Primary User Activity Prediction in DSA Networks using Recurrent Structures

Debashri Roy*, Tathagata Mukherjee†, Mainak Chatterjee* Eduardo Pasiliao‡

* Computer Science
University of Central Florida
Orlando, FL 32826
{debashri, mainak}@cs.ucf.edu

† Computer Science
University of Alabama
Huntsville, AL 35899
{tm0130}@uah.edu

‡ Munitions Directorate
Air Force Research Laboratory
Eglin AFB, FL, 32542
{eduardo.pasiliao}@us.af.mil

***Abstract*—For unlicensed (secondary) users to opportunistically access the shared radio spectrum on a non-interfering basis, it is important that they are able to sense the transmission activities of the licensed (primary) users. However, spectrum sensing expend a considerable amount of energy and time, which can be reduced by reliably predicting the primary user activities. In this paper, we present recurrent neural network models which are able to accurately predict the primary users' activity in dynamic spectrum access (DSA) networks so that the secondary users can opportunistically access the unused spectrum. Using Universal Software Radio Peripheral (USRP) Software Defined Radios (SDRs), we collect over-the-air data from 8 primary users and train the learning models that we use in conjunction with a central spectrum sensor. We start by implementing two machine learning models: (i) traditional linear regression and (ii) neural network model using Long Short Term Memory (LSTM). These models are able to predict the primary users' activity with 75% and 97% accuracy respectively. To further improve the prediction accuracy, we exploit the spatio-temporal correlation in the collected data by implementing a Convolutional LSTM model– which achieves 99% accuracy for predicting the long-term activity of primary users. The experimental results demonstrate that the proposed models are able to successfully predict the primary users' activities, thereby reducing both the under-utilizations and interference violations in DSA networks.**

Keywords: Primary Users, Prediction Models, Long Short Term Memory, Convolutional LSTM, Spatio-temporal Property, DSA networks.

I. Introduction

The ever-increasing demands for spectrum access from different emerging wireless applications have made it necessary to better manage and utilize the radio spectrum. The successful deployment of such spectrum management approaches requires intelligent and adaptive systems, in order to accurately assess the radio environment such that unlicensed (secondary) users are able to opportunistically access the spectrum of licensed (primary) users when such spectrum is not in use, thereby increasing the spectrum utilization. However, such spectrum management and deployment practices must take into account the fact that spectrum access policies prohibit any *interference violation* by the secondary users (SUs) when primary users (PUs) use the spectrum. The Citizens Broadband Radio Service (CBRS) [1] is an example of a DSA implementation, where variety of commercial users share the 3.5 GHz band with incumbent federal and non-federal licensed users.

The Federal Communications Commission (FCC) [2] mandated that all SUs must release the occupied spectral bands as soon as any PU starts to transmit on that band, ensuring uninterrupted availability to the licensed users. To ensure this, the SUs must have knowledge about the spectrum availability. This awareness is typically achieved by sensing the transmission activities on the target spectrum bands using various techniques like: use of beacons, geolocation database, and local energy sensing at the receivers [3], [4].

In this paper, we focus on spectrum sensing performed at a central spectrum sensor (SS), as the use of a centralized SS has broaden applications and lower infrastructure costs [5].

We illustrate the various aspects of spectrum sensing in Fig. 1. The first row represents the PU's ON-OFF activity and the remaining rows show the SU's activities that include sensing and transmissions when PU is in the OFF state. Note that the first two strategies do not involve separate SS and we also assume that the SUs sense the spectrum continuously. The last two strategies involve the SUs deciding to access the spectrum based on the information obtained from a local centralized SS. Thus in the first case, a conservative strategy is imposed on the SU for use of the channel. The cognitive SU continuously senses the channel and whenever it finds that the channel is free, it transmits in the next timestamp, going back to the sensing state after that. The second strategy is also conservative but here the SU transmits in bursty mode. Both these conservative strategies aim to avoid interference violations; however, under-utilization is high. As for the third strategy, the spectrum sensors send information to secondary user whenever it senses the channel to be free. The secondary user transmits from one timestamp and consults the spectrum sensor for the next timestamp. Note that here we assume that the time taken to consult the spectrum sensor is less than the spectrum sensing time. This assumption coupled with the strategy minimizes the under-utilization of the previous two strategies. The fourth strategy is proposed in this paper. In this case, the spectrum sensor is intelligent and trained over the historical data of primary user activity. It can predict the primary use's activity for the next timestamp. The secondary user uses that prediction to transmit in bursty mode. It is evident from all the four strategies that long term prediction from the SS enables the SU to transmit efficiently over the shared channel by minimizing both the under-utilization and

978-1-7281-2376-9/19/$31.00 ©2019 IEEE

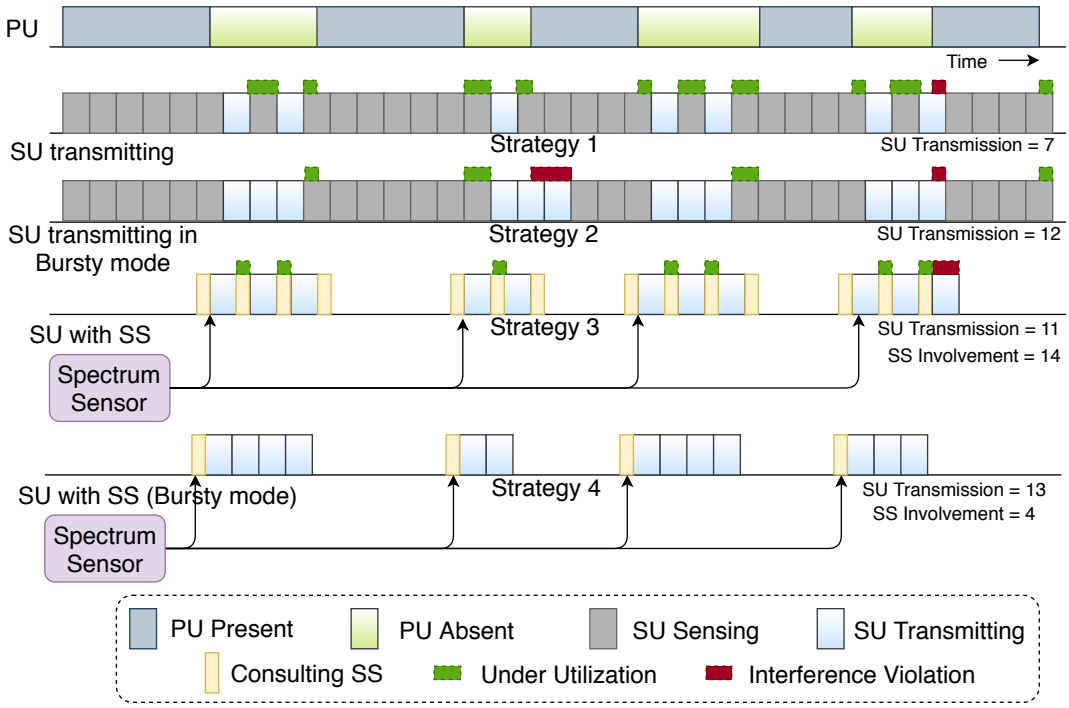


Fig. 1. Typical Spectrum Sensing Scenarios

the interference violations.

In this paper, we propose neural network models based on Long Short Term Memory (LSTM) and convolutional LSTM (ConvLSTM) to predict the primary user's future inactive time. We use both the LSTM and ConvLSTM based neural network models for exploiting the recurrent structure in historical over-the-air data from each primary user. To the best of our knowledge, this paper is the first one to propose a strategy for long term prediction of the activity of the primary user [6]. The main contributions of this paper are: (1) We propose three machine learning based models for long-term prediction of the primary user's activity. They are: (i) Linear regression; (ii) Recurrent neural network (RNN) with Long Short Term Memory (LSTM) cells; and (iii) RNN using Convolutional LSTM (ConvLSTM) cells. (2) Using a testbed, we record transmission activities of 8 software defined radios (USRP B210) [7], which are used as primary users. We collect over-the-air I/Q data from these radios using a RTL-SDR [8]. I/Q values are the Inphase (I) and Quadrature (Q) phase components of the signal. (3) A central spectrum sensing module is trained using the proposed models on the collected dataset for multiple epochs by minimizing the mean squared error over the training data. The trained models are then used by the spectrum sensor and used for long-term prediction of the activities of the primary user, to be used by the secondary users during the deployment phase. (4) We deployed the trained version of all the models in a spectrum sensor and predicted the shared channel availability for the secondary users with 75%, 97%, and 99% accuracy respectively for linear regression, LSTM, and ConvLSTM. Note that intuitively the ConvLSTM based model is able to achieve this high accuracy by exploiting the spatio-temporal correlation present within the recurrent structure of the collected I/Q data. (5) We also show through testbed evaluation that the proposed models can decrease interference violations by 0.2%, 99.3%, and 100% for linear regression, LSTM and ConvLSTM models, respectively. Under-utilizations are decreased by 98.9%, 99.5%, and 100% respectively, for the aforementioned models.

In the next section, we present a survey of the existing primary user prediction models followed by system model and problem formulation in section III and different prediction models in section IV. The testbed setup and experimental results are presented in section V. Conclusions are drawn in the last section.

II. Related Works

In this section, we discuss the main premise for using learning techniques for the primary user's activity prediction modeling. We also present some previous important work in this area that have used machine learning techniques.

Various traditional methods of spectrum sensing and their applications have been presented in [5]. The idea of cooperative spectrum sensing was presented as the solution to security related problems in the spectrum sensing domain. Though the concept of estimating spectrum usage in multiple dimensions such as time, frequency and space, was introduced, no prediction attempts were made. In [9], the authors presented opportunistic spectrum access based on a Markovian model that accounted for the bursty nature of ON/OFF traffic models. Through empirical results, the authors have shown that the selection of a channel for the secondary user will depend on the probability distribution of the primary user's traffic as well the elapsed OFF time. This paper established the feasibility of

using PU usage modeling for designing the spectrum access methods for SU bursty traffic.

There are few existing researches on spectrum sensing and prediction that use machine learning based techniques. An artificial neural network based approach for spectrum sensing in noisy environment was proposed in [10]. The simulation results accomplished better sensing reliability than traditional techniques for signals with low signal-to-noise ratio (SNR). A machine learning based spectrum prediction approach was proposed in [11]. A multi-layer perception (MLP) based neural network was used to predict the primary user's activity, the resulting model was validated through extensive simulations.

In [6], the authors give an analytical overview of various existing methods for spectrum prediction, that are based on: moving average, autoregressive models, hidden Markov models, Bayesian interference and static neighbor graph. It was established that all these models have limited scope for accurate long-term prediction. A deep cooperative sensing scheme was proposed by Lee *et al.* in [12]. A convolutional neural network (CNN) model was proposed to exploit both spectral and spatial correlation of individual sensing outcomes for multiple PUs and SUs. All these studies demonstrate the necessity of a practical long-term spectrum usage prediction method for the PUs that leverage machine learning techniques.

Our focus is on the effectiveness of recurrent neural networks [13] which have been used extensively for modeling *temporal data* such as speech [14]. There is limited amount of work that recognizes the potential of using recurrent structures in the Radio frequency (RF) domain. Though such use of I/Q data for building machine learning systems for communication has been limited in the past, recently it has been used in several applications [15]–[18]. RF data (being a time series data) has both spatial and temporal property within the I/Q components. However, to the best of our knowledge there is no recurrent neural network based application which exploits both the spatial and temporal property of the RF data and use that for a long-term prediction model for PU's presence or absence.

III. Problem Description

In this section, we lay out the assumptions, present the system model, and formulate the problem.

A. Primary User Activity Pattern

The primary users alternate between ON (busy) and OFF (idle) periods. Random variables r_{on} and r_{off} determine the duration of ON and OFF periods respectively. The probability distribution of r_{ON} and r_{off} depends on the specific activity pattern of the primary user. We assume that the primary user's ON and OFF times are independently and identically distributed and use a Poisson point process to model them. Thus we model the probability distributions for the ON and OFF times of the primary users as:

$$f_{on}(r_{on};\beta_{on}) = \begin{cases} \frac{1}{\beta_{on}} e^{-\frac{r_{on}}{\beta_{on}}} & r_{on} \geq 0 \\ 0 & r_{on} < 0 \end{cases} \quad (1)$$

$$f_{off}(r_{off};\beta_{off}) = \begin{cases} \frac{1}{\beta_{off}} e^{-\frac{r_{off}}{\beta_{off}}} & r_{off} \geq 0 \\ 0 & r_{off} < 0 \end{cases} \quad (2)$$

where, β_{on} and β_{off} are the mean ON and OFF times respectively.

We define the *activity factor* of the primary users as the ratio of the mean PU ON time to the sum of the mean of PU ON and OFF times and thus this is given by:

$$PU_{activity} = \frac{\beta_{on}}{\beta_{on} + \beta_{off}} \quad (3)$$

It must be noted that, it is not required to pre-define a model for learning; the proposed prediction models are able to learn any kind of PU activity pattern. We considered the Poisson process for our testbed experiments.

B. System Model

We assume that PUs and SUs co-exist in a geographical area. The SUs can utilize the spectrum bands when they are not being used by the PUs. The PUs have priority and can transmit when they need to. On the other hand, SUs always have backlogged traffic to transmit and must yield to the PU as soon as they require the spectrum. The transmission times for the PUs obey the pattern (distribution) as discussed earlier. As a result of this system model, prior knowledge of PU activity is vital for SUs to effectively and efficiently share the spectrum. We assume that there is a centralized spectrum sensor that monitors the PU transmission activities and maintains records of all past observations. The trained models corresponding to the proposed neural network architectures are co-located with the spectrum sensor and use the past observations as inputs in order to predict the activity pattern of the PUs.

C. Problem Formulation

The trained neural network model predicts the primary user's activity at time τ for the next Γ timestamps and the SS sends that information to the secondary user. Depending on the prediction information received from the spectrum sensor, the secondary user schedules its transmission for the next Γ timestamps, thus minimizing the interference violation as well as the under-utilization.

We denote the states of the primary user's and secondary user's transmission as s_{PU} and s_{SU}. The ON state of each is represented as 1, and OFF state as 0. We also denote tolerable thresholds for interference violation and under-utilization as I_{thr}, and U_{thr} respectively.

At time τ, the trained model predicts the PU activity for the next Γ timestamps. In our system we let the SS inform the secondary user when it sees continuous PU inactivity for more than 50% of the predicted Γ timestamps. Note that this is a *heuristic* for solving the underlying optimization problem as given by equation 4. However in the most general setting, at the time τ, depending on the information received from the SS, the secondary user sets its state s_{SU} to 0 or 1 for the

next Γ timestamps. Let $I_{ch}(t)$ denote the indicator variable at time t ($\tau < t \leq \tau + \Gamma$), indicating whether the secondary user causes interference to the primary user on channel ch. Note that: $I_{ch}(t) = s_{PU}(t) \times s_{SU}(t)$. Similarly, let $U_{ch}(t)$ denote an indicator variable representing the under-utilization of ch channel at time t. Notice that: $U_{ch}(t) = \overline{s_{PU}(t) + s_{SU}(t)}$, where $+$ denotes the logical *OR* operation. $s_{PU}(t)$ and $s_{SU}(t)$ represent the PU's and SU's states respectively at time t. We note that a conservative strategy for dynamic spectrum allocation will result in under-utilization, whereas an aggressive strategy will lead to a considerable amount of interference violation, as discussed in connection to Fig. 1.

The secondary user can transmit on the channel when the primary user is in the OFF state (i.e., s_{PU} = 0). It should terminate its transmission as soon as the primary user starts to transmit (i.e., s_{PU} = 1). However, while the SU is transmitting ($\tau < t \leq \tau + \Gamma$), it will not be aware of this state change of the PU (that is it will not be able to detect the change in the $I_{ch}(t)$ states). Thus the possibility of having accurate knowledge of $I_{ch}(t)$ will depend on how well the neural network can model the underlying probability distribution of the activity of the primary user.

Finally, our objective is to maximally assign the secondary user to the ON state constrained over the thresholds of interference violation (I_{thr}) and under-utilization (U_{thr}). Thus, the general problem can be formulated as follows: let the number of timestamps where the SU is active be denoted by η where η can takes values between $\tau + 1$ and $\tau + \Gamma$. Then the problem can be formulated as that of maximizing η subject to the constraints on the interference and under-utilization being bounded above by the respective thresholds. Thus we have:

$$\begin{aligned} &\max \eta \\ &\text{subject to} \sum_{t=1}^{\eta} I_{ch}(t) \leq I_{thr} \\ &\qquad\qquad \sum_{t=1}^{\eta} U_{ch}(t) \leq U_{thr} \end{aligned} \tag{4}$$

IV. Proposed RNN Based Prediction Models

We present two types of neural network models that leverage the temporal and spatio-temporal correlation in the historical data of the PU activities, in order to predict future PU activities. We take the I/Q values of over-the-air signal data as raw features for future state prediction, as the I/Q values do not require further sophisticated signal processing. Hence they are capable of providing an end-to-end solution for our problem using machine learning techniques.

A. Recurrent Neural Network Model

Fully-connected and convolutional neural networks that are traditionally used for deep learning lack the capability to exploit the context available in temporal data. Additionally, there is the problem of *vanishing gradients*, when trying to use *back propagation* with temporal data. Both these problems are addressed by Recurrent Neural Networks (RNN) [19] which we now describe in brief in the context of our problem.

1) Temporal Property of I/Q data: Given T training samples (for T timestamps) where each sample is a vector of size M, and each component of the vector is a tuple $(I, Q) \in \mathcal{C}$ representing a number in the complex plane, we represent this vector for a given time stamp t as $x_t = [[(I, Q)_i]^t; i = 1, 2, \cdots, M] \in \mathcal{C}^M$ where $t = 1, 2, \cdots, T$, and use it as an input to the neural network. We use a sample size (M) of 1024 as a default for our experiments. We want to find the probability of $s_{PU}(t) = 0$ for the next input vector (x_{t+1}) for $t = t+1$. The probability $P(s_{PU}(t) = 0 | x_{t+1})$ can be written as:

$$P(s_{PU}(t) = 0 | x_{t+1}) = \frac{P(x_t | s_{PU}(t) = 0) P(s_{PU}(t) = 0)}{P(x_t | s_{PU}(t) = 0) P(s_{PU}(t) = 0) + P(x_t | s_{PU}(t) = 1) P(s_{PU}(t) = 1)} \tag{5}$$

where $P(x_t | s_{PU}(t) = 0)$ and $P(x_t | s_{PU}(t) = 1)$ are the conditional probabilities of x_t given $s_{PU}(t)$ was set to 0 and 1 respectively. $P(s_{PU}(t) = 0)$ and $P(s_{PU}(t) = 1)$ are the marginal probabilities, values of which will depend on the activity factor of the primary user. The predicted value of vector x at the next timestamp $(t + 1)$ will depend on the predicted value of the current one [20], which is given by:

$$\tilde{x}_{t+1} = \underset{x_{t+1}}{arg\,max} (P(x_{t+1} | \tilde{x}_t)) \tag{6}$$

2) The LSTM Model: To exploit the temporal property, we first use the idea of LSTM cells as shown in Fig. 2. In one LSTM cell, there are typically (i) three types of gates: input (i), forget (f), and output (o); and (ii) a state update of internal cell memory (c). The most important part of the LSTM cell is the "forget" gate, which at time t is denoted by f_t. The forget gates decide whether to keep a cell state memory (c_t) or not. The forget gates are designed as per the equation (7) on the input value x_t at time t and output (h_{t-1}) at time $(t - 1)$. Note that W_{xf} and b_f represent the associated weight and bias respectively, between input (x) and the forget gate (f).

$$f_t = \sigma(W_{xf} x_t + W_{hf} h_{t-1} + b_f) \tag{7}$$

where, σ denotes the *sigmoid* activation function. Once f_t determines which memories to forget, the input gates (i_t) decide which cell states ($\widetilde{c_t}$) to update as per equations (8) and (9).

$$i_t = \sigma(W_{xi} x_t + W_{hi} h_{t-1} + b_i) \tag{8}$$

$$\widetilde{c_t} = tanh(W_{xc} x_t + W_{hc} h_{t-1} + b_{c_{t-1}}) \tag{9}$$

In equation (10), the old cell state (c_{t-1}) is updated to the new cell state (c_t) using forget gates (f_t) and input gates (i_t).

$$c_t = f_t \circ c_{t-1} + i_t \circ \widetilde{c_t} \tag{10}$$

where, $\circ$ is the Hadamard product. Finally, we filter the output values through output gates (o_t) based on the cell states (c_t) as per equations (11) and (12).

$$o_t = \sigma(W_{xo} x_t + W_{ho} h_t + b_o) \tag{11}$$

$$h_t = o_t \circ tanh(c_t) \tag{12}$$

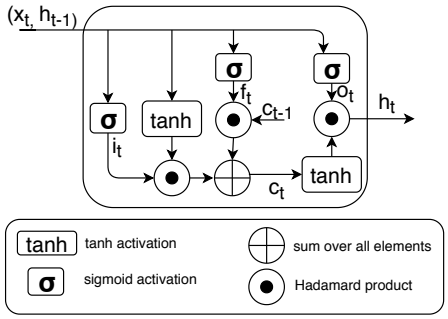


Fig. 2. LSTM Cell Architecture Used in the Proposed RNN Model

B. Convolutional Recurrent Neural Network

The recurrent neural networks using LSTM cells do not consider the spatial information encoded in the input-to-state or state-to-state transitions [20]. However, there are many applications where spatio-temporal correlation exists within the dataset. To address this issue, we use a convolution within the recurrent structure of the RNN. We first discuss the spatio-temporal property of RF data and then model a convolutional LSTM cell to exploit the same.

1) Spatio-temporal Property of I/Q Data: Suppose that a radio signal is represented as a time varying series over a spatial region using R rows and C columns. Here R represents the time varying nature of the signal and as such in our case it represents the total number of time stamps at which the signal was sampled (T in our case). C on the other hand represents the total number of features sampled at each time stamp (in our case its 2048 since there are 1024 features sampled each of dimension 2). Note that each cell corresponding to one value of R and one value of C represents a particular feature (I or Q) at a given point in time. In order to capture the spatial variation of the signal we need to consider samples from the signal at consecutive time stamps. In our case we consider time intervals of length γ and hence we represent this by sub-matrices of the original $R \times C$ matrix representing the whole time varying signal. Each of these sub-matrices have γ rows corresponding to the selected timestamps and C columns. Thus in a nutshell: to capture the temporal property only, we made use of a sequence of vectors for timestamps $1, 2, \cdots, T$, namely, $x_1, x_2, \cdots, x_T$ whereas now, to capture both spatial and temporal properties, we introduce a new vector $\chi_{t,t+\gamma}$, which is formulated as: $\chi_{t,t+\gamma} = [x_t, x_{t+1}, \cdots, x_{t+\gamma-1}]$. So the vector $\chi_{t,t+\gamma}$ eventually preserves the spatial properties with an increment of γ in time. Thus, we get a sequence of new vectors $\chi_{1,\gamma}, \chi_{\gamma,2\gamma}, \cdots \chi_{t,t+\gamma}, \cdots, \chi_{t+(\psi-1)\gamma,t+\psi\gamma}$, where ψ is $\lfloor R/\gamma \rfloor$. We formulate the probability of primary user to be idle ($P(s_{PU}(t) = 0|\chi_{t,\,t+\gamma})$) for the next γ-length sequence as:

$$P(s_{PU}(t) = 0|\chi_{t,\,t+\gamma}) = \frac{P(\chi_{t-\gamma,t}|s_{PU}(t) = 0)P(s_{PU}(t) = 0)}{P(\chi_{t-\gamma,t}|s_{PU}(t) = 0)P(s_{PU}(t) = 0) + P(\chi_{t-\gamma,t}|s_{PU}(t) = 1)P(s_{PU}(t) = 1)} \tag{13}$$

The marginal probabilities for the primary user to be idle or busy are modeled as $P(s_{PU}(t) = 0)$ and $P(s_{PU}(t) = 1)$ respectively. $P(\chi_{t-\gamma,\,t}|s_{PU}(t) = 0)$, and $P(\chi_{t-\gamma,\,t}|s_{PU}(t) = 1)$ are the conditional probabilities of $\chi_{t-\gamma,\,t}$ given $s_{PU}(t)$ was set to 0 and 1 respectively. The predicted value of γ-length sequence vector χ at timestamp t will depend on the predicted value of the previously predicted γ length sequence [20], which is given by:

$$\tilde{\chi}_{t,\,t+\gamma} = \underset{\chi_{t+1}\cdots\chi_{t+\gamma}}{arg\,max}\,(P(\chi_{t,\,t+\gamma}|\tilde{\chi}_{t-\gamma,\,t})) \tag{14}$$

It must be noted that in our case $\gamma = \Gamma$ where Γ is as described in Section III-C.

2) The ConvLSTM Model: We model the ConvLSTM cell as presented in Fig. 3. It is similar to an LSTM cell, but the input transformations and recurrent transformations are both convolutional in nature [20]. We formulate the input values, cell state and hidden states as 3-dimensional vectors, where the first dimension is the number of features ($C/2$) and varies over time, and the last two dimensions contain the spatial information (rows (R) and columns (C)).

We represent these as: (i) the inputs: $\chi_{1,\gamma}, \chi_{\gamma,2\gamma}, \cdots \chi_{t,t+\gamma}, \cdots, \chi_{t+(\psi-1)\gamma,t+\psi\gamma}$ (previously stated); (ii) cell outputs: $\mathcal{C}_1, \cdots, \mathcal{C}_t$, and (iii) hidden states: $\mathcal{H}_1, \cdots, \mathcal{H}_t$. We represent the gates in a similar manner as in the LSTM model. The parameters t, i_t, f_t, o_t, W, b have the same meaning as in section IV-A2. The key operations are defined in equations 15, 16, 17, 18, and 19. The probability of next γ-sequence to be in a particular class (from equation 13) is used within the implementation and execution of the model.

$$i_t = \sigma(W_{xi}\chi_{t,t+\gamma} + W_{hi}\mathcal{H}_{t-1} + b_i) \tag{15}$$

$$f_t = \sigma(W_{xf}\chi_{t,t+\gamma} + W_{hf}\mathcal{H}_{t-1} + b_f) \tag{16}$$

$$\mathcal{C}_t = f_t \circ \mathcal{C}_{t-1} + i_t . \tanh(W_{xc}\chi_{t,t+\gamma} + W_{hc}\mathcal{H}_{t-1} + b_c) \tag{17}$$

$$o_t = \sigma(W_{xo}\chi_{t,t+\gamma} + W_{ho}\mathcal{H}_{t-1} + b_o) \tag{18}$$

$$\mathcal{H}_t = o_t \circ \tanh(\mathcal{C}_t) \tag{19}$$

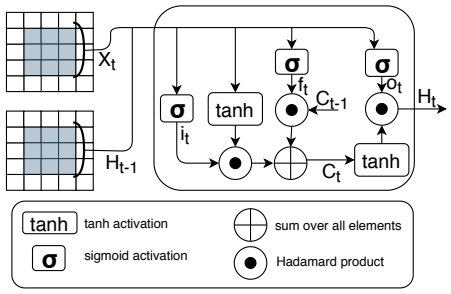


Fig. 3. ConvLSTM Cell Architecture Used in the Proposed RNN Model

C. Proposed PU Activity Prediction Model

We propose two models based on recurrent neural networks to train the spectrum sensor using the historical data of primary user activities. The objective is to learn every primary user's activity pattern and use it for scheduling SU activity. We also use the traditional linear regression model for prediction in order to test the performance of the neural network based

models against classical techniques. We present the training strategy of each model in Fig. 4. First, we divide the dataset into subsets for training and validation. In each epoch, we validate the trained model on the validation data and depending on the validation error we tune the hyper-parameters of each model. A single epoch consists of a forward pass and a backward pass through the implemented architecture for the entire dataset. We normalize the data for both training and validation to balance each feature. We design three different models using three different approaches on the normalized data.

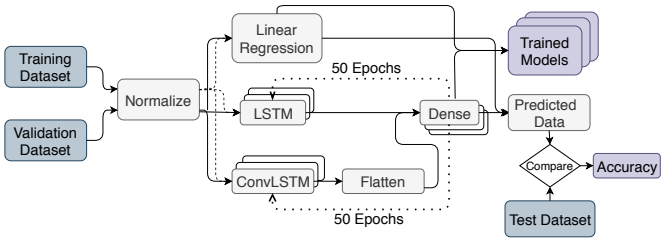


Fig. 4. PU Activity Prediction Training of the Spectrum Sensor

The details of the RNN with ConvLSTM cells is presented in Fig. 5. The first two hidden layers consist of *ConvLSTM* cells with 1024 and 256 filers respectively. The last two hidden layers are *Dense* layers, applied after flattening the output from the last *ConvLSTM* layer. We also apply Dropout [21] of 0.5 between the different layers to avoid over-fitting of the trained model. The LSTM implementation is also similar, with the exception of not having the *Flatten* layer in between the *LSTM* and *Dense* layers. Increasing the number of filters or layers did not help in achieving higher accuracy for either of the models. The output layer consists of a fully connected *Dense* layer with 1 neuron to generate the predicted value. We run and validate the training data for 50 epochs for both the LSTM and ConvLSTM models, beyond which we did not notice any further improvement. During the testing phase, once we get the predicted values, we compare them with the known values for the test/validation dataset and estimate the accuracy of the trained models.

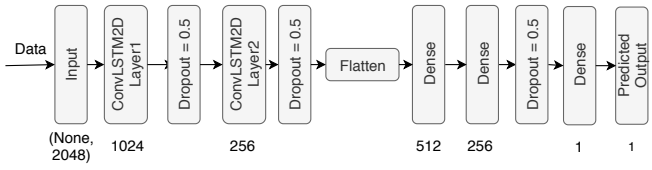


Fig. 5. RNN Implementation with ConvLSTM Cells for PU Activity Prediction

Next we present the conceptual overview of the deployment phase of the proposed prediction models for a cognitive radio network as shown in Fig. 6. The trained models are deployed within the spectrum sensor. The SS also collects and stores the historical data of all the primary users for enough time to build a robust trained model. This trained model also gets updated on the newly learned primary user's activity after some specific

Primary User	Size (GB)
PU#1	3.26
PU#2	3.5
PU#3	3.29
PU#4	3.2
PU#5	3.14
PU#6	3.16
PU#7	3.1
PU#8	3.22

TABLE I
COLLECTED DATASET SIZES FOR DIFFERENT PRIMARY USERS

duration of time. Determining the data collection time and model updating time are application specific. Once the spectrum sensor can get a primary user's absence prediction from the trained model, it relays the information to the secondary users. The secondary users later access the idle channel in a cooperative manner, details of which is beyond the scope of this article. Now we are ready to describe our experimental results.

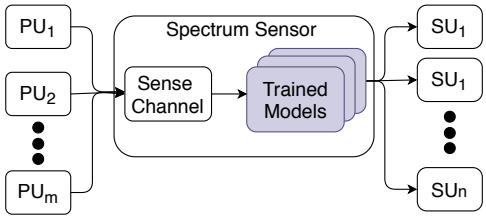


Fig. 6. Deployment of the Proposed PU Prediction Model

V. IMPLEMENTATION AND RESULTS

In this section we present the details of our implementation and the experimental results.

A. Experimental Environment

In order to validate the proposed models, we collected over-the-air data in an indoor lab environment from 8 universal software radio peripheral (USRP) B210s [7] acting as primary users. We name these 8 radios as PU#1 to PU#8. We collected the dataset on an i7 machine with 16 GB RAM. We conducted the proposed model training and evaluation on a Ryzen 8 Core system with 64 GB RAM, a GTX 1080 Ti GPU unit with 11 GB graphics memory.

B. Data Collection Environment

The data collection environment is presented in Fig. 7. A random signal is generated using GNURadio [22] and modulated with Quadrature Phase Shift Keying (QPSK). We programmed the USRP B210s to alternate between ON or OFF states such that the activity factor remained between 0.7 and 0.75. Thus, the collected dataset has more PU presence data than absence, which helps the learning process. The duration of ON and OFF times follows the model described in Section III-A. We generated datasets of different sizes for different PUs. Collected dataset sizes are presented in Table I.

The transmitted signal is received by a RTL-SDR [8] that used the *rtlsdr* python library. The AND gate (in Fig. 7) after the primary user block, is used to represent the fact that either

Parameters	Values
Transmitter Gain	45 dB
Transmitter Frequency	904 MHz (ISM)
Bandwidth	200 KHz
Sample Size	1024
Samples/Transmitter	Variable
Primary Users	USRP B210
Receiver	RTL-SDR
# Primary Users	8

TABLE II
PRIMARY USER TRANSMISSION CONFIGURATION PARAMETERS

the noise or the signal from primary user is transmitted at a particular timestamp. Primary user's OFF time is the absence of radio signal data, therefore it is represented as "Noise". We generate different datasets for all the primary users. The "over-the-air" transmission data was collected in an indoor lab environment where the B210 transmitters and the RTL-SDR receiver were at a distance of 10 feet with a direct line of sight. Thus, the underlying channel can be best modeled as a Rician fading channel. There was also multi-path effects due to the reflections from the walls.

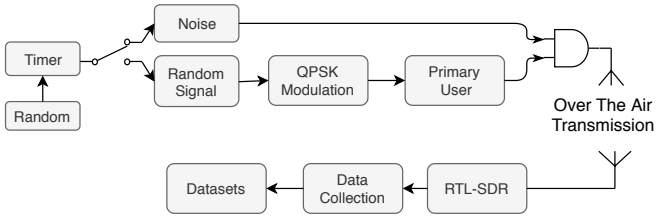


Fig. 7. Data Collection Procedure for each Primary User

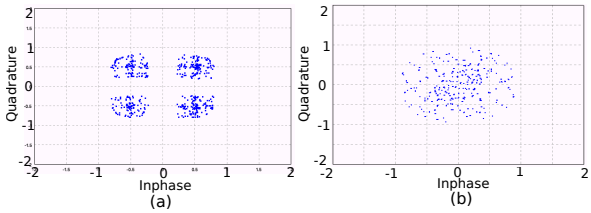


Fig. 8. I/Q Values Representation: (a) Signal Present (b) Noise

During the primary user's ON time, the I/Q values were organized in a constellation (please refer to Fig. 8(a)). During the primary user's OFF time the I/Q values are random (please refer to Fig. 8(b)). Each data sample had 2048 entities consisting of 1024 I and 1024 Q values. We chose 1024 as sample size as it was sufficient to capture the spatial properties and at the same time the training was not computationally intensive. The configuration parameters for the radios are given in Table II.

C. Signal to Noise Ratio of Data Collection Environment

To measure the signal to noise ratio (SNR) of the testbed environment, we use a RTL-SDR [8] dongle and Spektrum [23] which is an open source spectrum analyzer available for both Windows and Linux. The screenshots of Spektrum are shown in Fig. 9 and 10 when the signal is absent and present respectively. We calibrate the SNR using the Spektrum software (rather than using a spectrum analyzer) in order to avoid the associated costs and also in order to show the robustness of our methods to imprecise measurements (as measurements in software are always inferior to actual hardware measurements). From Fig. 9, we found that the noise floor was between -20 dB and -30 dB. Fig. 10 shows that the signal strength for the 200 KHz (from 904.9 MHz to 904.1 MHz) channel was between 0 dB and 10 dB. We set the transmitter gain to 45 dB and calculated the SNR as the difference between the noise floor and the signal strength. Our calculated SNR was 5 dB - (-25 dB) = 30 dB, with a 45 dB transmitter gain. It is to be noted that the signal strengths (in dB) of noise and signal measured by Spektrum is relative, but the difference between them is absolute.

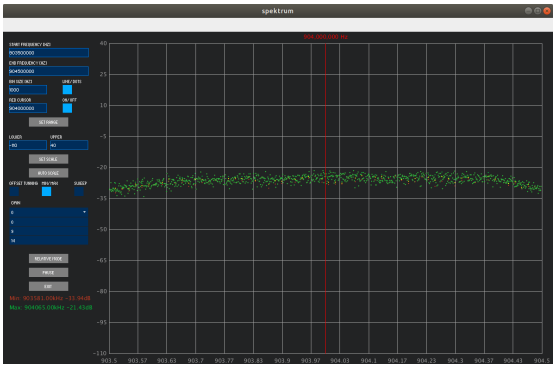


Fig. 9. Noise Floor Plot using Spektrum [23] Software

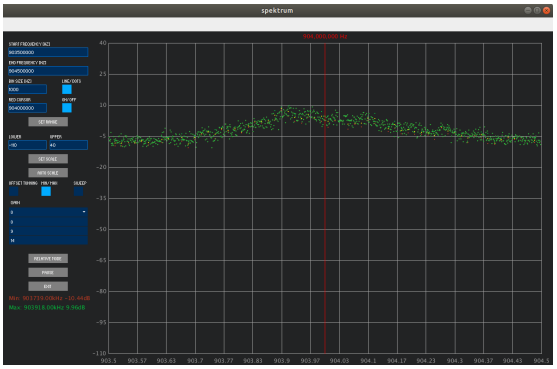


Fig. 10. Signal Level Plot using Spektrum [23] Software

D. Used Machine Learning Libraries and Performance Metrics

We used *Keras* [24] as the frontend and *Tensorflow* [25] as the backend for desiging the proposed neural network models. Keras is an overlay on neural network primitives with Tensorflow [25] or Theano [26] that provides a customizable interface for quick deployment of complex neural networks. We also use *Numpy*, *Scipy* and *Matplotlib* Python libraries for linear regression and other traditional methods.

"Accuracy" is used as the typical performance metric to measure the effectiveness of the proposed neural networks. However, accuracy can sometimes be misleading and incomplete when the data is skewed. In our dataset, the total PU ON time is greater than OFF times, making the dataset skewed.

A confusion matrix overcomes this problem by showing how confused the classification model is on its predictions. It provides more insights into the performance by identifying not only the number of errors, but also the types of those, i.e., false positives and false negatives.

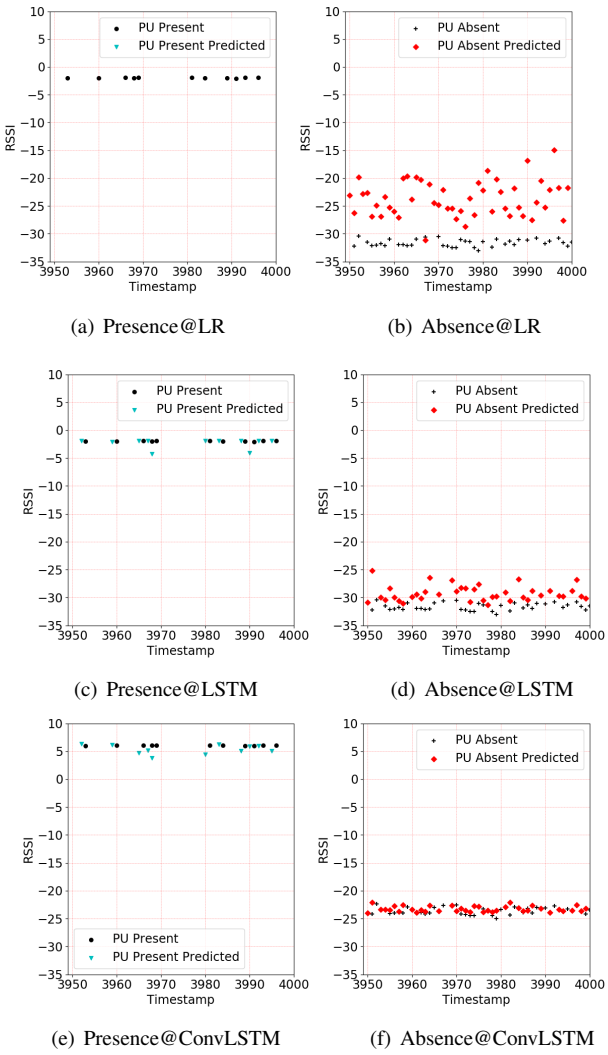
We use accuracy and confusion matrix to demonstrate the reliability of the proposed models. We show the interference violation and under-utilization to show the feasibility and applicability of the proposed models in cognitive radio networks.

E. Experimental Results

We train three different machine learning models on the collected dataset of each primary user. The linear regression model was straight forward and trained with 90% of each dataset. For both the LSTM and ConvLSTM based models, we use 90%, 5%, and 5% of data for training, validation, and testing respectively. During each training, we set the maximum epoch to 50 with an early stopping condition, such as, if there is no improvement of validation loss for consecutive 5 epochs, then the training is stopped. We choose 50 epochs because we observed through multiple runs of training, that each model reaches optimum accuracy within 50 epochs. We use Adam [27] based optimization with $10e-4$ learning rate and mean squared error loss for training both the LSTM and ConvLSTM based models. We adjusted the hyper-parameter's values such that the model gives the best possible accuracy with training time trade-off.

Once each model is trained, we predict each primary user's activity for the next 4000 timestamps using the proposed models. We only present last 50 timestamps of those 4000 timestamps in Fig 11, for better display quality and understanding. The plot demonstrates the primary user's presence and absence using a threshold on the received signal strength indicator (RSSI) for PU#1. The threshold is application specific; it is set to -10 dB for our testbed. Although PU#1's activity factor was set between 0.7 to 0.75, we see more absence for the last 50 timestamps, which shows the robustness of the data collection procedure. It is evident from the plots that the predicted values for both primary user's presence and absence are getting closer to actual values for LSTM and ConvLSTM models, however, ConvLSTM yields the best results. We observe similar results for the other primary users (PU#2 - P#8) as well.

1) Analysis on Proposed Prediction Models: The prediction accuracy for all the primary users (PU#1 - PU#8), for all the proposed models are presented in Table III. We notice that linear regression gives 73-76% accuracy, whereas the LSTM and ConvLSTM models manage to get 97%-99% accuracies respectively. This phenomenon can be justified by the presence of correlation within the recurrent structure of RF data. The linear regression based model does not leverage that property. The LSTM based model exploits only the temporal property of that recurrent structure giving a better accuracy (∼97%). However, ConvLSTM based model exploits the spatio-temporal property within the data and hence gives



(a) Presence@LR (b) Absence@LR

(c) Presence@LSTM (d) Absence@LSTM

(e) Presence@ConvLSTM (f) Absence@ConvLSTM

Fig. 11. Predictions of last few Timestamps for different Models for PU#1

the best accuracy(∼99% - 100%) among all the proposed prediction models for all the 8 primary users.

2) Performance of ConvLSTM Model: The accuracies for training and validation are presented in Fig. 12 for PU#1. It is observed that training and validation accuracy saturates within a few epochs of the start of training. The model behaves in the same manner for the other PUs as well. Once the ConvLSTM model is trained, we find the confusion matrices for all the 8 PUs for the next 4000 timestamps, as presented in Fig. 13. Since the PU's activity factor was set between 0.7 and 0.75, we notice a skewed behavior for the total presence and absence times of the primary users respectively. It is clear from the confusion matrices that the ConvLSTM based model yields negligible numbers of false positives and false negatives during the deployment phase.

3) Analysis on Interference Violations and Under-utilization: We calculate the total number of interference violations (IV) and under-utilization (UU) for each model. Table IV presents the total numbers and improvement of IV and UU after using all the proposed learning models, over using the conservative allocation strategies. It is clear that

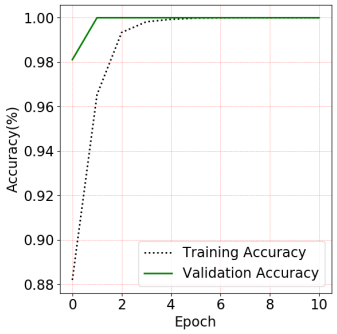
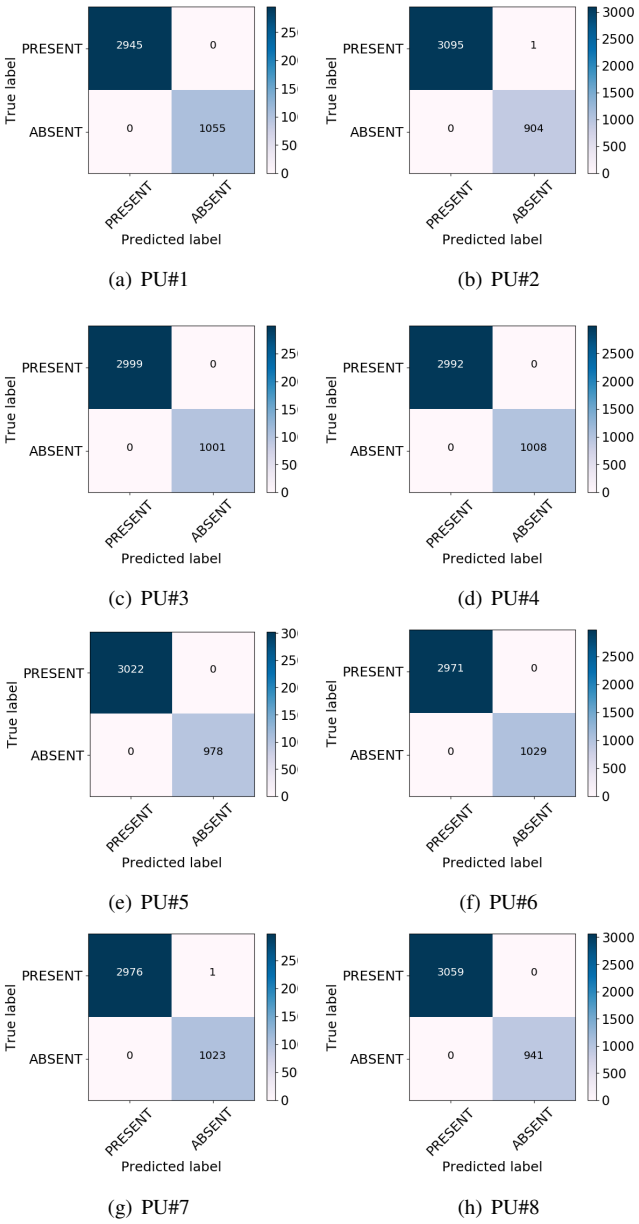


Fig. 12. Prediction accuracy for ConvLSTM



(a) PU#1 (b) PU#2

(c) PU#3 (d) PU#4

(e) PU#5 (f) PU#6

(g) PU#7 (h) PU#8

Fig. 13. Confusion Matrices for Prediction using ConvLSTM for 8 PUs

	Linear Regression	RNN (LSTM)	RNN (ConvLSTM)
PU #1	73.98%	97.62%	100%
PU #2	77.60%	97.83%	99.95%
PU #3	75.15%	97.92%	100%
PU #4	75.15%	97.60%	99.98%
PU #5	75.80%	97.88%	99.98%
PU #6	74.58%	97.62%	99.98%
PU #7	74.72%	97.75%	100%
PU #8	76.75%	97.72%	99.98%

TABLE III
ACCURACIES OF IMPLEMENTED MODELS FOR DIFFERENT PRIMARY USERS

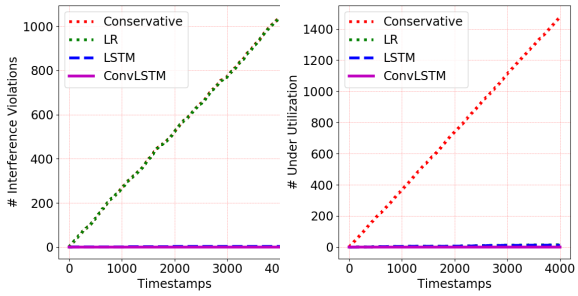
Techniques	Interference Violation (IV)	IV Change	Under Utilization (UU)	UU Change
Conservative	1043	-	1478	-
Linear Regression	1041	↓0.2%	15	↓98.9%
RNN-LSTM	7	↓99.3 %	6	↓99.5%
RNN-ConvLSTM	0	↓100 %	0	↓100%

TABLE IV
COMPARISON OF INTERFERENCE VIOLATION AND UNDER-UTILIZATION FOR THE PROPOSED MODELS FOR PU#1

the long-term prediction from ConvLSTM based model has no interference violation and under-utilization. Fig. 14 gives the graphical overview of how the IV and UU change as the time increases for all types of models. It is evident that the cumulative number of interference violation is significantly high for the conservative approach than the proposed learning based models. However, the cumulative under-utilization is high for both the conservative and linear regression models. The changes of LSTM and ConvLSTM based models are not quite visible in this figure, so we present the enhanced view of their changes in Fig. 15. In summary we have shown that, (1) Linear regression based model gives 73-76% accuracy for primary user's activity prediction. (2) LSTM based recurrent neural network model increases that accuracy to 97% for the activity prediction of all the 8 primary users. (3) ConvLSTM based RNN model gives the best accuracy ($\sim$100%) for the long-term prediction of primary user's activity. (4) The ConvLSTM based model also decreased the number of interference violations and under-utilizations by 100% compared to the conservative one. (5) Trained ConvLSTM based RNN models can be deployed in a central spectrum sensor for a robust and efficient cognitive radio network.

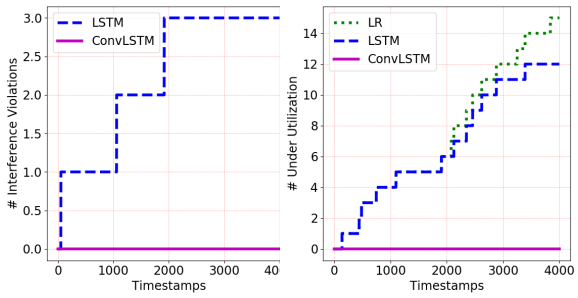
4) Computational Complexities: We focus on the computational time complexity for the training phase only, as the trained model gives the output within constant time ($O(1)$) during the deployment phase. Computing the time complexity for training a neural network is still evolving. In [28], the authors proved that a neural network of depth δ can be trained in $poly(s^{2^{\delta}})$, where s is the dimension of the input, and $poly(.)$ takes a polynomial time depending on the machine configuration. Here s depends on the dataset size.

Suppose each dataset has T samples. As mentioned earlier, we use 95% of data for training and validation purpose. The complexity for RNN models with 6 layers using 95% of T data samples for training and validation, is $poly(0.95 \times Te3^{2^6})$. For



(a) Interference Violations (b) Under-utilization

Fig. 14. Cumulative Interference Violations and Under-utilization for Conservative and all Proposed Models for PU#1



(a) Interference Violations (b) Under-utilization

Fig. 15. Enhanced Comparison of Cumulative Interference Violations and Under-utilization for the Proposed Models for PU#1

linear regression model, the complexity is $O(p^2T+p^3)$, where p is the number of features. The prediction complexity is $O(p)$.

VI. Conclusions

The *opportunistic usage* of spectrum by secondary users has the possibility of leading to an uniform and efficient usage of overly crowded radio frequency bands. In this regard we present the use of machine learning techniques to predict the possible *opportunities* for such spectrum usage by secondary users. We investigate the spatio-temporal aspect of over-the-air radio data for that purpose. The long-term pattern of primary user's ON and OFF times are learned by the proposed neural network models. The comparative analysis with linear regression shows that exploiting the recurrent structures with respect to temporal and spatial variations achieves the best possible accuracy, as seen from the proposed prediction models. Leveraging the memory of earlier transmission helps the sensing network to determine primary user activity pattern accurately over time. Successful implementation of the proposed models will improve spectrum utilization and lower interference violation for dynamic spectrum access networks.

References

[1] Rulemakings 12-354 and 17-258, "3.5 GHz Band / Citizens Broadband Radio Service," https://www.fcc.gov/wireless/bureau-divisions/mobility-division/35-ghz-band/35-ghz-band-citizens-broadband-radio-service, 2015.

[2] FCC ET Docket No. 08-260, "Second Report and Order and Memorandum Opinion and Order – Unlicensed Operation in the TV Broadcast Bands / Additional Spectrum for Unlicensed Devices Below 900 MHz and in the 3GHz Band," US Federal Communications Commission, Washington DC, FCC 08-260, 2008.

[3] Y. Zhao, L. Morales, J. Gaeddert, K. K. Bae, J. Um, and J. H. Reed, "Applying Radio Environment Maps to Cognitive Wireless Regional Area Networks," in *IEEE DySPAN*, 2007, pp. 115–118.

[4] M. J. Marcus, "Unlicensed Cognitive sharing of TV Spectrum: The Controversy at the Federal Communications Commission," *IEEE Communications Magazine*, vol. 43, no. 5, pp. 24–25, 2005.

[5] T. Yucek and H. Arslan, "A Survey of Spectrum Sensing Algorithms for Cognitive Radio Applications," *IEEE Communications Surveys Tutorials*, vol. 11, no. 1, pp. 116–130, 2009.

[6] X. Xing, T. Jing, W. Cheng, Y. Huo, and X. Cheng, "Spectrum prediction in cognitive radio networks," *IEEE Wireless Communications*, vol. 20, no. 2, pp. 90–96, 2013.

[7] Ettus Research, "USRP B210," https://www.ettus.com/product/details/UB210-KIT, 2018.

[8] NooElec, "USRP B210," http://www.nooelec.com/store/sdr/sdr-receivers/nesdr-mini-rtl2832-r820t.html, 2018.

[9] K. W. Sung, S. Kim, and J. Zander, "Temporal Spectrum Sharing Based on Primary User Activity Prediction," *IEEE Transactions on Wireless Communications*, vol. 9, no. 12, pp. 3848–3855, 2010.

[10] Y. Tang, Q. Zhang, and W. Lin, "Artificial Neural Network Based Spectrum Sensing Method for Cognitive Radio," in *Proceedings IEEE WiCOM*, 2010, pp. 1–4.

[11] V. K. Tumuluru, P. Wang, and D. Niyato, "A Neural Network Based Spectrum Prediction Scheme for Cognitive Radio," in *IEEE International Conference on Communications*, 2010, pp. 1–5.

[12] W. Lee, M. Kim, and D. Cho, "Deep Cooperative Sensing: Cooperative Spectrum Sensing Based on Convolutional Neural Networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 3, pp. 3005–3009, 2019.

[13] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[14] J. S. Ren, Y. Hu, Y.-W. Tai, C. Wang, L. Xu, W. Sun, and Q. Yan, "Look, Listen and Learn-A Multimodal LSTM for Speaker Identification," in *AAAI*, 2016, pp. 3581–3587.

[15] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional Radio Modulation Recognition Networks," in *Engineering Applications of Neural Networks*, 2016, pp. 213–226.

[16] T. Mukherjee *et al.*, "RSSI-Based Supervised Learning for Uncooperative Direction-Finding," in *Proceedings of ECML-PKDD*, ECML-PKDD. Springer, 10 2017.

[17] T. OShea and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017.

[18] D. Roy, T. Mukherjee, M. Chatterjee, and E. Pasiliao, "Detection of Rogue RF Transmitters using Generative Adversarial Nets," in *IEEE Wireless Communications and Networking Conference (WCNC)*, 2019.

[19] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[20] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo, "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, 2015, pp. 802–810.

[21] N. Srivastava *et al.*, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[22] GNURadio, "GNU Radio," https://www.gnuradio.org, 2018.

[23] Pavel Sorejs, "Spektrum," https://github.com/pavels/spektrum, 2015.

[24] F. Chollet *et al.*, "Keras: The Python Deep Learning library," https://keras.io, 2015.

[25] M. Abadi *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *CoRR*, 2016.

[26] R. Al-Rfou *et al.*, "Theano: A python framework for fast computation of mathematical expressions," *CoRR*, 2016.

[27] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *CoRR*, vol. abs/1412.6980, 2014.

[28] R. Livni, S. Shalev-Shwartz, and O. Shamir, "On the Computational Efficiency of Training Neural Networks," in *Advances in Neural Information Processing Systems*, 2014, pp. 855–863.