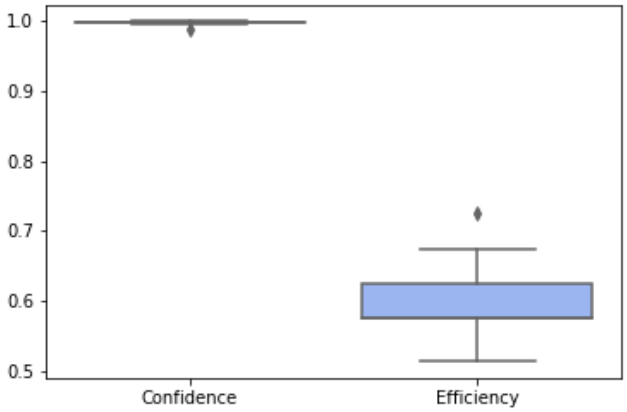


Figure 4-7 Experiment C: Confidence and Efficiency Plot

4.6 Experiment D: Cognitive Radio v2 and v1

The goal behind this experiment is to see how the Cognitive Radio device responds to another cognitive device without a periodic behavior participating in the same network. Same as before, Figure 4.8 shows the network activity diagram.

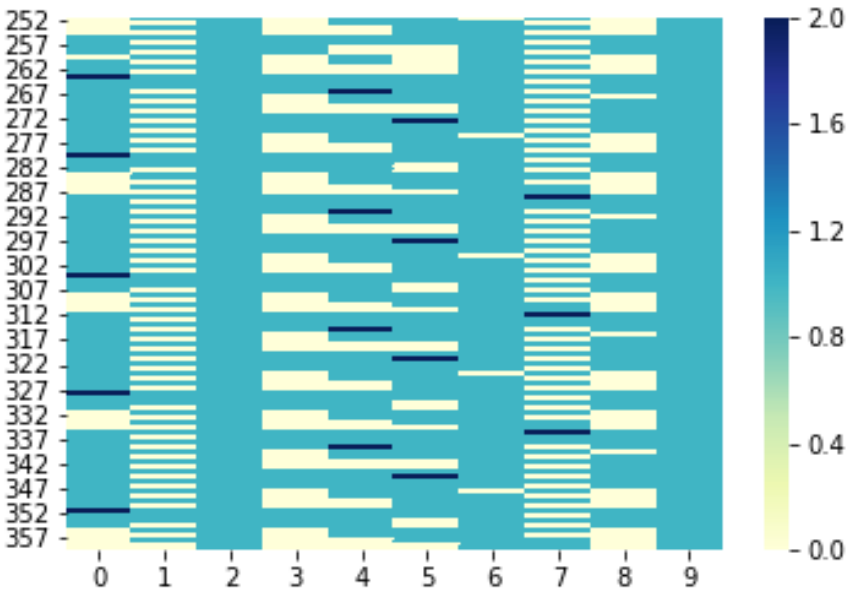


Figure 4-8 Experiment D: Spectrum Activity

Figures 4.9 and Figure 4.10 show the Accuracy and the Loss on each epoch. As expected, the accuracy is slightly lower than Experiment C. However, the graph still shows signs of improving every epoch so it would be safe to assume that the accuracy will keep improving with more epochs.

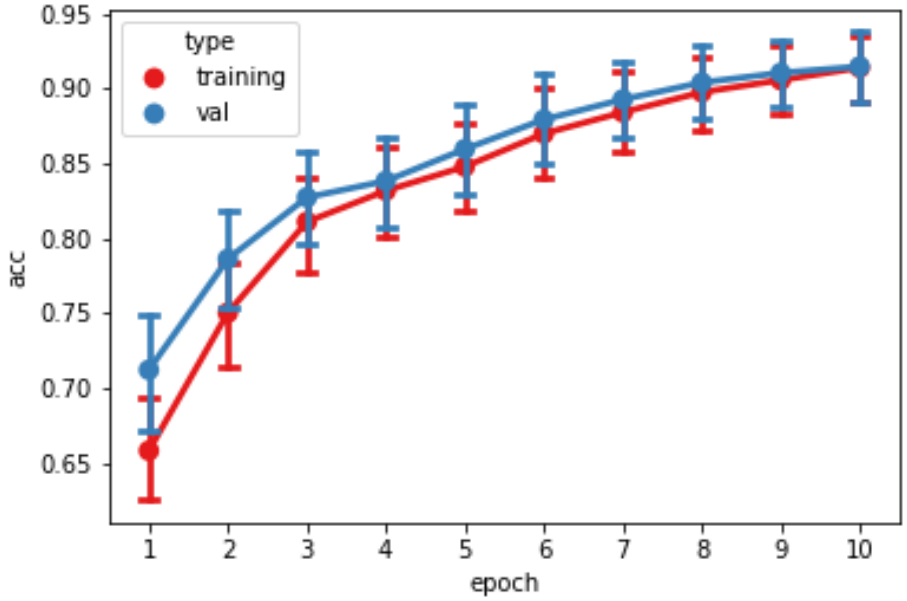


Figure 4-9 Experiment D: Training and Test Accuracy

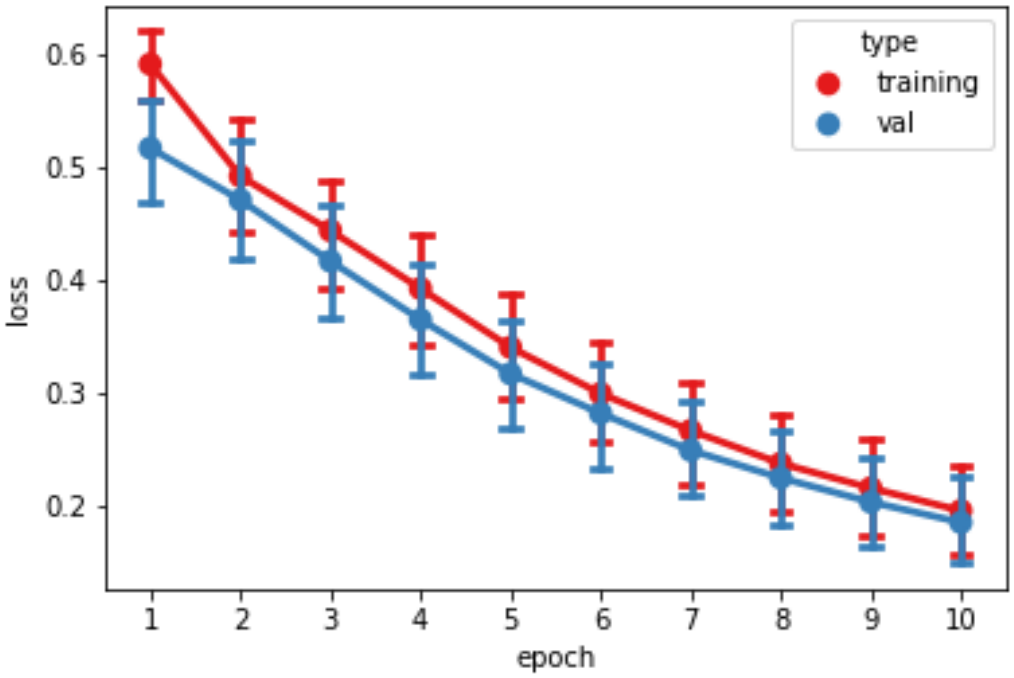


Figure 4-10 Experiment D: Training and Test Loss

Figure 4.11 presents the Efficiency and the Network Confidence of Experiment D. We can see slight improvement on the network efficiency, which is due to the fact that there are currently 12 radio devices in the network. We can also see that we have a wider range of Confidence levels around 99%.

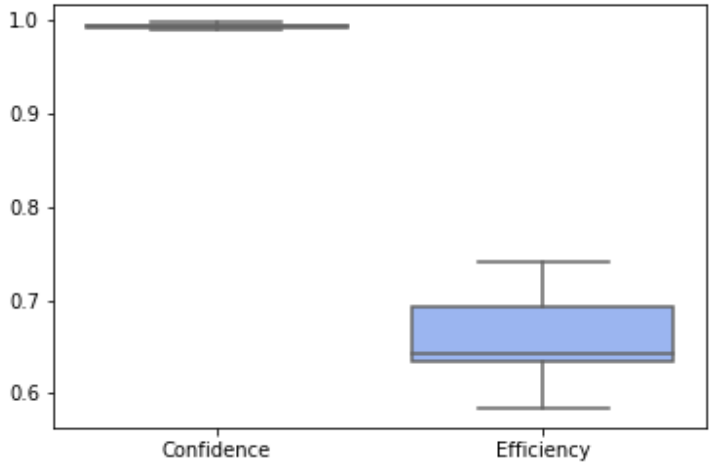


Figure 4-11 Experiment D: Efficiency and Confidence

## 4.7 Experiment E: Cognitive Radio v2 with Pure Random Sources

The goal of this experiment is to see how the Cognitive Radio device powered by LSTM neural networks responds to a network of pure random sources as primary users. Figure 4.12 shows the network's activity for this experiment. As predicted, the LSTM neural networks perform poorly in the presence of true random sources. Hence, is it necessary that the conventional radio have a pattern that the LSTMs can learn. Another evidence of the poor performance of LSTM neural networks with pure random sources can be seen in Figure 4.13 and Figure 4.14.

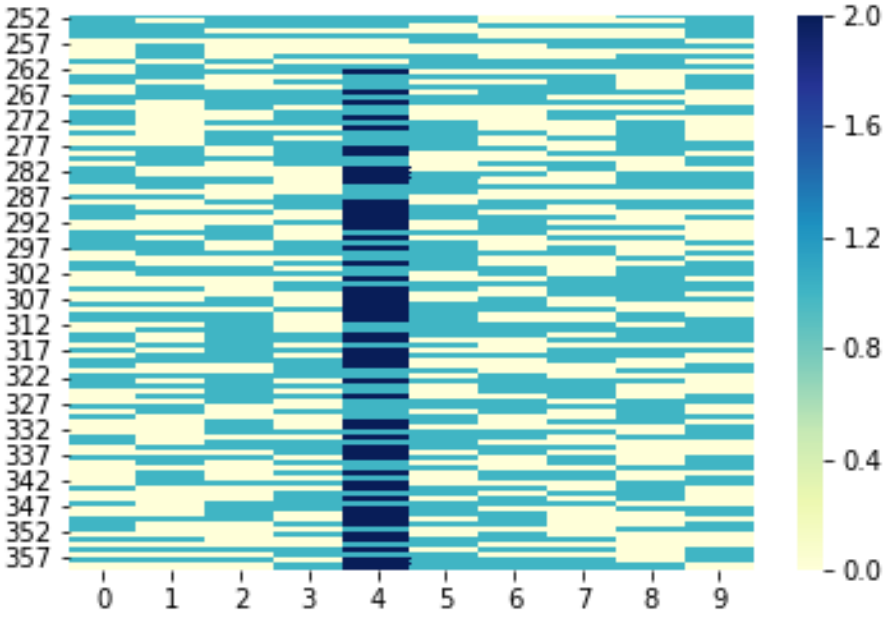


Figure 4-12 Experiment E: Network Activity

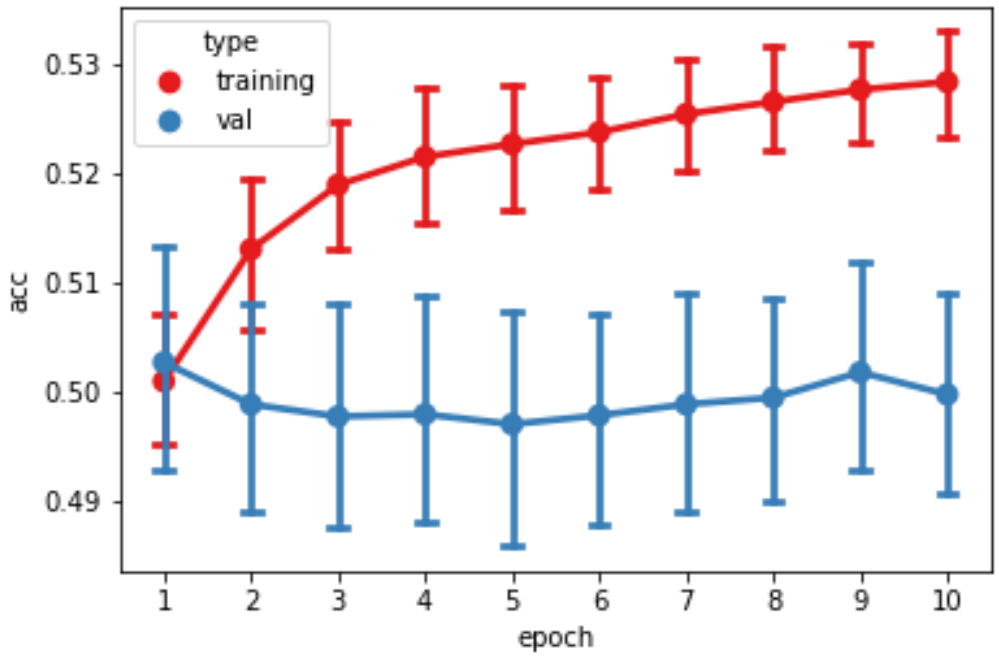


Figure 4-13 Experiment E: Accuracy on Training and Test Set

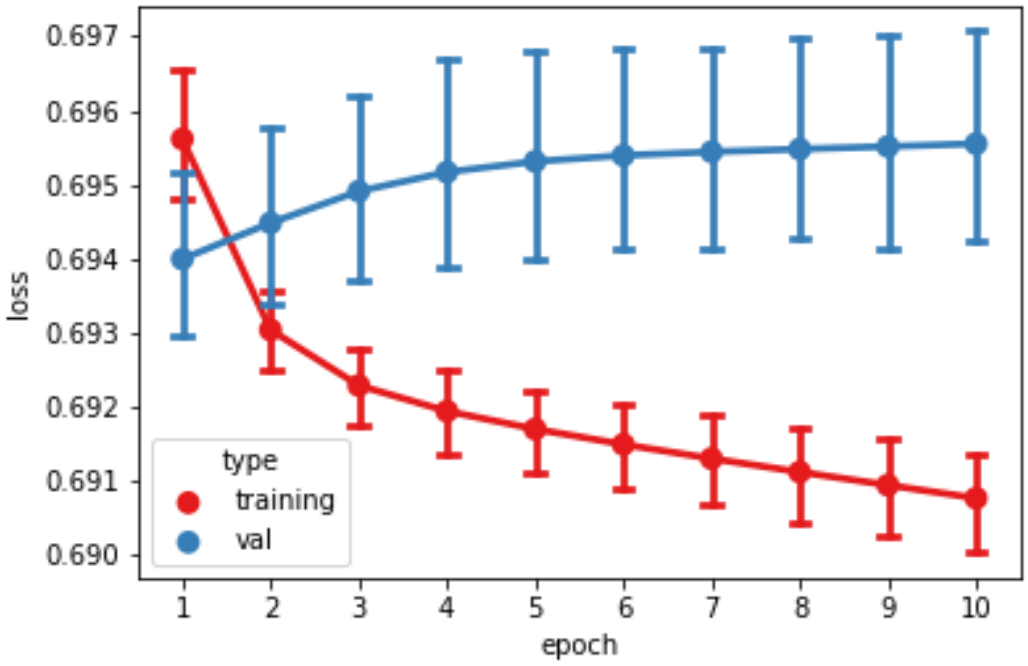


Figure 4-14 Experiment E: Loss on Training and Test Set

## 4.8 Comparison between Cognitive Radios

In this thesis work we have shown two implementations for cognitive devices. The first one, described in [19], is the simplest implementation of all the cognitive radio tasks: signal decoding, spectrum sensing via energy detection and trivial spectrum allocation. On the other hand, the second implementation uses LSTM neural networks to improve the performance of the cognitive device by predicting the spectrum usage in the next time slot. The principal difference between the two of them is that the first one does not need any time for adapting to the RF spectrum while the second one needs a training period before it can start transmitting in the spectrum. Table 4.1 summarizes the differences between the two cognitive radios shown in this thesis work.

Table 4.1 Comparison Between Cognitive Radios

| **Cognitive Radio v1** | **Cognitive Radio v2 with LSTMs** |
|---|---|
| Able to transmit instantly after joining the network | Needs a training period to adapt and learn the behavior of the PUs in the network before starts transmitting |
| Collisions depends on the behavior of the PU | Able to avoid collisions with a confidence higher than 99% |
| No need to recalibrate | Retraining or recalibration is recommended after extended periods of time or when accuracy drops |
| Fixed spectrum sensing technique | Can work in conjunction with other types of spectrum sensing |

## 4.9 Conclusion

This thesis work presents the implementation and design of two Cognitive Radio devices in conjunction with a simulation framework to develop and evaluate the performance of those cognitive devices on an existing network with conventional primary users. With the LSTM solution we managed to improve the spectral efficiency of the previous cognitive radio device from 95% to higher than 99%, avoiding almost all the spectrum collisions between secondary users and primary users.

Long Short Term Memory neural networks are being used in all kinds of time series applications, which make them special to tackle the spectrum usage problem as it was redefined

as a temporal problem. Even though they can predict behavior from primary users and other secondary users, these need to have a periodic pattern in their spectrum activity. True random sources are by definition unpredictable; therefore, the LSTM networks are unable to predict behavior with primary users with random behavior.

Finally, this prediction method using LSTM can be used in conjunction with any type of spectrum sensing technique available already. Energy Detection or Waveform methods can generate the current state of the spectrum to be used as an input to the LSTM. Cooperative sensing techniques can also be applied in conjunction with LSTMs. The cognitive devices report the spectrum activity to a central node where the LSTM can use the reported activity as input and predict the future usage of the spectrum.

## 4.10 Future Work

With the number of wireless devices like IoT devices increasing every day, the future of the research on cognitive devices is due to increase to satisfy the need of available frequency bands in an already underutilized RF spectrum. In this thesis work we showed an initial calibration of the LSTM based cognitive radio. However, it is possible to recalibrate the LSTMs by retraining the networks when the accuracy on the predictions drops, due to the addition of new radio devices to the network. But, it is also possible to keep training the network indefinitely. This is known as online learning and it might be a good fit for LSTM networks predicting the RF spectrum as it can adapt quickly to changes in the network architecture.

The next step in this research will be to apply a more complex allocation technique, where the network is mostly composed of secondary users with a few primary users and each secondary

user has preferred frequency bands on which to transmit and has a periodic active state, which means that the secondary is not transmitting all the time but instead only transmits every now and then. This way the LSTM will have to learn not only the behavior of the primary user but the preferred channels of each secondary user to provide a better performance of the network.

REFERENCES


[1] H. D. Lüke, "The Origins of the Sampling Theorem," *IEEE Communications Magazine,* pp. 106-108, 1999.

[2] "General Survey of Radio Frequency Bands 30 MHz to 3 GHz," 23 September 2010. [Online]. Available: http://www.sharedspectrum.com/wp-content/uploads/2010_0923-General-Band-Survey-30MHz-to-3GHz.pdf.

[3] E. Axell, G. Leus, E. G. Larsson and H. V. Poor, "State-of-the-art and recent advances Spectrum Sensing for Cognitive Radio State-of-the-art and recent advances," *IEEE signal processing magazine,* vol. 29, no. 3, pp. 101-116, 2012.

[4] R. Tucker, "naund bladeRF Overview," 16 September 2013. [Online]. Available: http://drop.hoopycat.com/bladerf_overview_web.pdf. [Accessed 21 January 2017].

[5] "What is Software Defined Radio?," [Online]. Available: http://www.wirelessinnovation.org/. [Accessed 21 January 2017].

[6] "Naund bladeRF Software Defined Radio," Naund . [Online].

[7] "Ettus Research - The Leader in Software Defined Radio," Ettus Research LLC, [Online]. Available: https://www.ettus.com/. [Accessed 22 January 2017].

[8] T. Yucek and H. Arslan, "A Survey of Spectrum Sensing Algorithms for Cognitive Radio Applications," *IEEE COMMUNICATIONS SURVEYS & TUTORIALS,* vol. 11, no. FIRST QUARTER 2009, pp. 116-129, 2009.

[9] Y. Zeng, Y.-C. Liang, A. T. Hoang and a. R. Zang, "A Review on Spectrum Sensing for Cognitive Radio: Challenges and Solutions," *EURASIP Journal on Advances in Signal Processing,* p. 15, 2010.

[10] M. Song, C. Xin, Y. Zhao and X. Cheng, "DYNAMIC SPECTRUM ACCESS:FROM COGNITIVE RADIO TO NETWORK RADIO," *IEEE Wireless Communications,* vol. 19, no. 1, pp. 23-29, February 2012.

[11] A. Goldsmith, S. A. Jafar, I. Maric´ and S. Srinivasa, "Breaking Spectrum Gridlock With Cognitive Radios: An Information Theoretic Perspective," *Proceedings of the IEEE,* vol. 97, no. 5, pp. 894-914, May 2009.

[12] M. S.L.Ho, "The use of ARIMA models for reliability forecasting and analysis," *Computers & Industrial Engineering,* vol. 35, no. 1-2, pp. 213-216, 1998.

[13] R. Nau, "Statistical forecasting: notes on regression and time series analysis," Deku University , [Online]. Available: http://people.duke.edu/~rnau/411arim.htm#arima010. [Accessed 5 August 2017].

[14] L. E. Baum and T. Petrie, "Statistical Inference for Probabilistic Functions of Finite State Markov Chains," *The Annals of Mathematical Statistics.,* p. 1554–1563, 1966.

[15] L. R. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," *Proceedings of the IEEE,* p. 257–286, 1989.

[16] C. G. C.W. Omlin, "Constructing Deterministic Finite-State Automata in Recurrent Neural Networks," *Journal of the ACM,* pp. 937-972, 1996.

[17] C. Olah, "colah's blog," 27 August 2015. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed 24 January 2017].

[18] Keras, "Keras Documentation," Keras: The Python Deep Learning library, [Online]. Available: https://keras.io/. [Accessed 24 January 2017].

[19] K. A. A. Bhadane, *A Cognitive Radio Application through Opportunistic Spectrum Access,* University of North Texas , 2017.

[20] "Spectrum Policy Task Force," November 2002. [Online]. Available: http://sites.nationalacademies.org/cs/groups/bpasite/documents/webpage/bpa_048826.pdf.

[21] "Federal Communications Commission Spectrum Policy Task Force," 15 November 2002. [Online]. Available: https://transition.fcc.gov/sptf/files/SEWGFinalReport_1.pdf.

[22] H. Tang, "Some physical layer issues of wide-band cognitive radio systems," *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005.,* pp. 151-159.

[23] "Introduction to GNU Radio and Software Radio," GNU Radio , [Online]. Available: http://gnuradio.org/redmine/projects/gnuradio/wiki/Guided_Tutorial_Introduction.

[24] "What is GNU Radio and why do I want it?," GNU Radio, [Online]. Available: http://gnuradio.org/redmine/projects/gnuradio/wiki/WhatIsGR.

[25] I. E. Igbinosa, O. O. Oyerinde, V. M. Srivastava and S. Mneney, "Spectrum Sensing Methodologies for Cognitive Radio Systems: A Review," *International Journal of Advanced Computer Science and Applications,* vol. 6, no. 12, pp. 13-22, 2015.

[26] R. A. Rashid, M. A. Sarijari, N. Faisal, S. Yosuf and N. H. Mahalin, "Spectrum Sensing Measurement using GNU Radio and USRP Software Radio Platform," in *The Seventh International Conference on Wireless and Mobile Communications*, 2011.

[27] "Spectrum Sensing," *End to End efficiency [E3 ] white paper,* Nov, 2009.

[28] J. Mitola III and G. Q. Maguire, "Cognitive Radio: Making Software Radios More Personal," *IEEE Personal Communications,* vol. 6, no. 4, pp. 13-18, Aug. 1999.

[29] F. A. Gers, J. Schmidhuber and F. Cummins, "Learning to Forget: Continual Prediction with LSTM," *Neural Computation. ,* 2000.

[30] S. Hochreiter, Y. Bengio, P. Frasconi and J. Schmidhuber, "Gradient Flow in Recurrent Nets: the Difficulty of Learning Long-Term Dependencies," *Field Guide to Dynamical Recurrent Neural Networks. ResearchGate. IEEE Press.*

[31] Tensorflow, "Tensorflow Getting Started," [Online]. Available: https://www.tensorflow.org/get_started/get_started. [Accessed 24 January 2017].